CoAP Protocol Binding
Intellectual Property Rights

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

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

Foreword

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

The present document will cover the protocol specific part of communication protocol used by oneM2M compliant systems as ‘RESTful CoAP binding’.

The scope of this specification is (not limited to as shown below):

- Binding oneM2M primitives to CoAP messages.
- Binding oneM2M Response Status Codes to CoAP Response Codes.
- Defining behaviour of a CoAP Client and Server depending on oneM2M parameters.

2 References

2.1 Normative references

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

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

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

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


2.2 Informative references

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

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

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

[i.1] oneM2M Drafting Rules.

NOTE: Available http://member.onem2m.org/Static_pages/Others/Rules_Pages/oneM2M-Drafting-Rules-V1_0.doc.
3 Abbreviations and acronyms

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

- **DTLS**: Datagram Transport Layer Security
- **HTTP**: Hyper Text Transfer Protocol
- **IP**: Internet Protocol
- **TCP**: Transport Control Protocol
- **TLS**: Transport Layer Security
- **TLV**: Tag - Length - Value (data structure)
- **UDP**: User Datagram Protocol
- **URI**: Uniform Resource Identifier

4 Conventions

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

5 Overview

The clause describes which features need to be supported in CoAP layer and introduces a message format and several features of CoAP used in this protocol binding specification.

5.1 Required Features

This clause explicitly specifies the required features of the CoAP layer for oneM2M to properly bind oneM2M primitives into CoAP messages.

- The 4-byte binary CoAP message header is defined in Section 3 of [1].
- Confirmable (CON), Acknowledgement (ACK) and Reset (RST) messages shall be supported. The Reset message is used to send an error message in response to a malformed Confirmable message in CoAP layer.
- GET, PUT, POST and DELETE methods shall be supported. oneM2M primitives map to these methods.
- A subset of Response Code specified in clause 6.2.4 shall be supported for oneM2M Response Status Code parameter mapping.
- The Uri-Host, Uri-Port, Uri-Path, and Uri-Query shall be supported.
- The Content-Type Option shall be used to indicate the media types of the payload.
- The Token Option may be used.
- Block-wise transfers feature may be supported to carry large payloads.
- Caching feature may be supported.
5.2 Introduction of CoAP

This clause describes a message format, and caching and block-wise transfers features which may be used to map oneM2M primitives to CoAP messages.

5.2.1 Message Format

This clause specifies details about the CoAP [1] message format:

- CoAP message occupies the data section of one UDP datagram.
- CoAP message format supports a 4-byte fixed-size header.
- Fixed-size header is followed by a Token value of length 0 or 8 bytes.
- The Token value is followed by a sequence of zero or more CoAP Options in TLV format.
- CoAP Options are followed by the payload part.

For more details on the CoAP message format and the supported header fields, refer [1].

5.2.2 Caching


5.2.2.1 Freshness

- CoAP server shall use Max-Age CoAP Option to specify the explicit expiration time for the CoAP Response's resource representation. This indicates that the response is not fresh after its age is greater than the specified number of seconds.
- Max-Age Option defaults to a value of 60 (seconds). In case, Max-Age Option is not present in the cacheable response, the response shall not be considered fresh after its age is greater than 60 seconds.
- The CoAP server shall set the Max-Age Option value to 0 (zero) to prevent or disable caching.
- The CoAP client, having a fresh stored response, can make new request matching the request for that stored response. In this case, the new response shall invalidate the old response.

5.2.2.2 Validity

- A CoAP endpoint with stored responses but not able to satisfy subsequent requests (for example, the response is not fresh), shall use the ETag Option to perform a conditional request to the CoAP server where the resource is hosted.
- If the cached response with the CoAP client is still valid, the server shall include the Max-Age Option in the response along with a code of 2.03 - Valid. This shall update the freshness of the cached response at the CoAP client.
- If the cached response with the CoAP client is not valid, the server shall respond with an updated representation of the resource with response code 2.05 - Content. The CoAP client shall use the updated response to satisfy request and may also replace/update the stored or cached response.

