|  |
| --- |
|  |

|  |
| --- |
| CHANGE REQUEST |
| Meeting ID:\* | SDS 49 |
| Source:\* | Miguel Angel Reina Ortega, ETSI, MiguelAngel.ReinaOrtega@etsi.org Neeta Meshram, C-DOT, neeta@cdot.inBob Flynn, Exacta GSS, bob.flynn@exactagss.com  |
| Date:\* | 2021-02-03 |
| Reason for Change/s:\* | notificationEventType for timeSeries |
| CR against: Release\* | Rel-4  |
| CR against: WI\* | [x]  Active < WI-0077> [ ]  MNT maintenance / < Work Item number(optional)>Is this a mirror CR? Yes [x]  No [ ] mirror CR number: SDS-2020-0362R04-TS-0004\_notificationEventType\_for\_timeSeries\_R3[ ]  STE Small Technical Enhancements / < Work Item number (optional)>Only ONE of the above shall be ticked |
| CR against: TS/TR\* | TS-0004 v4.4.0 |
| Clauses \* | 6.3.4.2.19, 6.3.5.7, 7.5.1.2.9, 7.4.8.2.1, 7.4.8.2.3 |
| Type of change: \* | [ ]  Editorial change[ ]  Bug Fix or Correction[x]  Change to existing feature or functionality[ ]  New feature or functionalityOnly ONE of the above shall be ticked |
| Other TS/TR(s) impacted |  TS-0001 |
| Post Freeze checking:\* | This CR contains only essential changes and corrections? YES [x]  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

This CR also provides an update to the Create and Update subscription procedures to check presence of missingData when notificationEventType is provided and has “Report on generated missing data points” as a value.

And finally, the CR proposes a new value for notificationContentType specific for timeSeries notifications.

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

6.3.4.2.19 m2m:notificationEventType

Used for ***eventNotificationCriteria*** conditions and in the *notificationEvent*element.

**Table 6.3.4.2.19‑1: Interpretation of notificationEventType**

|  |  |  |
| --- | --- | --- |
| **Value** | **Interpretation** | **Note** |
| 1 | Update\_of\_Resource | Default |
| 2 | Delete\_of\_Resource |  |
| 3 | Create\_of\_Direct\_Child\_Resource |  |
| 4 | Delete\_of\_Direct\_Child\_Resource |  |
| 5 | Retrieve\_of\_Container\_Resource\_With\_No\_Child\_Resource | Context: A RETRIEVE request targets a subscribed-to <container> resource with the Result Content parameter set to either "child-resources" or "attributes+child-resources".A notification is initiated if the <contentInstance> child resource is obsolete or not present in the targeted parent resource. |
| 6 | Trigger\_Received\_For\_AE\_Resource | Context: A notification is initiated when a Trigger is Received by a Registrar CSE targeting the AE-ID associated with the <*AE*> resource of a Registree AE. |
| 7 | Blocking\_Update |  |
| 8 | Report on generated missing data points | Context: A notification is initiated when a number of missing data points in a given duration are detected according to missingData attribute in <*subscription*> resource |

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

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

6.3.5.7 m2m:eventNotificationCriteria

Used for the *eventNotificationCriteria* attribute of the <subscription> resource.

**Table 6.3.5.7‑1: Type Definition of m2m:eventNotificationCriteria**

|  |  |  |  |
| --- | --- | --- | --- |
| **Element Path** | **Element Data Type**  | **Multiplicity** | **Note** |
| createdBefore | m2m:timestamp | 0..1 |  |
| createdAfter | m2m:timestamp | 0..1 |  |
| modifiedSince | m2m:timestamp | 0..1 |  |
| unmodifiedSince | m2m:timestamp | 0..1 |  |
| stateTagSmaller | xs:positiveInteger | 0..1 |  |
| stateTagBigger | xs:nonNegativeInteger | 0..1 |  |
| expireBefore | m2m:timestamp | 0..1 |  |
| expireAfter | m2m:timestamp | 0..1 |  |
| sizeAbove | xs:nonNegativeInteger | 0..1 |  |
| sizeBelow | xs:positiveInteger | 0..1 |  |
| operationMonitor | m2m:operationMonitor | 0..n |  |
| attribute | m2m:attributeList | 0..1 |  |
| notificationEventType | m2m:notificationEventType | 0..7 |  |
| childResourceType | list of m2m:resourceType | 0..1 |  |
| missingData | m2m:missingData | 0..1 |  |
| filterOperation | m2m:filterOperation | 0..1 |  |

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

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

7.5.1.2.9 Notification for missing Time Series Data

***Originator(Hosting CSE):***

No change from the procedures in clause 7.2.2.1 except the following addition in Step1.0:

See details in oneM2M TS-0001 [Error: Reference source not found], clause 10.2.4.29.

