TTC仕様書
Technical Specification

TS-M2M-0008v4.3.0

サービス層 API 仕様（CoAP 用）

CoAP Protocol Binding

2023 年 3 月 17 日制定

一般社団法人
情報通信技術委員会

THE TELECOMMUNICATION TECHNOLOGY COMMITTEE

TS-M2M-0008v4.3.0

サービス層 API 仕様（CoAP 用）　[CoAP Protocol Binding]

＜参考＞ [Remarks]

１．英文記述の適用レベル [Application level of English description]

適用レベル [Application level]：E2

本標準の本文、付属資料および付録の文章および図に英文記述を含んでいる。

[English description is included in the text and figures of main body, annexes and appendices.]

２．国際勧告等の関連 [Relationship with international recommendations and standards]

本標準は、oneM2M で承認された Technical Specification TS-0008-V4.3.0 に準拠している。

[This standard is standardized based on the Technical Specification TS-0008-V4.3.0 approved by oneM2M.]

３．上記国際勧告等に対する追加項目等 [Departures from international recommendations]

原標準に対する変更項目 [Changes to original standard]

原標準が参照する標準のうち、TTC 標準に置き換える項目。[Standards referred to in the original standard, which are replaced by TTC standards.]

原標準が参照する標準のうち、それらに準拠した TTC 標準等が制定されている場合は自動的に最新版 TTC 標準等に置き換え参照するものとする。　[Standards referred to in the original standard should be replaced by derived TTC standards.]

４．工業所有権 [IPR]

本標準に関わる「工業所有権等の実施の権利に係る確認書」の提出状況は、ＴＴＣホームページによる。

[Status of “Confirmation of IPR Licensing Condition” submitted is provided in the TTC web site.]

５．作成専門委員会 [Working Group]

oneM2M 専門委員会 [oneM2M Working Group]

oneM2M Technical Specification	
Document Number	TS-0008- V-4.3.0
Document Name:	CoAP Protocol Binding
Date:	2022-10-24
Abstract:	The specification will cover the protocol specific part of communication protocol used by oneM2M compliant systems as 'CoAP binding'

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

Contents

1 Scope 5

2 References 5
2.1 Normative references 5
2.2 Informative references 5

3 Abbreviations and acronyms 5

4 Conventions 6

5 Overview 6
5.0 Introduction 6
5.1 Required Features 6
5.2 Introduction to CoAP 6
5.2.0 Introduction 6
5.2.1 Message Format 7
5.2.2 Caching 7
5.2.2.0 Introduction 7
5.2.2.1 Freshness 7
5.2.2.2 Validity 7
5.2.3 Blockwise Transfers 7

6 CoAP Message Mapping 8
6.1 Introduction 8
6.2 Primitive Mapping to CoAP Message 8
6.2.0 Introduction 8
6.2.1 Header 8
6.2.2 Configuration of Token and Options 9
6.2.2.0 Introduction 9
6.2.2.1 Token 9
6.2.2.2 Content Format Negotiation Options 9
6.2.2.3 URI Options 10
6.2.2.4 Definition of New Options 10
6.2.2.4.0 Introduction 10
6.2.2.4.1 From 11
6.2.2.4.2 Request Identifier 11
6.2.2.4.3 Void 11
6.2.2.4.4 Originating Timestamp 11
6.2.2.4.5 Request Expiration Timestamp 11
6.2.2.4.6 Result Expiration Timestamp 11
6.2.2.4.7 Operation Execution Time 11
6.2.2.4.8 notificationURI of Response Type 11
6.2.2.4.9 Event Category 12
6.2.2.4.10 Response Status Code 12
6.2.2.4.11 Group Request Identifier 12
6.2.2.4.12 Resource Type 12
6.2.2.4.13 Content Offset 12
6.2.2.4.14 Content Status 12
6.2.2.4.15 Assigned Token Identifiers 12
6.2.2.4.16 Release Version Indicator 12
6.2.2.4.17 Vendor Information 12
6.2.2.4.18 Group Request Target Members 12
6.2.2.4.19 Authorization Signatures 13
6.2.2.4.20 Authorization Signature Request Information 13
6.2.2.4.21 Ontology Mapping Resources 13
6.2.2.4.22 Primitive Profile Identifier 13
6.2.2.4.22 M2M Service User 13
6.2.3 Payload 13
6.2.4 Response Codes Mapping 13
6.3 Accessing Resources in CSEs 16

6.3.0 Introduction 16
6.3.1 Blocking case 16
6.3.2 Non-Blocking Asynchronous case 17
6.3.3 Non-Blocking Synchronous case 18
6.3.4 Flex Blocking case 18
6.4 Mapping rules of caching 19
6.5 Usage of Blockwise Transfers 19
7 Security Consideration 19
Annex A (informative): Example Procedures 20
A.1 Blocking case of AE Registration 20
A.2 Non-blocking synchronous case of AE Registration 21
Annex B (normative): Multicast group fan out procedure 22
B.0 Introduction 22
B.1 Security 22
B.2 Caching 23
History 24

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 the present document 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 referenced document (including any amendments) applies.

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

[1] IETF RFC 7252: "The Constrained Application Protocol (CoAP)".

[2] oneM2M TS-0004: " Service Layer Core Protocol Specification".

[3] IETF RFC 7959: "Block-Wise Transfers in the Constrained Application Protocol (CoAP)".

[4] oneM2M TS-0003: "Security solutions".

[5] IETF RFC 6347: "Datagram Transport Layer Security Version 1.2".

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.

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://www.onem2m.org/images/files/oneM2M-Drafting-Rules.pdf.

3 Abbreviations and acronyms

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

ACK	CoAP Acknowledgement message
AE	Application Entity
CON	CoAP Confirmable message
CSE	Common Service Entity
DTLS	Datagram Transport Layer Security
HTTP	Hyper Text Transfer Protocol
IANA	Internet Assigned Numbers Authority
IP	Internet Protocol

NON	CoAP Non-confirmable message
RST	CoAP ReSeT message
TCP	Transport Control Protocol
TLS	Transport Layer Security
TLV	Tag - Length - Value (data structure)
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
XML	eXtensible Markup Language

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

5.0 Introduction

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 IETF RFC 7252 [1].
- Confirmable (CON), Non-confirmable (NON), 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.
- The CoAP Response Codes 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.
- Block-wise transfers feature may be supported to carry large payloads.
- The Caching feature may be supported.