5.2.3 Blockwise Transfers

CoAP Block [3] Options shall be used when CoAP endpoints need to transfer large payloads e.g. firmware, software updates. Instead of relying on IP fragmentation, CoAP Block Option shall be used for transferring multiple blocks of information in multiple request-response pairs.
6 CoAP Message Mapping

6.1 Introduction

When AE or CSE binds oneM2M primitives to CoAP messages, or binds CoAP messages to oneM2M primitives, it is required that:

- AE shall host a CoAP client and should host a CoAP server; or
- CSE shall host both a CoAP client and a CoAP server.

Basically single oneM2M request primitive is mapped to single CoAP request message, and single oneM2M response primitive is mapped to single CoAP response message. However, single oneM2M request/response primitive is mapped to multiple CoAP request/response messages respectively when CoAP block-wise transfers feature is used.

Mapping between CoAP message and oneM2M primitive shall be applied in the following cases:

- when the Originator sends a request primitive;
- when the Receiver receives a CoAP message(s);
- when the Receiver sends a response primitive;
- when the Originator receives a CoAP message(s).

The following clauses specify how to map each oneM2M primitive parameter defined in [2] to a corresponding CoAP message field to compose a CoAP request/response message.

6.2 Primitive Mapping to CoAP Message

This clause describes where to map oneM2M parameters in a primitive to header, Option and payload fields in a CoAP message.

6.2.1 Header

This clause specifies how to configure CoAP header information:

- The Version field shall be configured as 1.
- The Type field shall be configured according to clause 6.3. The Reset message is used to send a error message in response to a malformed Confirmable message in CoAP layer.
- In case of a request, the Code field indicates CoAP Method. The oneM2M Operation parameter shall be mapped to a CoAP Method according to the table 6.2.1-1.
- In case of a response, the Code field indicates CoAP Response Code. The oneM2M Response Status Code parameter shall be mapped to CoAP Response Code as specified in clause 6.2.4.

The configurations of Token Length and Message ID are left to implementation.

Table 6.2.1-1: oneM2M Operation Parameter Mapping

<table>
<thead>
<tr>
<th>oneM2M Operation Parameter</th>
<th>CoAP Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATE</td>
<td>POST</td>
</tr>
<tr>
<td>RETRIEVE</td>
<td>GET</td>
</tr>
<tr>
<td>UPDATE</td>
<td>PUT</td>
</tr>
<tr>
<td>DELETE</td>
<td>DELETE</td>
</tr>
<tr>
<td>NOTIFY</td>
<td>POST</td>
</tr>
</tbody>
</table>
At the Receiver, CoAP request message with POST method shall be mapped to oneM2M CREATE or NOTIFY Operation parameter in accordance with the existence of Resource Type parameter. If Resource Type parameter exists then value of the Operation parameter is CREATE and if Resource Type parameter doesn't exist, the value of Operation parameter is NOTIFY.

6.2.2 Configuration of Token and Options

This clause describes configuration of Token and Options based on oneM2M parameters.

6.2.2.1 Token

Due to size limitation, Request Identifier is not mapped to Token option. However, Token may be used in CoAP layer to match a CoAP request and response.

6.2.2.2 Content Format Negotiation Options

The CoAP Accept Option may be used to indicate which Content-Format is acceptable to an Originator. If a Hosting CSE supports the Content-Format specified in Accept Option of the request, the Hosting CSE shall respond with that Content-Format. If the Hosting CSE doesn't support the Content-Format specified in Accept Option of the request, 4.06 “Not Acceptable” shall be sent as a response, unless another error code takes precedence for this response.

Possible values for Content-Format and Accept options are listed below:

- application/xml.
- application/json.
- media types specified in clause 6.7 “oneM2M specific MIME media types” of [2].

6.2.2.3 URI Options

This clause describes how to configure CoAP Uri-Host, Uri-Port, Uri-Path, and Uri-Query Options.

