|  |  |
| --- | --- |
| CHANGE REQUEST | |
| Meeting ID:\* | SDS#40 |
| Source:\* | Hyundai Mortors and KETI |
| Date:\* | 2019-05-20 |
| Contact:\* | JaeSeung Song, KETI, jssong@sejong.ac.kr  Youngjin Na, Hyundai Motors, [yjra@hyundai.com](mailto:yjra@hyundai.com)  Minbyeong Lee, Hyundai Motors, [minbyeong.lee@hyundai.com](mailto:minbyeong.lee@hyundai.com) |
| Reason for Change/s:\* | Suggest a solution for oneM2M message delivery repetition. |
| CR against: Release\* | R4 |
| CR against: WI\* | Active < Work Item number(optional)>  MNT maintenance / < Work Item number(optional)>  Is this a mirror CR? Yes  No  STE Small Technical Enhancements / < Work Item number(optional)>  Only ONE of the above shall be ticked |
| CR against: TS/TR\* | TR-0053 V0.2.1 |
| Clauses \* | Section 6.1.2 (Limitation of oneM2M messages), 6.1.3 (Potential Requirements), 6.1.4 (Potential Solutions) |
| 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

This contribution proposes a potential solution to the oneM2M system to efficiently support message repetition.

The general concept of message repetition is to allow the delivery of the same message multiple times to IoT devices and applications as a common service via oneM2M IoT platform.

The proposing solution suggest to enhance followings:

* Add an optional parameter called “Message repetition” to oneM2M Request
* Use the <schedule> resource to implement message repetition
* Consider the type of a target resource and use proper mechanisms, e.g., update <subscription> resource if the target resource has subscribers

## ------------------------------Start of change 1-------------------------------------

6.1.4 Potential Solutions

*Editor’s Note: The section describes potential solutions related to optimizing/enhancing the oneM2M function to address the identified limitations and requirements.*

* + - 1. Potential solution for message repetition

***Update to Request***

In TS-0001 Section 8.1.2 Request, various parameters are defined for oneM2M Request message over the Mca and Mcc reference points.

A message repetition parameter can be introduced as an optional parameter as follows:

**Optional Parameters:**

* ***Message repetition:*** optional message repetition delivery identifier: Indicates the message needs to be delivered at a given number of times repeatedly within a specified time duration

Example usage of message repetition: Originator wants to send a CREATE request message that creates a new <*contentInstance*> resource with the ‘0’ value but delivered this information to the subscribers of this resource 5 times within 60 seconds in the format of 5:60. As a service application using this value interprets the value ‘0’ as a warning message for high temperature, a corresponding message, e.g., “CAUTION: THE TEMPERATURE IS HIGH”, can be notified a given number of times to the subscribed receiver(s) of the parent container.

Once a request message with the message repetition parameter is received, the Receiver (i.e., CSE) can do the following:

* **Use additional parameters in the <*subscription*> resource**: In this case, the message is used to update the target resource with proper configuration such as allowing multiple notifications via additional attribute to the <*subscription*> resource. Because the current <*subscription*> resource does not provide any means to support sending a notification repeatedly, additional attributes should be defined.

For example, if there exists the <*subscription*> resource related to the target resource of the received message, the <*subscription*> resource related to the target resource can be configured with the following additional attributes:

* *notiRepetition* : if this value is 0 or 1, then the notification is executed as a normal way. But if this value is greater than 1, notification should be sent the given number of times. For example, if the value is 3, notification should be sent three times.
* *notiInterval* : specifies an interval for repeated notifications (e.g., 4 times during 60 seconds)

Editor’s note: It if FFS how the message repetition mechanism affects to other CRUDN operations.

The following figure shows a possible procedure for performing the message delivery repetition.



Figure 6.1.4.1-1. Call flow for message repetition

* **Step 0**: In this call flow, it is assumed that ADN-AE creates <*container*> and <*contentInstance*> resources for “Warning Alarm” in IN-CSE. In addition, IN-AE (e.g., a smartphone application displaying alarm message based on the value of ADN-AE’s <*contentInstance*> resource) subscribes to the <*container*> resource for “Warning Alarm” with two additional attributes for performing message repetition.
* **Step 1**: When there is an accident, ADN-AE creates a new <*contentInstance*> representing a warning alarm under the target <*container*> resource for “Warning Alarm”. This request is sent from ADN-AE to IN-CSE with message repetition information (for example, send notification three times in a minute).
* **Step 2**: IN-CSE creates a new <*contentInstance*> resource and configures the <*subscription*> resource under the <*container*> for “Warning Alarm” based on the request message. This means that *notiRepetition* and *notiInterval* attributes of <*subscription*> will be configured accordingly to ‘1’ and ‘3:60’.

Editor’s note: It needs to be specified how <*subscription*> can be updated with information related to the message repetition.

* **Step 3**: Based on the updated configuration and the newly created <*contentInstance*>, the first notification to IN-AE is sent out from IN-CSE to notify that a new <*contentInstance*> is created.
* **Step 4**: After 20 seconds later, IN-CSE send the second notification to IN-AE based on the message repetition policy written in the <*subscription*> resource under the <*container*> resource of “Warning Alarm”.
* **Step 5**: After 20 seconds later from the second notification, IN-CSE send the third notification to IN-AE based on the message repetition policy written in the <*subscription*> resource under the <*container*> resource of “Warning Alarm”.