|  |
| --- |
|  |

|  |  |
| --- | --- |
| CHANGE REQUEST | |
| Meeting ID:\* | SDS 52.3 |
| Source:\* | Miguel Angel Reina Ortega, ETSI, [MiguelAngel.ReinaOrtega@etsi.org](mailto:MiguelAngel.ReinaOrtega@etsi.org) |
| Date:\* | 2022-01-25 |
| Reason for Change/s:\* | operationResult handling |
| CR against: Release\* | Rel-3 |
| CR against: WI\* | Active < WI-0077>  MNT maintenance / < Work Item number(optional)>  Is this a mirror CR? Yes  No  mirror CR number:  STE Small Technical Enhancements / < Work Item number (optional)>  Only ONE of the above shall be ticked |
| CR against: TS/TR\* | TS-0004 v3.24.0 |
| Clauses \* | 7.2.2.1, 7.2.2.2, 7.4.14.2.5 |
| 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 |
| Other TS/TR(s) impacted | None |
| 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 2019 (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 CR proposes a change for handling of operationResult attribute in <request> resource for nonBlockingSync operations. Change 1 is about the procedure as several mistakes were detected, change 2 as described in issue <https://git.onem2m.org/issues/issues/-/issues/45> and change 3 is an attribute name typo.

## ----------------------- Start of Change 1--------------------------------------------

#### 7.2.2.1 Generic resource request procedure for originator

A generic resource Request procedure shall be comprised of the following actions. Additional actions specific to individual procedures are listed in the respective clauses by referencing these actions and providing additional steps. The Originator shall execute the following steps in order.



Figure 7.2.2.1‑1: Generic procedure of Originator

Orig-1.0 "Compose Request primitive": Refer to clause 7.3.1.1 for details.

Orig-2.0 "Send a Request primitive to the Receiver CSE": The Request primitive shall include the mandatory parameters ***Operation***, ***To***, ***From*** and ***Request*** ***Identifier***. Refer to clause 7.3.1.2 for details.

Orig-3.0 "Check Response Type": In this step, the Originator checks that the communication method is either blockingRequest, nonBlockingRequestSynch, nonBlockingRequestAsynch or flexBlocking by using the ***Response*** ***Type*** parameter (see detail in clause 8.1.2 in the oneM2M TS-0001 [6]). If the ***Response Type*** parameter does not exist, the communication method is "blockingRequest" as specified at clause 6.4.1.

If the Response Type is blockingRequest the Originator waits for the Response primitive and goes to step Orig-4.0. If the Response Type is nonBlockingRequestSync, it waits for an acknowledgement Response primitive and goes to step Orig-4.1. If the Response Type is nonBlockingRequestAsynch, it waits for an acknowledgement Response primitive and goes to step Orig-4.1. If the ***Response Type*** is flexBlocking, the Originator shall wait for a Response primitive as in Orig-4.0 and Orig-4.1 below, if the Response primitive is an acknowledgement it shall proceed according to Orig-4.1 (nonBlockingRequestSynch or nonBlockingRequestAsynch) otherwise it shall proceed according to Orig-4.0 (blockingRequest).

Orig-4.0 and Orig-4.1 "Wait for Response primitive": Refer to clause 7.3.1.3 for details.

Orig-5.0 "Send a Request primitive with op=R": The op=R means Retrieve operation. The Request primitive shall include the mandatory parameters ***Operation***, ***To***, ***From*** and ***Request*** ***Identifier***. The ***Response Type*** of the "Request" primitive shall be blockingRequest. See clause 7.3.1.4 for details.

Orig-5.1 "Receive a Response primitive from the Hosting CSE": The Originator shall receive the mandatory parameters which are ***Response*** ***Status*** ***Code*** and ***Request*** ***Identifier***. The ***Request*** ***Identifier*** shall be identical to the value of that parameter from Orig-5.0. Depending on the ***Result Content*** and ***Primitive Profile Identifier*** parameter values of the request primitive and the ***Response*** ***Status*** ***Code*** parameter of the response primitive, the Originator may receive a ***Content*** parameter with information about the <request> resource. . When the ***Response*** ***Status*** ***Code*** is successful and ***Content*** parameter exists, it goes to Orig-5.2. When the ***Response*** ***Status* *Code*** is an error such as Originator error (4XXX) or Receiver error (5XXX) or Network error (6XXX) or the ***Content*** parameter is absent, it goes to Orig-5.0

Orig-5.2 "Completion of operation by ***Request Status attribute***: When the ***Request Status*** is COMPLETED, it goes to Orig-5.3. When the ***Request Status* is** PENDING or FORWARDED or PARTIALLY\_COMPLETEDwhich indicates processing at the Receiver, it goes to Orig-5.0. When the ***Request Status*** is FAILED, it goes to finish with error.