Host and port part of the address specified in pointOfAccess attribute of <remoteCSE> resource shall be mapped to Uri-Host and Uri-Port respectively.

To parameter shall be mapped to Uri-Path.

Resource Type, responseType element of Response Type, Result Persistence, Delivery Aggregation, Result Content, parameters of Filter Criteria, and Discovery Result Type parameters shall be carried in Uri-Query Option in a short name form as specified in clause 8.2.2 [2].

6.2.2.4 Definition of New Options

This clause describes new CoAP Options used for binding several oneM2M request/response parameters. The table 6.2.2.4-1 contains definitions of the new CoAP Options and sub-clauses specify oneM2M parameter mapping with the newly defined CoAP Options in the table 6.2.2.4-1.
Table 6.2.2.4-1: Definition of New Options

<table>
<thead>
<tr>
<th>No</th>
<th>C</th>
<th>U</th>
<th>N</th>
<th>R</th>
<th>Name</th>
<th>Format</th>
<th>Length</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>256</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>oneM2M-FR</td>
<td>string</td>
<td>0-255</td>
<td>(None)</td>
</tr>
<tr>
<td>257</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>oneM2M-RQI</td>
<td>string</td>
<td>0-255</td>
<td>(None)</td>
</tr>
<tr>
<td>258</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oneM2M-NM</td>
<td>string</td>
<td>0-255</td>
<td>(None)</td>
</tr>
<tr>
<td>259</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oneM2M-OT</td>
<td>string</td>
<td>15</td>
<td>(None)</td>
</tr>
<tr>
<td>260</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oneM2M-RQET</td>
<td>string</td>
<td>15</td>
<td>(None)</td>
</tr>
<tr>
<td>261</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oneM2M-RSET</td>
<td>string</td>
<td>15</td>
<td>(None)</td>
</tr>
<tr>
<td>262</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oneM2M-OET</td>
<td>string</td>
<td>15</td>
<td>(None)</td>
</tr>
<tr>
<td>263</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oneM2M-RTURI</td>
<td>string</td>
<td>0-255</td>
<td>(None)</td>
</tr>
<tr>
<td>264</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oneM2M-EC</td>
<td>uint</td>
<td>1</td>
<td>(None)</td>
</tr>
<tr>
<td>265</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oneM2M-RSC</td>
<td>uint</td>
<td>2</td>
<td>(None)</td>
</tr>
<tr>
<td>266</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oneM2M-GID</td>
<td>string</td>
<td>0-255</td>
<td>(None)</td>
</tr>
</tbody>
</table>

NOTE: Table 6.2.4-1 follows the template used in section 5.10 Option Definitions of CoAP specification [1].

6.2.2.4.1 From

The *From* parameter shall be mapped to the oneM2M-FR Option.

6.2.2.4.2 Request Identifier

The *Request Identifier* parameter shall be mapped to the oneM2M-RQI Option.

6.2.2.4.3 Name

The *Name* parameter shall be mapped to the oneM2M-NM Option.

6.2.2.4.4 Originating Timestamp

The *Originating Timestamp* parameter shall be mapped to the oneM2M-OT Option.

6.2.2.4.5 Request Expiration Timestamp

The *Request Expiration Timestamp* parameter shall be mapped to the oneM2M-RQET Option.

6.2.2.4.6 Result Expiration Timestamp

The *Request Expiration Timestamp* parameter shall be mapped to the oneM2M-RSET Option.

6.2.2.4.7 Operation Execution Time

The *Operation Execution Time* parameter shall be mapped to the oneM2M-OET Option.

6.2.2.4.8 notificationURI of Response Type

The notificationURI element of *Response Type* parameter shall be mapped to the oneM2M-RTURI Option.

6.2.2.4.9 Event Category

The *Event Category* parameter shall be mapped to the oneM2M-EC Option.

6.2.2.4.10 Response Status Code

The *Response Status Code* parameter shall be mapped to the oneM2M-RSC Option.