5.2 Introduction to CoAP

5.2.0 Introduction

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 to 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 to IETF RFC 7252 [1].

5.2.2 Caching

5.2.2.0 Introduction

CoAP [1] supports caching of responses to fulfil future equivalent requests to the same resource. Caching is supported using freshness and validity information carried with CoAP [1] responses.

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 may be used when CoAP endpoints need to transfer large payloads e.g. firmware, software updates. Instead of relying on IP fragmentation, CoAP Block Option should 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 sub-clauses specify how to map each oneM2M primitive parameter defined in oneM2M TS-0004 [2] to a corresponding CoAP message field to compose a CoAP request/response message.

6.2 Primitive Mapping to CoAP Message

6.2.0 Introduction

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 indicate an error in response to a malformed message in the CoAP layer.
- In case of a request, the Code field indicates the CoAP Method. If the oneM2M operation is sent as a Blocking request the oneM2M ***Operation*** parameter shall be mapped to a CoAP Method according to the table 6.2.1-1. In non-blocking and flex blocking cases, the request shall use the CoAP POST method, and the Operation parameter shall be mapped as described in clause 6.2.2.3.
- In case of a response, the Code field indicates the CoAP Response Code. The oneM2M ***Response Status Code*** parameter shall be mapped to a CoAP Response Code as specified in clause 6.2.4.
- The Originator and Receiver shall set the 16 bit MessageId in accordance with the CoAP specification [1] and shall retry transmission of all unacknowledged Confirmable messages, as required by that specification.

Table 6.2.1-1: oneM2M Operation Parameter Mapping

oneM2M Operation Parameter	CoAP Method	CoAP Method Code
CREATE	POST	0.02
RETRIEVE	GET	0.01
UPDATE	PUT	0.03
DELETE	DELETE	0.04
NOTIFY	POST	0.02

At the Receiver, a CoAP request message with a POST method that does not carry an ***Operation*** parameter shall be mapped to a oneM2M CREATE or NOTIFY operation in accordance with the existence of the ***Resource Type*** parameter. If a ***Resource Type*** parameter exists then the value of the ***Operation*** parameter is CREATE and if the ***Resource Type*** parameter does not exist, the value of the ***Operation*** parameter is NOTIFY.

6.2.2 Configuration of Token and Options

6.2.2.0 Introduction

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

6.2.2.1 Token

The CoAP token is used by the CoAP layer to match a response to a request, in a manner that is similar to the oneM2M ***Request Identifier***. Due to size limitations, a Request Identifier cannot be used directly as the CoAP Token.

The use of tokens by Originator and Receiver shall comply with requirements of the CoAP specification [1].

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 does not 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 (41);
- application/json (50);
- application/cbor (60);
- media types specified in clause 6.7 "oneM2M specific MIME media types" of oneM2M TS-0004 [2].

Numeric values for oneM2M defined media types are listed in table 6.2.2.2-1.

Table 6.2.2.2-1: CoAP oneM2M Specific Content-Formats

oneM2M Specific Media Type	ID
vnd.onem2m-res+xml	10014
vnd.onem2m-res+json	10001
vnd.onem2m-ntfy+xml	10002
vnd.onem2m-ntfy+json	10003
vnd.onem2m-preq+xml	10006
vnd.onem2m-preq+json	10007
vnd.onem2m-prsp+xml	10008
vnd.onem2m-prsp+json	10009
vnd.onem2m-res+cbor	10010
vnd.onem2m-ntfy+cbor	10011
vnd.onem2m-preq+cbor	10012
vnd.onem2m-prsp+cbor	10013

NOTE: ID values for oneM2M specific media type are subject to change after IANA registration.

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.

If ***To*** parameter contains absolute format, then the first URI-Path Option shall contain a letter "_" and map ***To*** parameter removing starting "//" into next URI-Path Option(s).

If ***To*** parameter contains SP-relative format, then the first URI-Path Option shall contain a letter "~" and map ***To*** parameter removing starting "/" into next URI-Path Option(s).

If ***To*** parameter contains CSE-relative format, then ***To*** parameter shall be mapped to URI-Path Option(s).

Table 6.2.2.3-1 shows valid mappings between the ***To*** request primitive parameter and the Uri-Path of the CoAP.

CSEBase represents the resource name of a <CSEBase> resource, CSEBase/ae12/cont27/contInst696 represents a structured CSE-relative resource ID, and cin00856 an unstructured CSE-relative resource ID.

Table 6.2.2.3-1: Mapping examples between To parameter and Uri-Path of the CoAP

Method		**Request Scope**		
		CSE-Relative	**SP-Relative**	**Absolute**
Structured	***To***	CSEBase/ae12/cont27/contInst696	/CSE178/CSEBase/ae12/cont27/contInst696	//mym2msp.org/CSE178/CSEBase/ae12/cont27/contInst696
	Uri-Path	CSEBase		_
			~	mym2msp.org
			CSE178	CSE178
			CSEBase	CSEBase
		ae12	ae12	ae12
		cont27	cont27	cont27
		contInst696	contInst696	contInst696
Unstructured	***To***	cin00856	/CSE178/cin00856	//mym2msp.org/CSE178/cin00856
	Uri-Path	cin00856		_
			~	mym2msp.org
			CSE178	CSE178
			cin00856	cin00856
NOTE: How to read this table: *To* primitive - from left to right, Uri-Path - from top to bottom.				

The *responseTypeValue* element of ***Response Type***, and the ***Result Persistence***, ***Delivery Aggregation***, ***Result Content***, parameters of ***Filter Criteria***, ***Desired Identifier Result Type***, ***Token Request Indicator***, ***Tokens***, ***Token IDs, Local Token IDs***, ***Role IDs, Authorization Relationship Indicator, Authorization Signature Indicator, Semantic Query Indicator*** and ***Operation*** parameters shall be carried, if needed in the Uri-Query Option in a short name form as specified in clause 8.2.2 of oneM2M TS-0004 [2].

6.2.2.4 Definition of New Options

6.2.2.4.0 Introduction

This clause describes new CoAP Options used for binding several oneM2M request/response parameters.
Table 6.2.2.4.0-1 contains definitions of the new CoAP Options and sub-clauses specify oneM2M parameter mapping with the newly defined CoAP Options in the table 6.2.2.4.0-1.