Orig-5.3 "Extract a result from Response primitive of Orig-5.1": The information in the *operationResult* attribute of the <request> resource in the ***Content*** parameter from Orig-5.1 is extracted from the Response primitive which included the ***Request*** ***Identifier***, ***Response*** ***Status*** ***Code*** and optional ***Content*** parameters. The <request> resource shall include mandatory attributes as specified in clause 9.6.12 of oneM2M TS-0001 [6]. The ***Request*** ***Identifier*** in the *operationResult* attribute shall be identical to that in Orig‑2.0.

Orig-6.0 "Process Response primitive": The ***Request*** ***Identifier*** shall be identical to that in Orig-2.0. The Originator processes the response.

Orig-7.0 "Receive a Request primitive with op=N": The op=N means Notify operation. The Originator receives a Request primitive with mandatory parameters ***Operation***, ***To***, ***From***, ***Request*** ***Identifier*** and ***Content*** unless the Originator had included a ***Primitive Profile Identifier*** parameter in the Request primitive sent in Orig-2.0 that instructed the Receiver CSE to filter the ***Content*** parameter that the Originator does not want to receive as defined in clause 10.2.25 in oneM2M TS-0001 [6]. The ***Operation*** parameter shall be Notify. The ***Content*** parameter is the notification information as specified in clause 7.5.1.1.

Orig-8.0 "Create a Response primitive": The Originator creates Response primitive with mandatory parameters ***Response*** ***Status*** ***Code*** and ***Request*** ***Identifier***. The ***Request*** ***Identifier*** shall be identical to that in Orig-7.0.

Orig-9.0 "Send a Response primitive": The Response primitive which is created at Orig-8.0 shall be sent to the Receiver. Refer to clause 7.3.2.3 for details.

Orig-9.1 "Extract Response primitive of Orig-2.0 from Orig-7.0": The information in the *operationResult* attribute of the <request> resource from Orig-7.0 in Response primitive includes ***Request*** ***Identifier***, ***Response*** ***Status*** ***Code*** and optional ***Content*** parameters. The <request> resource shall include mandatory attributes as specified in clause 9.6.12 of oneM2M TS-0001 [6]. The ***Request*** ***Identifier*** in the *operationResult* attribute shall be identical to that in Orig‑2.0.

-------------------------------------------------- End of Change 1---------------------------------------

## ----------------------- Start of Change 2--------------------------------------------

#### 7.3.2.2 Create <request> resource locally

Creation of a <request> resource can only be done implicitly by a Receiver CSE. When the Receiver CSE receives a request in non-blocking mode (i.e. the ***Response Type*** parameter of the request is set to either "nonBlockingRequestSynch" or "nonBlockingRequestAsynch") targeting any other resource type or requesting a notification, and if the Receiver CSE supports the <request> resource type as indicated by the *supportedResourceType* attribute of the <CSEBase> resource, the Receiver CSE shall create an instance of <request> resource based on the following steps. If the Receiver CSE does not support the <request> resource type, the "nonBlockingRequestSynch" request shall be rejected with a ***Response Status Code*** indicating "NON\_BLOCKING\_SYNCH\_REQUEST\_NOT\_SUPPORTED" error. In the case of a "nonBlockingRequestAsynch" request, a Receiver CSE that does not support the *<*request*>* resource type shall respond to an acceptable request with a response containing an Acknowledgement without a reference to a resource containing the context of the request.

The Receiver CSE of a non-blocking request is the Hosting CSE for the <request> resource that shall be associated with the non-blocking request.

1. Assign values to the common attributes of the <request> resource according to Table 7.3.2.2-1.

Table 7.3.2.2‑1: Common attributes settings for <request> resource

|  |  |
| --- | --- |
| Attribute Name | Value |
| *resourceType* | Request. |
| *resourceID* | Hosting CSE shall assign a value to this attribute. |
| *expirationTime* | The value of the *expirationTime* shall be chosen dependent on the ***Request Expiration Timestamp***, ***Result Expiration Timestamp***, ***Operation Execution Time*** and ***Result Persistence***parameters associated with the original request. If the value consistent with the ***Request Expiration Timestamp***, ***Result Expiration Timestamp***, ***Operation Execution Time*** and ***Result Persistence***parameters is too long, the Hosting CSE shall reject the request.  The *expirationTime* of the <request> resource should last longer than the specified ***Result Persistence*** if provided in the request. |
| *parentID* | The parent resource of a <request> resource shall be the <CSEBase> resource of the Hosting CSE. |
| *creationTime* | Date/time of creation of this resource. |
| *lastModifiedTime* | Date/time which is equal to the *creationTime*. |
| *accessControlPolicyIDs* | Populate with the resource identifier of an <accessControlPolicy> that contains the following:  In the privileges attribute:   * Allow RETRIEVE, UPDATE and DELETE operations for the Hosting CSE. * Allow RETRIEVE and DELETE operations for the Originator, i.e. the value of the parameter ***From*** in the associated non-blocking request.   In the selfPrivileges attribute:   1. Allow UPDATE operations for the Originator, i.e. the value of the parameter ***From*** in the associated non-blocking request. |
| *labels* | Originator ID. |
| *resourceName* | Hosting CSE shall assign a value to this attribute. |