6.2.2.4.11 Group Request Identifier

The *Group Request Identifier* parameter shall be mapped to the oneM2M-GID Option.
6.2.3 Payload

*Content* parameter shall be mapped to CoAP payload. Blockwise transfers mechanism may be used to deliver large size of *Content* parameter which is not fit into one CoAP message. Please refer to clause 6.5 for the detail information. If *Content* parameter contains URI and resource representation in a response to a create request, URI shall be mapped to Location-Path Option.

6.2.4 Response Codes Mapping

Table 6.2.4-1 defines a mapping between oneM2M *Response Status Code* parameter specified in [2] and CoAP Response Code.

In case of where multiple oneM2M *Response Status Code* parameters are mapped to single CoAP Status Code, *Response Status Code* parameter shall be specified in oneM2M-RSC Option.

<table>
<thead>
<tr>
<th>oneM2M Response Status Code</th>
<th>Description</th>
<th>Status Code of CoAP</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>ACCEPTED</td>
<td>None</td>
<td>Empty Acknowledgement Message shall be used</td>
</tr>
<tr>
<td>2000</td>
<td>OK</td>
<td>2.05</td>
<td>Content</td>
</tr>
<tr>
<td>2001</td>
<td>CREATED</td>
<td>2.01</td>
<td>Created</td>
</tr>
<tr>
<td>2002</td>
<td>DELETED</td>
<td>2.02</td>
<td>Deleted</td>
</tr>
<tr>
<td>2004</td>
<td>CHANGED</td>
<td>2.04</td>
<td>Changed</td>
</tr>
<tr>
<td>4000</td>
<td>BAD_REQUEST</td>
<td>4.00</td>
<td>Bad Request</td>
</tr>
<tr>
<td>4004</td>
<td>NOT_FOUND</td>
<td>4.04</td>
<td>Not Found</td>
</tr>
<tr>
<td>4005</td>
<td>OPERATION_NOT_ALLOWED</td>
<td>4.05</td>
<td>Method Not Allowed</td>
</tr>
<tr>
<td>4008</td>
<td>REQUEST_TIMEOUT</td>
<td>4.04</td>
<td>Not Found</td>
</tr>
<tr>
<td>4101</td>
<td>SUBSCRIPTION_CREATOR_HAS_NO_PRIVILEGE</td>
<td>4.03</td>
<td>Forbidden</td>
</tr>
<tr>
<td>4102</td>
<td>CONTENTS_UNACCEPTABLE</td>
<td>4.00</td>
<td>Bad Request</td>
</tr>
<tr>
<td>4103</td>
<td>ACCESS_DENIED</td>
<td>4.03</td>
<td>Forbidden</td>
</tr>
<tr>
<td>4104</td>
<td>GROUP_REQUEST_IDENTIFIER_EXISTS</td>
<td>4.00</td>
<td>Bad Request</td>
</tr>
<tr>
<td>4105</td>
<td>CONFLICT</td>
<td>4.03</td>
<td>Forbidden</td>