Table 6.2.2.4.0-1: Definition of New Options

No	C	U	N	R	Name	Format	Length	Default
279	X	X			oneM2M-FR	string	0-255	(None)
283	X	X			oneM2M-RQI	string	0-255	(None)
259	X	X			oneM2M-OT	string	15	(None)
291	X	X			oneM2M-RQET	string	15	(None)
295	X	X			oneM2M-RSET	string	15	(None)
299	X	X			oneM2M-OET	string	15	(None)
263	X	X			oneM2M-RTURI	string	0-255	(None)
303	X	X			oneM2M-EC	uint	1	(None)
307	X	X			oneM2M-RSC	uint	2	(None)
311	X	X			oneM2M-GID	string	0-255	(None)
267	X	X			oneM2M-TY	uint	2	(None)
319	X	X			oneM2M-CTO	uint	2	(None)
323	X	X			oneM2M-CTS	uint	2	(None)
327	X	X			oneM2M-ATI	string	0-255	(None)
271	X	X			oneM2M-RVI	string	1	(None)
331	X	X			oneM2M-VSI	string	0-255	(None)
335	X	X			oneM2M-GTM	string	0-512	(None)
339	X	X			oneM2M-AUS	string	0-255	(None)
275	X	X			oneM2M-ASRI	string	0-255	(None)
343	X	X			oneM2M-OMR	string	0-255	(None)
347	X	X			oneM2M-PRPI	string	0-255	(None)
351	X	X			oneM2M-MSU	string	0-255	(None)
NOTE 1: C, U, N, R means Critical, Unsafe, NoCacheKey and Repeatable respectively [1]. Table 6.2.2.4.0-1 follows the template used in clause 5.10 Option Definitions of CoAP specification [1].								
NOTE 2: CoAP Option numbers specified in table 6.2.2.4.0-1 are subject to change after review by IANA registration.								

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 Void

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.

6.2.2.4.12 Resource Type

The ***Resource Type*** parameter shall be mapped to the oneM2M-TY Option.

6.2.2.4.13 Content Offset

The ***Content Offset*** parameter shall be mapped to the oneM2M-CTO Option.

6.2.2.4.14 Content Status

The ***Content Status*** parameter shall be mapped to the oneM2M-CTS Option.

6.2.2.4.15 Assigned Token Identifiers

The ***Assigned Token Identifiers*** parameter shall be mapped to the oneM2M-ATI Option. The format of the oneM2M-ATI option shall be represented as a sequence of lti-value:tkid-value pairs separated by a colon ':' and multiple pairs separated by a '+' character.

EXAMPLE: The option looks as follows:

oneM2M-ATI: lti-value1:tkid-value1 + lti-value2:tkid-value2 + …

if the XML representation of the ***Assigned Token Identifiers*** parameter is given as (using short element names):

```
<ati>
    <ltia>
        <lti>lti-value1</lti>
        <tkid>tkid-value1</tkid>
    </ltia>
    <ltia>
       <lti>lti-value2</lti>
       <tkid>tkid-value2</tkid>
    </ltia>
    …
 </ati>
```

The data type m2m:dynAuthlocalTokenIdAssignments of the ***Assigned Token Identifiers*** parameter is defined in clause 6.3.5.43 of oneM2M TS-0004 [2].

6.2.2.4.16 Release Version Indicator

The ***Release Version Indicator*** parameter shall be mapped to the oneM2M-RVI Option.

6.2.2.4.17 Vendor Information

The ***Vendor Information*** parameter shall be mapped to the oneM2M-VSI Option.

6.2.2.4.18 Group Request Target Members

The ***Group Request Target Members*** parameter shall be mapped to the oneM2M-GTM Option.

6.2.2.4.19 Authorization Signatures

The ***Authorization Signatures*** parameter shall be mapped to the oneM2M-AUS Option.

6.2.2.4.20 Authorization Signature Request Information

The ***Authorization Signature Request Information*** parameter shall be mapped to the oneM2M-ASRI Option.

6.2.2.4.21 Ontology Mapping Resources

The ***Ontology Mapping Resources*** parameter shall be mapped to the oneM2M-OMR Option. The format of the oneM2M-OMR option shall be represented as a sequence of oneM2M resource identifiers separated by a '+'.

EXAMPLE: The option looks as follows:

oneM2M-OMR: /IN-CSE-0001/omr1+/IN-CSE-0001/omr2+…

6.2.2.4.22 Primitive Profile Identifier

The ***Primitive Profile Identifier*** parameter shall be mapped to the oneM2M-PRPI Option.

6.2.2.4.22 M2M Service User

The ***M2M Service User*** parameter shall be mapped to the oneM2M-MSU 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.

A ***Token Request Information*** parameter included in a response primitive shall be mapped into the payload. The Content-Format shall be set compliant with the data representation.

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 a single CoAP Response Code, the ***Response Status Code*** parameter shall be specified in oneM2M-RSC Option.

Table 6.2.4-1: Mapping between oneM2M Response Status Code and CoAP Response Code

oneM2M Response Status Code	Description	CoAP Response Code	Description
1000	ACCEPTED	None	Not used
1001	ACCEPTED for nonBlockingRequestSynch	2.01	Created (indicates that a <request> resource has been created
1002	ACCEPTED for nonBlockingRequestAsynch	2.01 or 2.04	2.01 (Created) is used if a <request> resource was created, otherwise 2.04 (Changed) is used
2000 (for RETRIEVE operation)	OK	2.05	Content
2000 (for NOTIFY operation)	OK	2.04	Changed. CoAP payload shall be empty.
2001	CREATED	2.01	Created

oneM2M Response Status Code	Description	CoAP Response Code	Description
2002	DELETED	2.02	Deleted
2004	UPDATED	2.04	Changed
4000	BAD_REQUEST	4.00	Bad Request
4001	RELEASE_VERSION_NOT_SUPPORTED	5.01	Not Implemented
4004	NOT_FOUND	4.04	Not Found
4005	OPERATION_NOT_ALLOWED	4.05	Method Not Allowed
4008	REQUEST_TIMEOUT	5.04	Gateway Timeout
4015	UNSUPPORTED_MEDIA_TYPE	4.15	Unsupported Content-Format
4101	SUBSCRIPTION_CREATOR_HAS_NO_PRIVILEGE	4.03	Forbidden
4102	CONTENTS_UNACCEPTABLE	4.00	Bad Request
4103	ORIGINATOR_HAS_NO_PRIVILEGE	4.03	Forbidden
4104	GROUP_REQUEST_IDENTIFIER_EXISTS	4.00	Bad Request
4105	CONFLICT	4.03	Forbidden
4106	ORIGINATOR_HAS_NOT_REGISTERED	4.03	Forbidden
4107	SECURITY_ASSOCIATION_REQUIRED	4.03	Forbidden
4108	INVALID_CHILD_RESOURCE_TYPE	4.03	Forbidden
4109	NO_MEMBERS	4.03	Forbidden
4110	GROUP_MEMBER_TYPE_INCONSISTENT	4.00	Bad Request
4111	ESPRIM_UNSUPPORTED_OPTION	4.03	Forbidden