1. Assign values to the resource-specific attributes of the <request> resource according to Table 7.3.2.2-2.

Table 7.3.2.2‑2: Resource-specific attributes settings for <request> resource

|  |  |
| --- | --- |
| Attribute Name | Value |
| *operation* | The value of the parameter ***Operation*** in the associated non-blocking request. |
| *target* | The value of the parameter ***To*** in the associated non-blocking request. |
| *originator* | The value of the parameter ***From*** in the associated non-blocking request. |
| *requestID* | The value of the parameter ***Request Identifier*** in the associated non-blocking request. |
| *metaInformation* | The content of this attribute is set to information in optional parameters described in clause 8.1.2 of oneM2M TS-0001 [6] given in the associated non-blocking request. |
| *content* | The value of the parameter ***Content***, if any, in the associated non-blocking request. |
| *requestStatus* | The Receiver CSE shall set this to "PENDING". |

-------------------------------------------------- End of Change 2---------------------------------------

## ----------------------- Start of Change 3 --------------------------------------------

##### 7.4.14.2.5 Aggregation of member responses

After receiving the member responses from the member hosting CSEs, the group-hosting CSE shall respond to the Originator with an aggregated response. To indicate which response is generated by which member resource, the Hosting CSE shall set that member's resource ID into the ***From*** response parameter in each member response.

If ***Response Type*, *Result Expiration Time*** or ***Result Persistence*** were set in the request, these affect the behaviour of the group-hosting CSE as follows:

* If ***Response Type*** is set to **blockingRequest**, the group-hosting CSE shall respond only once with the aggregated response. It shall do this before the time indicated by the ***Result Expiration Time*** is reached. The group-hosting CSE shall discard any member responses received after this time.
* If ***Response Type*** is set to **nonBlockingRequestSynch**, the group-hosting CSE shall create a <request> resource locally and respond the Originator with the address of this <request> resource. Until the ***Result Expiration Time*** is reached***,*** the group-hosting CSE shall aggregate the member responses and include this aggregated response in the *operationResult* of the <request> resource*.*
* If ***Response Type*** is set to **nonBlockingRequestAsynch**, the group-hosting CSE shall notify the Originator or the notification targets with aggregated responses before the ***Result Expiration Time*** expires. The group-hosting CSE may notify the Originator more than once during the period until the ***Result Expiration Time*** expires. Each notification shall contain different member responses.
* If ***Response Type*** is set to **flexBlocking**, the group-hosting CSE shall keep aggregating the member responses until the group-hosting CSE determines that it is time to send a response – this depends on the properties of the group-hosting CSE related to the <group> resource (the number of aggregated responses or the time period of the aggregation). By that time, if the aggregated response contains all the member responses, the group-hosting CSE shall respond with the aggregated response. However if only some of the member responses have been received, the group-hosting CSE shall create a <request> resource from the received request, and respond to the Originator with the reference to the created <request> resource as well as the currently aggregated responses. Until the time specified in ***Result Expiration Time*** is reached, the group-hosting CSE shall keep aggregating the remaining member responses and updating the aggregated response in the *operationResult* of the <request> resource. If **notificationTarget** is provided in the request, the group-hosting CSE shall notify the Originator with the aggregated response. Each notification shall contain different member responses.

If the group-hosting CSE supports <request> resource, in the **nonBlockingRequestAsynch**, **nonBlockingRequestSynch** and **flexBlocking** case, it shall set the *requestStatus* of the <request> resource to PARTIALLY\_COMPLETED if some of the member responses are received.If the group-hosting CSE has aggregated all the member responses, it shall set the *requestStatus* to COMPLETED.

In any of the cases above, member responses received after the ***Result Expiration Time*** shall be discarded. After the time specified in ***Result Persistence***, the aggregated response shall not be retrievable.

If the group-hosting CSE gets no response before the ***Result Expiration Timestamp*** expiry, then the Hosting CSE shall return error with the ***Response Status Code*** parameter set as "GROUP\_MEMBERS\_NOT\_RESPONDED". Otherwise, the group-hosting CSE shall return the successful ***Response Status Code*** parameter value "OK" regardless of the requested operation. Note that the "OK" successful ***Response Status Code*** parameter is set regardless of the ***Response Status Code*** parameter value in each response primitive from the group member(s).

When aggregating notifications the group-hosting CSE, upon receiving the first notification, shall use the group's *notifyAggregation* attribute and wait until *notifyAggregation*/*number* notifications have been received or until the *notifyAggregation*/*duration* has elapsed, whichever comes first, and send a Notify primitive containing the aggregatedNotification data object defined in Table 7.5.1.1-2. If the *notifyAggregation* attribute is not specified in the <group> resource then the group-hosting CSE shall use the *currentNrOfMembers* attribute of the <group> and a duration specified by the M2M Service Provider instead of the number and duration from the *notifyAggregation* attribute.

If any of the parameters mentioned above are missing from the request, the group-hosting CSE shall determine the time to respond using its local Policy.

-------------------------------------------------- End of Change 2---------------------------------------