</tr>
<tr>
<td>5000</td>
<td>INTERNAL_SERVER_ERROR</td>
<td>5.00</td>
<td>Internal Server Error</td>
</tr>
<tr>
<td>5001</td>
<td>NOT_IMPLEMENTED</td>
<td>5.01</td>
<td>Not Implemented</td>
</tr>
<tr>
<td>5103</td>
<td>TARGET_NOT_REACHABLE</td>
<td>4.04</td>
<td>Not Found</td>
</tr>
<tr>
<td>5105</td>
<td>NO_PRIVILEGE</td>
<td>4.03</td>
<td>Forbidden</td>
</tr>
<tr>
<td>5106</td>
<td>ALREADY_EXISTS</td>
<td>4.00</td>
<td>Bad Request</td>
</tr>
<tr>
<td>5203</td>
<td>TARGET_NOT_SUBSCRIBABLE</td>
<td>4.03</td>
<td>Forbidden</td>
</tr>
<tr>
<td>5204</td>
<td>SUBSCRIPTION_VERIFICATION_INITIATION_FAILED</td>
<td>5.00</td>
<td>Internal Server Error</td>
</tr>
<tr>
<td>5205</td>
<td>SUBSCRIPTION_HOST_HAS_NO_PRIVILEGE</td>
<td>4.03</td>
<td>Forbidden</td>
</tr>
<tr>
<td>5206</td>
<td>NON_BLOCKING_REQUEST_NOT_SUPPORTED</td>
<td>5.00</td>
<td>Internal Server Error</td>
</tr>
<tr>
<td>6003</td>
<td>EXTERNAL_OBJECT_NOT_REACHABLE</td>
<td>4.04</td>
<td>Not Found</td>
</tr>
<tr>
<td>6005</td>
<td>EXTERNAL_OBJECT_NOT_FOUND</td>
<td>4.04</td>
<td>Not Found</td>
</tr>
<tr>
<td>6010</td>
<td>MAX_NUMBER_OF_MEMBER_EXCEEDED</td>
<td>4.00</td>
<td>Bad Request</td>
</tr>
<tr>
<td>6011</td>
<td>MEMBER_TYPE_INCONSISTENT</td>
<td>4.00</td>
<td>Bad Request</td>
</tr>
<tr>
<td>6020</td>
<td>MGMT_SESSION_CANT_BE_ESTABLISHED</td>
<td>5.00</td>
<td>Internal Server Error</td>
</tr>
<tr>
<td>6021</td>
<td>MGMT_SESSION_ESTABLISHMENT_TIMEOUT</td>
<td>5.00</td>
<td>Internal Server Error</td>
</tr>
<tr>
<td>6022</td>
<td>INVALID_CMDTYPE</td>
<td>4.00</td>
<td>Bad Request</td>
</tr>
<tr>
<td>6023</td>
<td>INVALID_ARGUMENTS</td>
<td>4.00</td>
<td>Bad Request</td>
</tr>
<tr>
<td>6024</td>
<td>INSUFFICIENT_ARGUMENTS</td>
<td>4.00</td>
<td>Bad Request</td>
</tr>
<tr>
<td>6025</td>
<td>MGMT_CONVERSION_ERROR</td>
<td>5.00</td>
<td>Internal Server Error</td>
</tr>
<tr>
<td>6026</td>
<td>MGMT_CANCELLATION_FAILURE</td>
<td>5.00</td>
<td>Internal Server Error</td>
</tr>
<tr>
<td>6028</td>
<td>ALREADY_COMPLETE</td>
<td>4.00</td>
<td>Bad Request</td>
</tr>
<tr>
<td>6029</td>
<td>COMMAND_NOT_CANCELLABLE</td>
<td>4.00</td>
<td>Bad Request</td>
</tr>
</tbody>
</table>