oneM2M Response Status Code	Description	CoAP Response Code	Description
4112	ESPRIM_UNKNOWN_KEY_ID	4.03	Forbidden
4113	ESPRIM_UNKNOWN_ORIG_RAND_ID	4.03	Forbidden
4114	ESPRIM_UNKNOWN_RECV_RAND_ID	4.03	Forbidden
4115	ESPRIM_BAD_MAC	4.03	Forbidden
4116	ESPRIM_IMPERSONATION_ERROR	4.03	Forbidden
4117	ORIGINATOR_HAS_ALREADY_REGISTERED	4.03	Forbidden
4118	ONTOLOGY_NOT_AVAILABLE	4.04	Not Found
4119	LINKED_SEMANTICS_NOT_AVAILABLE	4.04	Not Found
4120	INVALID_SEMANTICS	4.02	Bad Option
4121	MASHUP_MEMBER_NOT_FOUND	4.04	Not Found
4122	INVALID_TRIGGER_PURPOSE	4.02	Bad Option
4123	ILLEGAL_TRANSACTION_STATE_TRANSITION_ATTEMPTED	4.00	Bad Request
4124	BLOCKING_SUBSCRIPTION_ALREADY_EXISTS	4.00	Bad Request
4125	SPECIALIZATION_SCHEMA_NOT_FOUND	5.01	Not Implemented
4126	APP_RULE_VALIDATION_FAILED	4.03	Forbidden
4127	OPERATION_DENIED_BY_REMOTE_ENTITY	4.03	Forbidden
4128	SERVICE_SUBSCRIPTION_NOT_ESTABLISHED	4.03	Forbidden
4130	ONTOLOGY_MAPPING_ALGORITHM_NOT_AVAILABLE	4.04	Not Found
4131	ONTOLOGY_MAPPING_POLICY_NOT_MATCHED	4.00	Bad Request
4132	ONTOLOGY_MAPPING_NOT_AVAILABLE	4.04	Not Found
4133	BAD_FACT_INPUTS_FOR_REASONING	4.00	Bad Request
4134	BAD_RULE_INPUTS_FOR_REASONING	4.00	Bad Request
4135	DISCOVERY_LIMIT_EXCEEDED	4.03	Forbidden
4136	PRIMITIVE_PROFILE_NOT_ACCESSIBLE	4.03	Forbidden
4137	PRIMITIVE_PROFILE_BAD_REQUEST	4.00	Bad Request
4138	UNAUTHORIZED_USER	4.03	Forbidden
4139	SERVICE_SUBSCRIPTION_NOT_ACTIVE	4.03	Forbidden
4140	SOFTWARE_CAMPAIGN_CONFLICT	4.03	Forbidden
4143	INVALID_SPARQL_QUERY	4.00	Bad Request
5000	INTERNAL_SERVER_ERROR	5.00	Internal Server Error
5001	NOT_IMPLEMENTED	5.01	Not Implemented
5103	TARGET_NOT_REACHABLE	4.04	Not Found
5105	RECEIVER_HAS_NO_PRIVILEGE	4.03	Forbidden
5106	ALREADY_EXISTS	4.00	Bad Request
5107	REMOTE_ENTITY_NOT_REACHABLE	4.04	Not Found
5203	TARGET_NOT_ SUBSCRIBABLE	4.03	Forbidden
5204	SUBSCRIPTION_VERIFICATION_INITIATION_FAILED	5.00	Internal Server Error
5205	SUBSCRIPTION_HOST_HAS_NO_PRIVILEGE	4.03	Forbidden
5206	NON_BLOCKING_SYNCH_REQUEST_NOT_SUPPORTED	5.01	Not Implemented
5207	NOT_ACCEPTABLE	4.06	Not Acceptable
5208	DISCOVERY_DENIED_BY_IPE	4.03	Forbidden
5209	GROUP_MEMBERS_NOT_RESPONDED	5.00	Internal Server Error
5210	ESPRIM_DECRYPTION_ERROR	5.00	Internal Server Error
5211	ESPRIM_ENCRYPTION_ERROR	5.00	Internal Server Error
5212	SPARQL_UPDATE_ERROR	5.00	Internal Server Error
5214	TARGET_HAS_NO_SESSION_CAPABILITY	4.03	Forbidden
5215	SESSION_IS_ONLINE	4.03	Forbidden
5216	JOIN_MULTICAST_GROUP_FAILED	5.00	Internal Server Error
5217	LEAVE_MULTICAST_GROUP_FAILED	5.00	Internal Server Error
5218	TRIGGERING_DISABLED_FOR_RECIPIENT	5.03	Service Unavailable
5219	UNABLE_TO_REPLACE_ REQUEST	5.03	Service Unavailable
5220	UNABLE_TO_RECALL_REQUEST	5.03	Service Unavailable
5221	CROSS_RESOURCE_OPERATION_FAILURE	5.00	Internal Server Error
5222	TRANSACTION_PROCESSING_IS_INCOMPLETE	4.03	Forbidden
5230	ONTOLOGY_MAPPING_ALGORITHM_FAILED	5.00	Internal Server Error
5231	ONTOLOGY_CONVERSION_FAILED	5.00	Internal Server Error
5232	REASONING_PROCESSING_FAILED	5.00	Internal Server Error
6003	EXTERNAL_OBJECT_NOT_REACHABLE	4.04	Not Found
6005	EXTERNAL_OBJECT_NOT_FOUND	4.04	Not Found

oneM2M Response Status Code	Description	CoAP Response Code	Description
6010	MAX_NUMBER_OF_MEMBER_EXCEEDED	4.00	Bad Request
6020	MGMT_SESSION_CANNOT_BE_ESTABLISHED	5.00	Internal Server Error
6021	MGMT_SESSION_ESTABLISHMENT_TIMEOUT	5.00	Internal Server Error
6022	INVALID_CMDTYPE	4.00	Bad Request
6023	INVALID_ARGUMENTS	4.00	Bad Request
6024	INSUFFICIENT_ARGUMENTS	4.00	Bad Request
6025	MGMT_CONVERSION_ERROR	5.00	Internal Server Error
6026	MGMT_CANCELLATION_FAILED	5.00	Internal Server Error
6028	ALREADY_COMPLETE	4.00	Bad Request
6029	MGMT_COMMAND_NOT_CANCELLABLE	4.00	Bad Request
6030	EXTERNAL_OBJECT_NOT_REACHABLE_BEFORE_RQET_TIMEOUT	5.04	Gateway Timeout
6031	EXTERNAL_OBJECT_NOT_REACHABLE_BEFORE_OET_TIMEOUT	5.04	Gateway Timeout
6033	NETWORK_QOS_CONFIG_ERROR	5.03	Internal Server Error
6034	REQUESTED_ACTIVITY_PATTERN_NOT_PERMITTED	4.03	Forbidden

