|  |
| --- |
|  |

|  |  |
| --- | --- |
| CHANGE REQUEST | |
| Meeting ID:\* | SDS #48 |
| Source:\* | Peter Niblett, IBM |
| Date:\* | 2020-12-14 |
| Reason for Change/s:\* | Clarify expected behaviour if non-confirmable messages are used (R4) |
| CR against: Release\* | Release 4 |
| CR against: WI\* | Active WI-xxxx  MNT maintenance / < Work Item number(optional)>  Is this a mirror CR? Yes  No  mirror CR number: (Note to Rapporteur - use latest agreed revision)  STE Small Technical Enhancements / < Work Item number (optional)>  Only ONE of the above shall be ticked |
| CR against: TS/TR\* | TS-0009 v.4.0.0 |
| Clauses \* | Modified clauses: 6.3.0, 6.3.1, 6.3.2, 6.3.3, 6.3.4 |
| Type of change: \* | Editorial change  Bug Fix or Correction  Change to existing feature or functionality  New feature or functionality  Only ONE of the above shall be ticked |
| Impacted other TS/TR(s) |  |
| Post Freeze checking:\* | This CR contains only essential changes and corrections? YES  NO  This CR may break backwards compatibility with the last approved version of the TS? YES  NO |
| Template Version: January 2017 (Do not modify) | |

**oneM2M Notice**

The document to which this cover statement is attached is submitted to oneM2M. Participation in, or attendance at, any activity of oneM2M, constitutes acceptance of and agreement to be bound by terms of the Working Procedures and the Partnership Agreement, including the Intellectual Property Rights (IPR) Principles Governing oneM2M Work found in Annex 1 of the Partnership Agreement.

GUIDELINES for Change Requests:

Provide an informative introduction containing the problem(s) being solved, and a summary list of proposals.

Each CR should contain changes related to only one particular issue/problem.

In case of a correction, and the change apply to previous releases, a separate “mirror CR” should be posted at the same time of this CR

Mirror CR: applies only when the text, including clause numbering are exactly the same.

Companion CR: applies when the change means the same but the baselines differ in some way (e.g. clause number).

Follow the principle of completeness, where all changes related to the issue or problem within a deliverable are simultaneously proposed to be made E.g. A change impacting 5 tables should not only include a proposal to change only 3 tables. Includes any changes to references, definitions, and acronyms in the same deliverable.

Follow the drafting rules.

All pictures must be editable.

Check spelling and grammar to the extent practicable.

Use Change bars for modifications.

The change should include the current and surrounding clauses to clearly show where a change is located and to provide technical context of the proposed change. Additions of complete clauses need not show surrounding clauses as long as the proposed clause number clearly shows where the new clause is proposed to be located.

Multiple changes in a single CR shall be clearly separated by horizontal lines with embedded text such as, start of change 1, end of change 1, start of new clause, end of new clause.

When subsequent changes are made to content of a CR, then the accepted version should not show changes over changes. The accepted version of the CR should only show changes relative to the baseline approved text.

Introduction

TS-0008 clauses 6.3.1, 6.3.2, 6.3.3 and 6.3.4 describe how the oneM2M request/response patterns map to the CoAP messaging model. All four of the say that requests shall be sent using Confirmable messages and the flows described assume this.

They do not say what happens if an Originator chooses to send a Request using a Non-confirmable message. They are also not 100% clear whether the receiver is required to use Confirmable messages when responding. In some cases the text does say this (or the diagrams say CON) but this requirement is missing in some places, for example 6.3.1 (blocking case) doesn't say.

This CR clarifies these points:

Originators should use Confirmable messages when sending requests, but use non-confirmable if there’s a good reason for doing this (e.g. they aren’t interested in the reply or whether the request actually happened)

The consequence of this is that a CoAP receiver should accept incoming non-confirmable messages (if it gets them)

In Blocking Mode, if a request is sent as Non-Confirmable then the response is sent as Non-Confirmable

In Non-Blocking Asynch, if a request is sent as Non-Confirmable the acknowledgement of that request is sent as Non-confirmable but the actual response notification is sent as Confirmable

In Non-Blocking Synch the immediate response is sent as Confirmable, but the originator could choose to use a Non-confirmable request to retrieve that actual response (since this is a blocking retrieve).

[Also “Confirmable Method” has been changed to “Confirmable message”]

### 

### \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Start of change 1 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### 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.

Requests should be sent as Confirmable messages, although an Originator can send them as Non-confirmable if there is a good reason for doing this (an Originator should not use Non-confirmable if it relies on getting a response to its requests). The recipient of a CoAP message shall accept that message even if it is Non-confirmable.

Responses should be sent as Confirmable messages, although cases where Non-confirmable may be used are indicated in clauses 6.3.1 to 6.3.4.

If the Originator sends a request as Confirmable it shall resend that request until it has been acknowledged, and the Receiver shall resend confirmable responses until they have been acknowledged. The recipient (Receiver or Originator) shall take care to de-duplicate confirmable messages as described in [1].

### 6.3.1 Blocking case

1. If ***Response Type*** parameter is configured as "blockingRequest" (blocking case), the Originator (CoAP client) shall send the 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 request, the Receiver shall send a CoAP response with a CoAP response code as given by Table 6.2.4-1. If the request was sent as a 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 the CoAP ACK. If the request was sent as Non-confirmable, the response is returned as a separate Non-confirmable CoAP message.
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 request to the Receiver (CoAP server) as a Confirmable message. This request shall be sent using a 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 CoAP response to the originator. If the request was sent as a Confirmable message, the Receiver may either piggyback (2a) this response to the request on the CoAP ACK message, or send the response as a separate CoAP Confirmable message after the CoAP ACK (2b). If the request was sent as Non-confirmable, the response is returned as a separate Non-confirmable CoAP 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 POST method (NOTIFY primitive) and whose payload 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 request to the Receiver (CoAP server) as a Confirmable message. This request shall be sent using a 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 CoAP response to the originator. It may either piggyback this response (2a) on the CoAP ACK message (if the request was sent as a Confirmable message) or send the response as a separate CoAP Confirmable message after the CoAP ACK (2b).

* 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\_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 request to the Receiver (CoAP server) as a Confirmable message. This request shall be sent using a 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 6.2.2, starting from step 2).
* If the Receiver chooses "nonBlockingRequestSynch" processing proceeds as described in clause 6.2.3, starting from step 2).

### \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* End of Change 1 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*