|  |
| --- |
|  |

|  |  |
| --- | --- |
| CHANGE REQUEST | |
| Meeting ID:\* | SDS 48 |
| Source:\* | Miguel Angel Reina Ortega, ETSI, [MiguelAngel.ReinaOrtega@etsi.org](mailto:MiguelAngel.ReinaOrtega@etsi.org) |
| Date:\* | 2020-12-02 |
| Reason for Change/s:\* | notificationEventType for timeSeries |
| 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-0001 v3.22.0 |
| Clauses \* | 9.6.8, 10.2.4.29 |
| 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 new notificationEventType which will be used for timeSeries. This will allow:

* Configure properly the subscription to the timeSeries to get notifications on the number of missingDataPoints
* Notifications generated from timeSeries procedure to be clearly distinguished

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

**Table 9.6.8-3: *eventNotificationCriteria* conditions**

| **Condition tag** | **Multiplicity** | **Matching condition** |
| --- | --- | --- |
| *createdBefore* | 0..1 | The *creationTime* attribute of the resource is chronologically before the specified value. |
| *createdAfter* | 0..1 | The *creationTime* attribute of the resource is chronologically after the specified value. |
| *modifiedSince* | 0..1 | The *lastModifiedTime* attribute of the resource is chronologically after the specified value. |
| *unmodifiedSince* | 0..1 | The *lastModifiedTime* attribute of the resource is chronologically before the specified value. |
| *stateTagSmaller* | 0..1 | The *stateTag* attribute of the resource is smaller than the specified value. |
| *stateTagBigger* | 0..1 | The *stateTag* attribute of the resource is bigger than the specified value. |
| *expireBefore* | 0..1 | The *expirationTime* attribute of the resource is chronologically before the specified value. |
| *expireAfter* | 0..1 | The *expirationTime* attribute of the resource is chronologically after the specified value. |
| *sizeAbove* | 0..1 | The *contentSize* attribute of the *<contentInstance>* resource is equal to or greater than the specified value. |
| *sizeBelow* | 0..1 | The *contentSize* attribute of the *<contentInstance>* resource is smaller than the specified value. |
| *notificationEventType* | 0..7 | The type of event that shall trigger a notification. If multiple *notificationEventType* tags are present, a notification shall be triggered if any of the configured events occur. Note that not all permutations of event type are meaningful. Possible notification event type values are:   1. Update to attributes of the subscribed-to resource 2. Deletion of the subscribed-to resource, 3. Creation of a direct child of the subscribed-to resource, 4. Deletion of a direct child of the subscribed-to resource 5. An attempt to retrieve a <*contentInstance*> direct-child-resource of a subscribed-to <*container*> resource is performed while this <*contentInstance*> child resource is an obsolete resource or the reference used for retrieving this resource is not assigned. This retrieval is performed by a RETRIEVE request targeting the subscribed-to resource with the Result Content parameter set to either "child-resources" or "attributes+child-resources". This value for the *eventNotificationType* tag implies that the subscribed-to resource shall be an <*container*> resource. Otherwise this setting is not valid. 6. Trigger Received targeting the MN/ASN-AE associated with the <AE> parent resource. This implies that the subscribed-to resource shall be an <*AE*> resource instance. Otherwise this setting is not valid. 7. Update to attributes of thesubscribed-to resource with blocking of the triggering UPDATE operation. For this *eventNotificationType* value setting, only one single Notification Target shall be present in the *notificationURI* attribute – see *notificationURI* attribute definition. This value for the *eventNotificationType* tag shall not be combined with any other *eventNotificationType* tag value. This value for *notificationEventType* establishes a subscription that is triggered for the same events as for the value “Update to attributes of the subscribed-to resource”. However, upon occurrence of a triggering UPDATE operation that has been validated and results in an authorized UPDATE operation, the triggering UPDATE operation shall be blocked by the Hosting CSE until a notification request was sent out and a corresponding response message was received or a timeout happens. When the response status code of the notification response message indicates a successful notification reception in combination with a successful notification action taken by the Notification Target entity, the triggering UPDATE operation shall be completed with a successful update of the targeted attribute(s). If the notification response message indicates an unsuccessful notification reception or a successful notification reception with unsuccessful notification action by the targeted entity or times out, the blocked UPDATE operation shall be completed with no success and no change of the targeted attribute(s). For any subscribed-to resource there shall exist a maximum of one subscription with this setting of *notificationEventType*. All other notification policies shall not be allowed when this setting of *notificationEventType* is used. The *notificationContentType* shall be “modified attibutes”. When an UPDATE operation has been blocked due to triggering this type of notification, any other occurring UPDATE or DELETE requests to the same resource shall be handled only after the blocked UPDATE operation has been completed. 