The Receiver shall use this table to determine the CoAP response code that is to be used in the response, based on the value of the oneM2M ***Response Status Code*** parameter.

6.3 Accessing Resources in CSEs

6.3.0 Introduction

This clause describes the behaviour of the CoAP layer depending on the ***Response Type*** parameter. Note that the CoAP messaging model defined in [1] applies to all message exchanges.

oneM2M Requests should be sent as CoAP Confirmable (CON) messages, although an Originator can send a request as a Non-confirmable (NON) message if there is a good reason for doing this. An Originator that relies on getting a response to its request should use a Confirmable rather than a Non-confirmable message for its request.

oneM2M Responses should be sent as CoAP CON messages, although there is one case where a NON message should be used. This is indicated in clause 6.3.1.

If the Originator or Receiver sends a CON message it shall retransmit that message if it does not receive a CoAP acknowledgement message, as required by [1]. The recipient (Receiver or Originator) shall take care to de-duplicate CON messages as described in [1].

The recipient of a CoAP message shall process the oneM2M request or response it contains, even if it was sent as Non-confirmable.

6.3.1 Blocking case

1) If the ***Response Type*** parameter is configured as "blockingRequest" (blocking case), the Originator (CoAP client) shall send the oneM2M request to the Receiver (CoAP server). The oneM2M ***Operation*** parameter shall be mapped to a CoAP Method according to Table 6.2.1-1.

2) After processing the oneM2M request, the Receiver shall send the oneM2M response in a CoAP response with a CoAP response code as given by Table 6.2.4-1.

    - If the oneM2M request was sent as a CoAP Confirmable message, the Receiver may either piggyback this response to the request on the CoAP ACK message, or send the response as a separate CoAP Confirmable message after it has sent the CoAP ACK.

- If the oneM2M request was sent as a Non-confirmable message, the oneM2M response shall be returned as a separate CoAP message. This response should be sent as a Non-confirmable CoAP message but it may be sent as Confirmable (this means that a receiver can, if it so chooses, send all Responses as Confirmable regardless of how the Request was sent).

3) The Originator's CoAP binding may generate a response primitive containing a oneM2M ***Response Status Code*** of "REQUEST_TIMEOUT" if it considers that it has taken too long for the CoAP response to come back from the Receiver. It shall ignore any response to the original request that it might receive after it has done this.

6.3.2 Non-Blocking Asynchronous case

1) If the ***Response Type*** parameter is configured as "nonBlockingRequestAsynch" (non-blocking asynchronous case), the Originator (CoAP client) should send the oneM2M request to the Receiver (CoAP server) as a CoAP Confirmable message. This request shall be sent using the CoAP POST method, and shall include the ***Operation*** parameter, mapped as described in clause 6.2.2.3.

2) The Receiver, after validating the request and before processing it fully, shall return a oneM2M response to the originator. It may either piggyback (2a) this response on the CoAP ACK message (if the request was sent as a Confirmable message) or send the response as a separate CoAP response message after it has sent the CoAP ACK (2b). In this latter case it shall send the response as a Confirmable message.

   - If the Receiver supports the <request> resource type, it shall respond with a 2.01 (Created) CoAP response code and a oneM2M ***Response Status Code*** of "ACCEPTED for nonBlockingRequestAsynch". The response shall include the URI of the new <request> resource in a sequence of one or more Location-Path and/or Location-Query Options.

   - If the Receiver does not support the <request> resource type, it shall respond with a 2.04 (Changed) CoAP response code and a oneM2M ***Response Status Code*** of "ACCEPTED for nonBlockingRequestAsynch".

3) The Receiver, upon successful processing of the request, shall send a new CoAP Confirmable request message using the CoAP POST method. This message contains a oneM2M NOTIFY primitive whose content contains the response to the original request.

4) The Originator may either piggyback a response to this request (4a) or send it as a separate CoAP response after the acknowledgment message (4b). This response shall contain the appropriate CoAP response code as defined in table 6.2.4-1 and have an empty payload.