-------------------------------------------------- End of Change 3---------------------------------------

## ----------------------- Start of Change 4 --------------------------------------------

<xs:complexType name="notification">

 <xs:sequence>

 <xs:element name="notificationEvent" minOccurs="0">

 <xs:complexType >

 <xs:sequence>

 <xs:element name="representation" type="m2m:representation" minOccurs="0"/>

 <xs:element name="operationMonitor" minOccurs="0">

 <xs:complexType>

 <xs:sequence>

 <xs:element name="operation" type="m2m:operation" minOccurs="0"/>

 <xs:element name="originator" type="m2m:ID" minOccurs="0"/>

 </xs:sequence>

 </xs:complexType>

 </xs:element>

 <xs:element name="notificationEventType" type="m2m:notificationEventType" minOccurs="1"/>

 </xs:sequence>

 </xs:complexType>

 </xs:element>

-------------------------------------------------- End of Change 4---------------------------------------

## ----------------------- Start of Change 5 --------------------------------------------

7.4.8.2.1 Create

***Originator:***

No change from the generic procedures in clause 7.2.2.1.

If *missingData* condition specifies the duration value greater than the *periodicInterval* attribute of the <timeSeries> resource no notification on missing data points will be generated.

***Receiver:***

The following are additional Hosting CSE procedures to the generic resource handling procedures (Figure 7.2.2.2‑1 in clause 7.2.2.2). The additional procedures shall be inserted from Recv-6.2 to Recv-6.5 as below.

Recv-6.3 The following step is in addition to the procedures defined in clause 7.3.3.15:

Check if the Originator has privileges for retrieving the subscribed-to resource. If the Originator does not have the privilege, the Hosting CSE shall return the response primitive with ***Response Status Code*** indicating "ORIGINATOR\_HAS\_NO\_PRIVILEGE" error.

Recv-6.4 The following steps are in addition to the procedures defined in clause 7.3.3.3:

1. Check if the subscribed-to resource, addressed in the ***To*** parameter in the Request, is subscribable. Subscribable resource types are defined in TS-0001 [6]; they have <subscription> resource types as their child resources. If it is not subscribable, the Hosting CSE shall return the Notify response primitive with a ***Response Status Code*** indicating "TARGET\_NOT\_SUBSCRIBABLE" error instead of the ***Response Status Code*** "INVALID\_CHILD\_RESOURCE\_TYPE".
2. Check if the *notificationEventType* is set to "Blocking\_Update".

If the subscribed-to resource already has a subscription with this *notificationEventType* the Hosting CSE shall return the response primitive with ***Response Status Code*** indicating "BLOCKING\_SUBSCRIPTION\_ALREADY\_EXISTS" error if more than one notification of this type could be sent.

If there is more than one *notificationURI* specified, the Hosting CSE shall return the response primitive with ***Response Status Code*** indicating "BAD\_REQUEST" error.

If any resource-specific attributes of the <subscription> resource other than *eventNotificationCriteria* or *notificationURI* are specified the Hosting CSE shall return the primitive with ***Response Status Code*** indicating "BAD\_REQUEST" error.

If any condition tag of the *eventNotificationCriteria* attribute other than *attribute* condition tag is specified, the Hosting CSE shall return the response primitive with ***Response Status Code*** indicating "BAD\_REQUEST" error.