The Receiver decides the *Response Status Code* parameter using the combination of CoAP Response Code and oneM2M-RSC Option information.
6.3 Accessing Resources in CSEs

This clause describes the behavior of CoAP layer depending on Response Type parameter.

6.3.1 Blocking case

- If Response Type parameter is configured as "blockingRequest" (blocking case), the Originator (CoAP client) shall use the Confirmable Method for the resource to the Receiver (CoAP server).
- In case of successful processing of the request at the Receiver, the Receiver shall piggyback the response with an appropriate response code in the Acknowledgment message that acknowledges the Confirmable request.

6.3.2 Non-Blocking Asynchronous case

- If Response Type parameter is configured as "nonBlockingRequestAsynch" (non-blocking asynchronous case), the Originator (CoAP client) shall use the Confirmable Method for the resource to the Receiver (CoAP server). Originator shall provide a unique Token value in the request.
- The Receiver shall provide acknowledgment of receipt of the request using Acknowledgment message.
- The Receiver, upon successful processing of the request, shall send an appropriate response in a separate Confirmable message. The Originator shall acknowledge the Confirmable response.

6.3.3 Non-Blocking Synchronous case

- If Response Type parameter is configured as "nonBlockingRequestSynch" (non-blocking synchronous case), the Originator (CoAP client) shall use the Confirmable Method for the resource to the Receiver (CoAP server). Originator shall provide a unique Token value in the request.
- The Receiver shall provide an acknowledgment of receipt of the request using Acknowledgment message.
- The Receiver, after validating the request and before processing it fully, shall send an appropriate response including a reference in a separate Confirmable message. The Originator shall acknowledge the Confirmable response. Alternatively, if possible for the Receiver, the response can be piggy-backed with acknowledgment message in the previous step.
- The Originator can use the reference or the token to synchronously access or retrieve the resource. The Receiver, upon receipt of the request, shall respond with the current state of the resource.

NOTE: If the Receiver is a Transit CSE, the Receiver acts as CoAP client and CoAP server.

6.4 Mapping rules of caching

This clause specifies how to enable or disable CoAP caching mechanism and how to use cached information.

If the CoAP end point supports caching mechanism by freshness, the CoAP server shall:

- set the Max-Age Option value to "0" (zero) to disable caching, in order to support complete oneM2M mapping; or
- set the Max-Age option value to another value (such as the default value), in order to use CoAP caching mechanism for constrained environment.

NOTE 1: In the second case, the new request from oneM2M layer can get the stored fresh response from CoAP client, not from CoAP server.

If the CoAP end point supports caching mechanism by validity:

- the CoAP server shall not present Etag in responses to disable caching, in order to support complete oneM2M mapping; or
the CoAP server shall present Etag in responses, in order to use CoAP caching mechanism for constrained environment.

NOTE 2: In the second case, the new request from oneM2M layer can get the stored fresh response from CoAP server, not from oneM2M layer.

6.5 Usage of Blockwise Transfers

Using Block Options, large oneM2M resource representations can be fragmented and reassembled by CoAP independently of the lower layers as well as the above application. The CoAP Block1 Option shall be used to define the size of the blocks used for oneM2M request primitives and the CoAP Block2 Option shall be used to define the size of the blocks used for oneM2M response responses. Refer [3] for further details.

7 Security Consideration

CoAP itself does not provide protocol primitives for authentication or authorization; where this is required, it shall be provided by DTLS.

Just as HTTP is secured using Transport Layer Security (TLS) over TCP, CoAP shall be secured using Datagram TLS (DTLS) [5].

All CoAP messages shall be sent as DTLS "application data". For matching an ACK or RST to a CON message or a RST to a NON message: The DTLS session shall be the same and the epoch shall be the same.

For matching a response to a request, the DTLS session shall be the same and the epoch shall be the same. The response to a DTLS secured request shall always be DTLS secured using the same security session and epoch.

OneM2M primitive parameters contained in CoAP messages may be protected by DTLS in a hop-by-hop manner. For the details, see oneM2M security solution specification [4].

NOTE: Some provisioning schemes of ETSI TS 118 103 [4] enable the provisioning of end-to-end credentials, but protocols to establish security associations between non-adjacent nodes are not addressed by oneM2M in the present release.
Annex A (informative):
Example Procedures

A.1 AE Registration

Figure A.1-1 illustrates CoAP mapping of AE Registration procedure described in clause 7.3.5.2.1 of ETSI TS 118 104 [2].

Step1: IN-AE creates <AE> resource at IN-CSE
Method: POST
Uri-Path: coap://cse02.m2m.sp1.com/CSE02Base
Uri-Query: ty=2
Content-Formats: application/vnd.onem2m-res+xml
Payload: <AE> representation to create

Step2: Check access controls, process request and create <AE> resource for IN-AE

Step3: Response
Code: 2.01 Created
Location-path: coap://cse02.m2m.sp1.com/CSE02Base/ae02
Content-Formats: application/vnd.onem2m-res+xml
Payload: <AE> representation created

Figure A.1-1: Binding Example - AE Registration
## History

<table>
<thead>
<tr>
<th>Document history</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1.0.0</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>