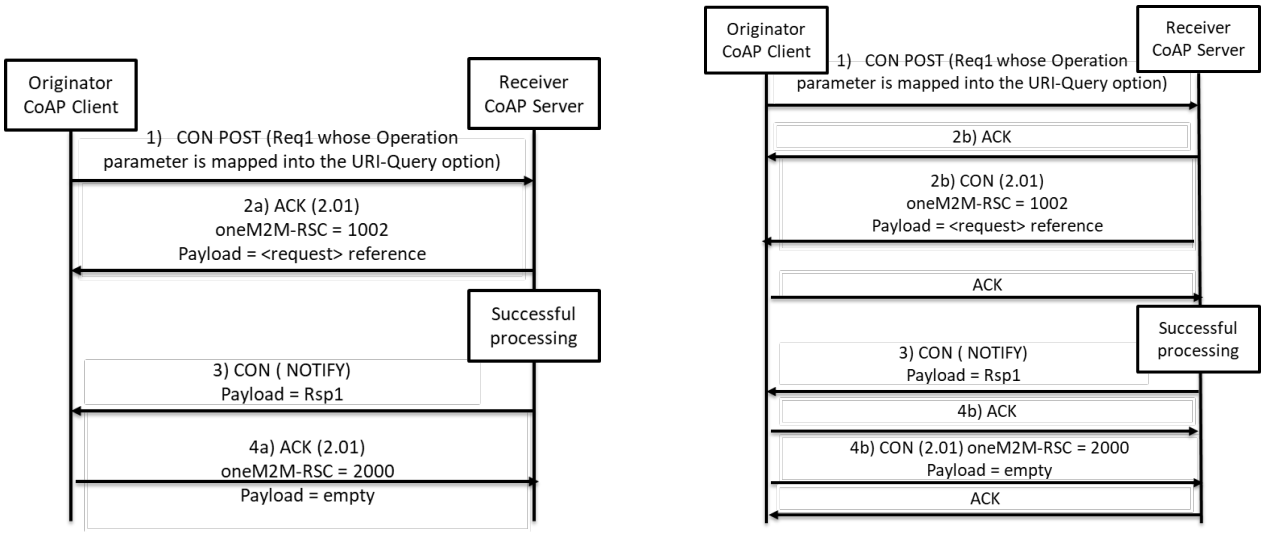

Figure 6.3.2-1: Non-Blocking Asynchronous Case

6.3.3 Non-Blocking Synchronous case

1) If the ***Response Type*** parameter is configured as "nonBlockingRequestSynch" (non-blocking synchronous case), the Originator (CoAP client) should send the oneM2M request to the Receiver (CoAP server) as a CoAP Confirmable message. This request shall be sent using the CoAP POST method, and shall include the ***Operation*** parameter, mapped as described in clause 6.2.2.3.

2) The Receiver, after validating the request and before processing it fully, shall return a oneM2M response to the originator. It may either piggyback this response (2a) on the CoAP ACK message (if the request was sent as a CON message) or send the response as a separate CoAP message after it has sent the CoAP ACK (2b). In this latter case it shall send the response as a Confirmable message.

   - If the Receiver supports the <request> resource type, it shall respond with a 2.01 (Created) CoAP response code and a oneM2M ***Response Status Code*** of "ACCEPTED for nonBlockingRequestSynch". The response shall include the URI of the new <request> resource in a sequence of one or more Location-Path and/or Location-Query Options.

   - If the Receiver does not support the <request> resource type, it shall respond with a 5.01 (Not implemented) CoAP response code and a oneM2M ***Response Status Code*** of "NON_BLOCKING_SYNCH_REQUEST_NOT_SUPPORTED".

3) The Originator can use the <request> resource reference to synchronously retrieve the <request> resource that contains the response to the original request.

4) The Receiver, upon receipt of this retrieve request, shall handle it as in clause 6.3.1 since it is a non-blocking request.

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

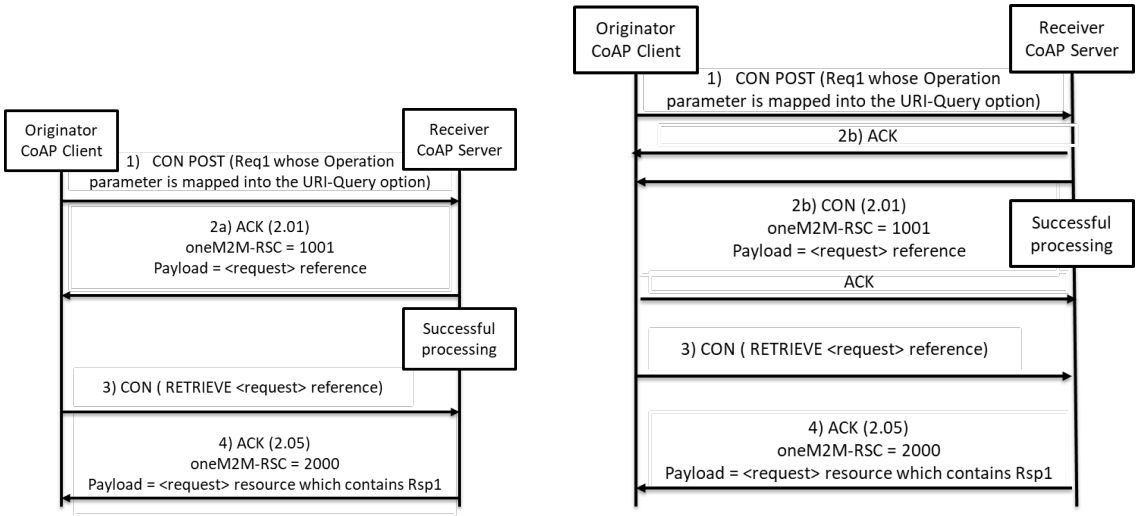

Figure 6.3.3-1: Non-Blocking Synchronous Case

6.3.4 Flex Blocking case

1) If the ***Response Type*** parameter is configured as "flex blocking", the Originator (CoAP client) should send the oneM2M request to the Receiver (CoAP server) as a CoAP Confirmable message. This request shall be sent using the CoAP POST method, and shall include the ***Operation*** parameter, mapped as described in clause 6.2.2.3.

2) The Receiver shall determine whether to handle the request using "nonBlockingRequestSynch" or "nonBlockingRequestAsynch" mode:

- If the Receiver chooses "nonBlockingRequestAsynch" processing proceeds as described in clause 6.3.2, starting from step 2).
- If the Receiver chooses "nonBlockingRequestSynch" processing proceeds as described in clause 6.3.3, starting from step 2).

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 to IETF RFC 7959 [3] for further details.

7 Security Consideration

This clause applies to CoAP unicast communication only. Security for multicast communication is addressed in clause B.1.

CoAP itself does not provide protocol primitives for authentication or authorization.

Just as HTTP is secured using Transport Layer Security (TLS) over TCP, CoAP can 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].

Annex A (informative): Example Procedures

A.1 Blocking case of AE Registration

Figure A.1-1 illustrates CoAP mapping of AE Registration procedure described in clauses 7.2.2.1, 7.4.6.2.2 and E.1 of oneM2M TS-0004 [2] and shows an example of blocking case which is described in clause 6.3.1.