8. Report on generated missing data points   The other conditions in *eventNotificationCriteria* conditions apply within the scope of the selected *notificationEventType.*  For example, if notificationEventType is "Creation of a direct child of the subscribed-to resource" then other *eventNotificationCriteria* conditions is applied to the direct child resources of the subscribed-to resource.  If this condition is not specified, the default value is "Update to attributes of the subscribed-to resource". This default value shall apply only if *operationMonitor* is not present in the resource.  The notion of "obsolete resource" is defined in clause 9.6.1.3.2 (Common attributes). |
| *operationMonitor* | 0..n | The operations and/or the Originators accessing the subscribed-to resource matches with the specified value. It allows monitoring which operation and/or which Originator is attempting to the access subscribed-to resource regardless of whether the operation is performed. This feature is useful to detect AEs that send requests to a subscribed-to resource and that result in a successful or failure response. Possible arguments are operation(s) (e.g.: CREATE, RETRIEVE, UPDATE, DELETE, NOTIFY) and/or Originator identifier(s).  If a set of Originator identifier(s) is included in this tag and no operations are listed, any operations initiated from any of the indicated Originator(s) shall trigger a notification.  If a set of operation(s) is included in this tag and no Originator identifier, any of the listed operations shall trigger a notification.  If both, a set of Originator identifiers and a set of operations are listed, then any of the listed operations initiated from any of the listed Originators shall trigger the notification. |
| *attribute* | 0..n | A list of attribute names of a subscribed-to-resource. This list is only applicable when *notificationEventType* has a value of "Update to attributes of the subscribed-to resource". or “Update to attributes of the subscribed-to resource with blocking of the triggering UPDATE operation”.  If this list is present, then it is used to specify a subset of a subscribed-to resource's attributes for which updates shall result in a notification. If ANY attribute specified on this list is updated, then a notification shall be generated. If an attribute that is not specified in this list is updated, then a notification shall not be generated.  If this list is not presented, then the default attribute list is the full set of a subscribed-to resource's attributes. If ANY attribute of a subscribed-to resource is updated, then a notification shall be generated. |
| *childResourceType* | 0.. 1 (L) | A list of resource types. This list is only applicable when *notificationEventType* has a value of "Creation of a direct child of the subscribed-to resource" or “Deletion of a direct child of the subscribed-to resource”.  If this list is present, then it is used to specify a subset of resource type for direct child resource of which creation or deletion shall result in a notification. If ANY resource type specified on this list is created or deleted, then a notification shall be generated. If a resource type that is not specified in this list is created or deleted, then a notification shall not be generated.  If this list is not present, then the default resource type list is the full set of a direct child resource. |
| *missingData* | 0..1 | The *missingData* includes two values: a minimum specified missing number of the Time Series Data within the specified window duration, and the window duration. The condition only applies to subscribed-to resources of type *<timeSeries>*. This condition only applies when *notificationEventType* has a value of “Report of generated missing data points”. If this attribute is modified by an UPDATE the associated timer/counter are stopped and restarted with the new values.  The first detected missing data point starts the timer associated with the window duration.  The window duration is restarted upon its expiry until such time as the entire subscription is terminated or not refreshed. More details about NOTIFICATIONS related to data reporting is found in section 10.2.4.29 |
| *filterOperation* | 0..1 | Indicates the logical operation (AND/OR/XOR) to be used for the condition tags *createdBefore, createdAfter, modifiedSince, unmodifiedSince, stateTagSmaller, stateTagBigger, expireBefore, expireAfter, sizeAbove, sizeBelow*. The default value is logical AND. |

The rules when multiple conditions are used together shall be as follows:

Different condition tags shall use the "AND/OR/XOR" logical operation based on the *filterOperation* specified;

Same condition tags shall use the "OR" logical operation.