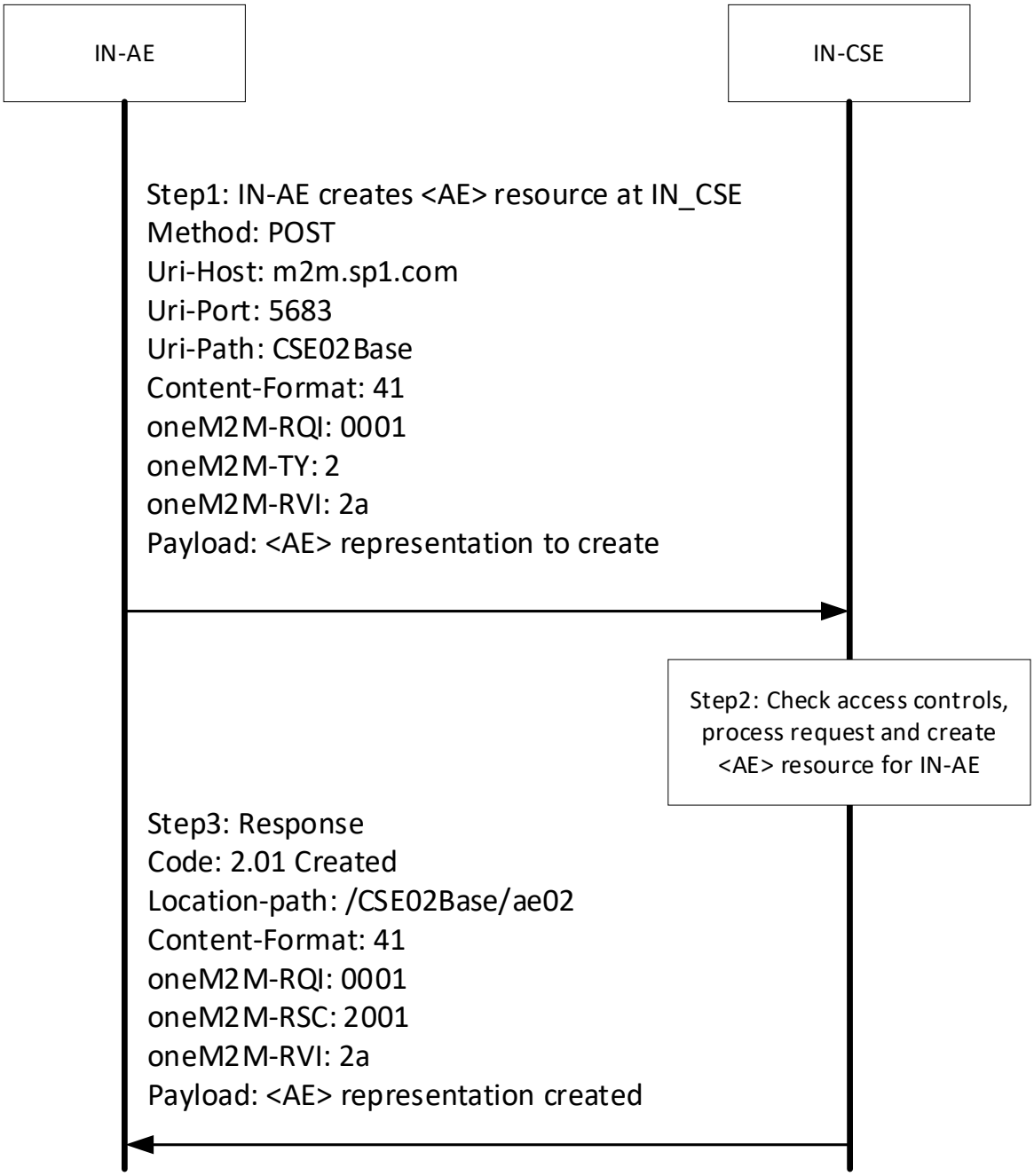

Figure A.1-1: Binding Example - Blocking case of AE Registration

A.2 Non-blocking synchronous case of AE Registration

Figure A.2-1 illustrates CoAP mapping of AE Registration procedure described in clauses 7.2.2.1, 7.4.6.2.2 and E.2 of oneM2M TS-0004 [2] and shows an example of non-blocking synchronous case which is described in clause 6.3.3.