The XOR operation evaluates to true if and only if an odd number of its inputs are true.

No mixed AND/OR/XOR filter operation will be supported.

Table 9.6.8-4 defines the default and allowed values of *notificationContentType* for each of the supported values of *notificationEventType*.

**Table 9.6.8-4:** **Default and allowed values of *notificationContentType***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **notificationEventType**  **notificationContentType** | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** |
| "modified attributes"; | valid | n/a | n/a | n/a | n/a | n/a | valid (default) | n/a |
| "all attributes"; | valid (default) | valid (default) | valid (default) | valid (default) | valid (default) | n/a | n/a | valid (default) |
| "ID" of the resource indicated in the *notificationEventType* condition. | valid | valid | valid | valid | valid | n/a | n/a | n/a |
| “Trigger Payload” | n/a | n/a | n/a | n/a | n/a | valid (default) | n/a | n/a |

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

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

10.2.4.29 Procedure for Time Series Data Detecting and Reporting

In the case that the *periodicInterval* is set and the *missingDataDetect* is TRUE, the Hosting CSE shall monitor the Time Series Data based on its *periodicInterval*. When the Hosting CSE detects a missing data point, the *dataGenerationTime* of the missing data point is inserted into the *missingDataList* attribute and the *missingDataCurrentNr* shall be increased by one. When the *missingDataCurrentNr* reaches the *missingDataMaxNr,* the oldest *dataGenerationTime* shall be removed from *missingDataList* to enable the insertion of the new missing data point information.

When an AE wants to be informed of the number of missing data points in a given renewable time duration, the AE shall request the creation of a *<subscription>* resource and set the *missingData* in the *eventNotificationCriteria* conditions to specify the reporting policy. The *notificationEventType* element in the *eventNotificationCriteria* shall have a value of “Report of generated missing data points”. This enables the AE to keep track of the number of missing data points and the corresponding time-stamps over a predefined but renewable duration (i.e. the “window duration” of the *missingData condition*).

When the Hosting CSE reports missing data points, it shall check the *missingData* condition in the subscription resources created for that purpose.

When the first missing data point is detected (i.e. a detection of the first discontinuous time-stamp) following the creation of a subscription, the Hosting CSE shall start a timer associated with that subscription and start counting the number of missing data points. The timer is set according to the “window duration” in the subscription’s *missingData*  condition. The reporting policy is governed by the rules below:

* If the total number of missing data points becomes equal to the “minimum specified missing number of the Time Series Data” specified in the subscription’s *missingData* conditionbefore the timer expires, a NOTIFY request shall be sent including the "number of missing data points" that have been detected since the start of the subscription’s timer. The missing data points counter shall continue counting while the timer continues to run (since it did not expire). A similar NOTIFY request shall be sent for each subsequent missing data point detected until the timer expires (see next bullet for behavior when the timer expires).
* If the timer expires, the missing data points counter is reset back to 0. The timer is restarted upon detection of next missing data.
* The reset of the timer and the missing data points counter upon timer expiry shall continue until such time as the subscription is cancelled or terminated.
* If no missing data points have been detected at all during the life time of a subscription, then no timer shall be started at all. But once a timer is started triggered by the first missing data point, then the above rules in the previous bullets shall apply.

Figure 10.2.4.29-1depicts the above rules.

****

**Figure 10.2.4.29-1: Time Series Data Detecting and Reporting Mechanism**

T1: when the first missing data point is detected, the timer is started and the missing data points counter is set to a value of 1.

T2: the NOTIFY Request is sent when the value of the missing data points counter becomes equal to the value in the missingData attribute.

T3: the NOTIFY Request is sent again because the missing data points counter is greater than the value in the missingData attribute.

T4: at the end of the “window duration” the missing data points counter is reset back to 0.

T5: the “window duration” timer is restarted since amissing data point is detected and the missing data points counter is set to a value of 1.

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