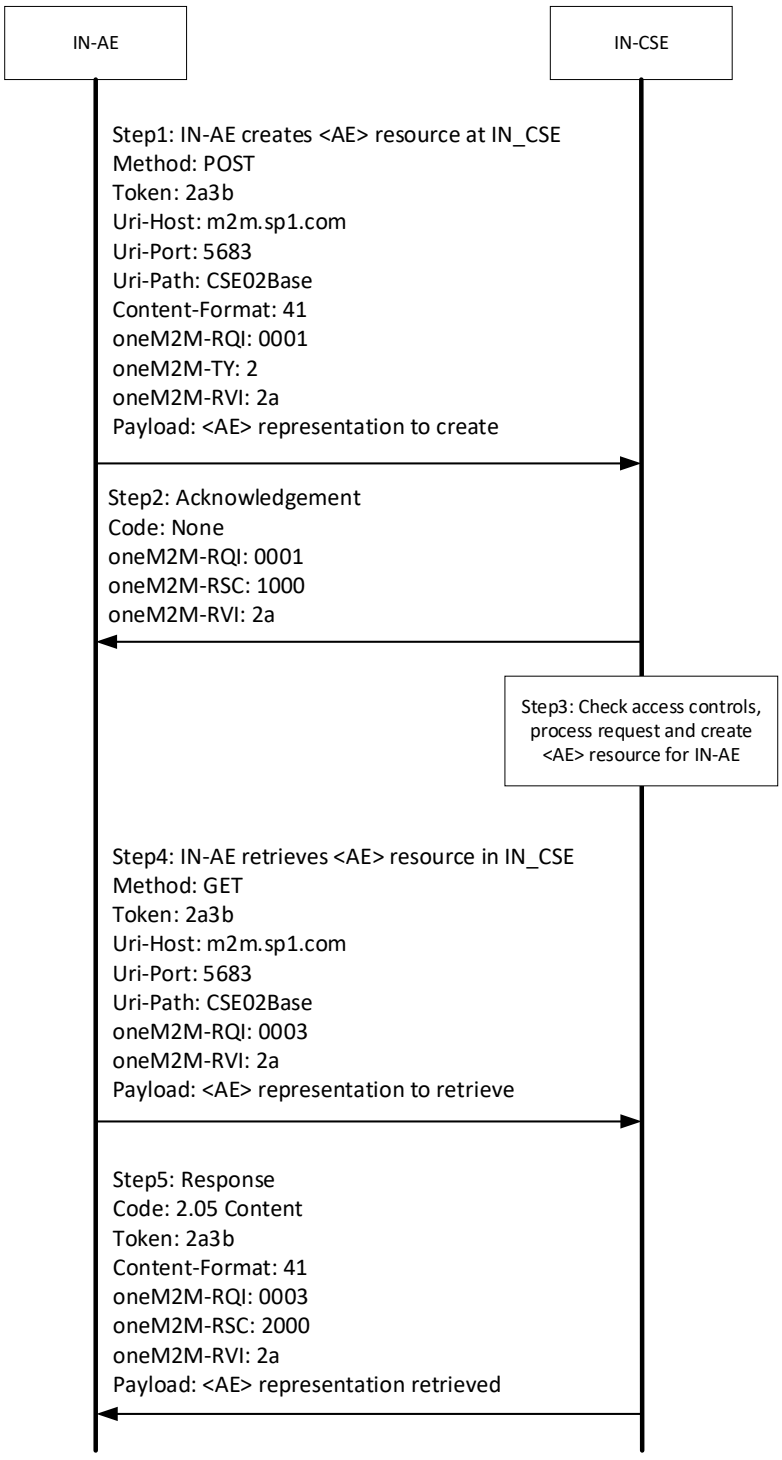

Figure A.2-1: Binding Example - Non-blocking synchronous case of AE Registration

Annex B (normative):
Multicast group fan out procedure

B.0 Introduction

This clause describes the behaviour of CoAP layer for multicast group fan out procedure. Figure B.0-1 illustrates the steps involved in the interaction.

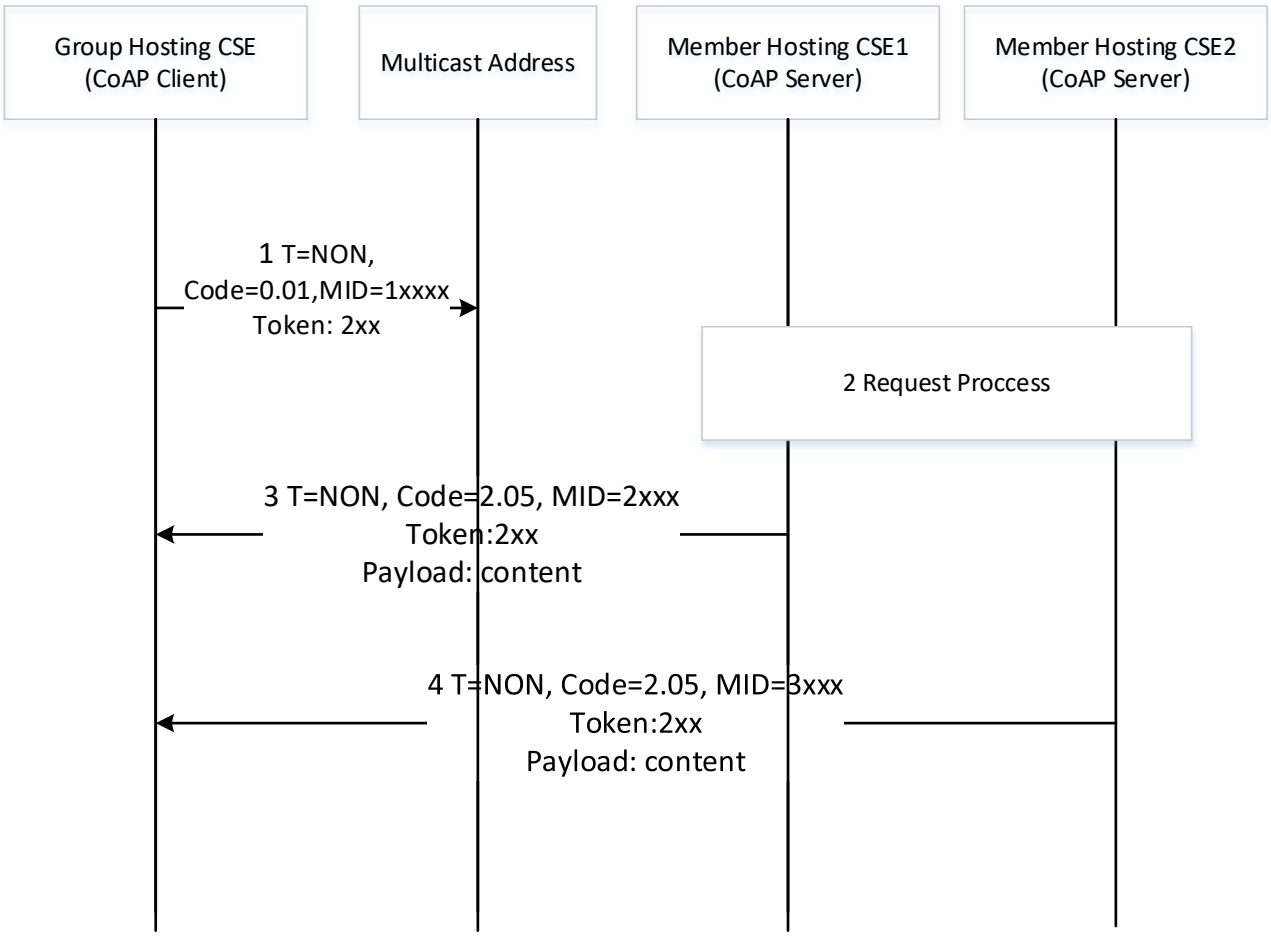

Figure B.0-1: Multicast group fan out procedure

- The Group Hosting CSE (CoAP client) shall use the Non-confirmable Method for the resource to the Member Hosting CSEs (CoAP server). The Group Hosting CSE shall provide a unique Token value in the request.
- The Member Hosting CSE, upon successful processing of the request, shall send an appropriate response in a separate Non-confirmable message with the same Token value.

B.1 Security

DTLS is not applicable to multicast group fan out messages. Security for multicast group fan out is addressed in clause 6.1.2.2.3 of oneM2M TS-0003 [4].

B.2 Caching

A GET request to a multicast group fan out shall not contain an ETag option.

History

Publication history		

Draft history		
V4.0.0	15 Jan 2020	Initial version using V3.5.0 as a baseline and incorporating CR: SDS-2019-586R02
V4.1.0	23 Feb 2021	Incorporated contribution SDS-2020-0100R04
V4.2.0	05 Oct 2021	Incorporate contribution SDS-2021-0194
V4.3.0	24 Oct 2021	Incorporate contributions from SDS 52 – SDS 56: SDS-2021-0268-TS-0008_SPARQL_CoAP_Status_Codes_R4 SDS-2020-0374R05-Allow_non-confirmable_messages_in_CoAP SDS-2020-0065R04-Changes_to_CoAP_options_for_IANA_registration The following issue is fixed: #24 Wrong spelling of "Authorization Signatures" request parameter