1. Check if the missingDdata element of eventNotificationCriteria is provided.
	1. if the subscribed-to resource, ***To*** parameter in the Request is not <timeSeries>, the request shall be rejected with ***Response Status Code*** indicating "BAD\_REQUEST" error.***.***
2. Check if the *notificationEventType* is set to “Report on generated mising data points” and *missingData* attribute is not provided, the request shall be rejected with ***Response Status Code*** indicating "BAD\_REQUEST" error***.***
3. If any of the *notificationURI* entries are not the Originator, the Hosting CSE may send a Subscription Verification request primitive to each of them as described in clause 7.5.1.2.3.

a) If the Hosting CSE cannot send one or more Subscription Verification request primitives, the Hosting CSE shall return the Create <subscription> response primitive with a ***Response Status Code*** indicating "SUBSCRIPTION\_VERIFICATION\_INITIATION\_FAILED" error.

b) If the Hosting CSE sent all the Subscription Verification request primitives, the Hosting CSE shall check if each Notify response primitive contains a ***Response Status Code*** indicating "OK". If not, the Hosting CSE shall return the Create <subscription> response primitive containing the ***Response Status Code*** indicating "SUBSCRIPTION\_VERIFICATION\_INITIATION\_FAILED" error.

1. If the *associatedCrossResourceSub* is provided, check that the Hosting CSE ID value in the *associatedCrossResourceSub* is the same as the ***From*** parameter of the request. If not, return the response primitive with a ***Response Status Code*** indicating "BAD\_REQUEST".
2. If the Originator provides a value of *childResourceType* which is not a valid child of the subscribed-to resource, the request shall be rejected with ***Response Status Code*** indicating “BAD\_REQUEST” error***.***
3. If both the *notificationEventType* and *operationMonitor* are present in the Request, the request shall be rejected ***Response Status Code*** indicating "BAD\_REQUEST" error***.***
4. If the *notificationContentType* is invalid for a given operation (refer to oneM2M TS-0001 [6] Table 9.6.8-4: Default and allowed values of *notificationContentType*) the request shall be rejected with ***Response Status Code*** indicating "BAD\_REQUEST" error***.***

Recv-6.5: The following steps are in addition to the procedures defined in clause 7.3.3.5:

1. If the Originator does not provide *notificationContentType*, the Hosting CSE shall set it according to the default shown in oneM2M TS-0001 [6] Table 9.6.8-4: Default and allowed values of *notificationContentType.*
2. If the *notificationURI* is not the Originator, the Hosting CSE shall set the Originator’s ID as the <subscription> resource's *creator* attribute.
3. If the *batchNotify* attribute is present in the Request but *batchNotify*/*duration* is not provided by the Originator, the Hosting CSE shall set the value of *batchNotify*/*duration* to the default duration as given by the M2M Service Provider.
4. If the *notificationStatsEnable* attribute is set to true, the Hosting CSE shall start recording notification statistics in the *notificationStatsInfo* attribute once the <subscription> resource is created.

-------------------------------------------------- End of Change 5---------------------------------------

## ----------------------- Start of Change 6 --------------------------------------------

7.4.8.2.3 Update

***Originator:***

The following change from the generic procedures in clause 7.2.2.1.

If *missingData* condition specifies the duration value greater than the *periodicInterval* attribute of the <timeSeries> resource no notification on missing data points will be generated.

Orig-1.0: The originator shall not specify *notificationEventType* set to "Blocking\_Update".

***Receiver:***

The following are additional Hosting CSE procedures to the generic resource handling procedures in clause 7.2.2.2.

Recv-6.4: The following steps are in addition to the procedures defined in clause 7.3.3.4:

1. Check if the *notificationEventType* in the request is set to "Blocking\_Update". If so, the request shall be rejected with a "BAD\_REQUEST" ***Response Status Code***.
2. Check if the missing data is provided.
	1. if the subscribed-to resource, ***To*** parameter in the Request is not <timeSeries>, the request shall be rejected with ***Response Status Code*** indicating "BAD\_REQUEST" error***.***
3. Check if the *notificationEventType* in the request is set to “Report of generated missing data points” and *missingData* attribute is not set or provided in the request, the request shall be rejected with ***Response Status Code*** indicating "BAD\_REQUEST" error***.***
4. If the Originator provides a value of *childResourceType* which is not a valid child of thesubscribed-to resource, the request shall be rejected with ***Response Status Code*** indicating “BAD\_REQUEST” error.
5. If the UPDATE operation would result in both *operationMonitor* and *notificationEventType* being present in the resource, the request shall be rejected with ***Response Status Code*** indicating “BAD\_REQUEST” error***.***
6. Check if a new *associatedCrossResourceSub* is provided. If so, check that the Hosting CSE ID value in the *associatedCrossResourceSub* is the same as the ***From*** parameter of the request.
7. If the *notificationContentType* is invalid for a given operation (refer to oneM2M TS-0001 [6] Table 9.6.8-4: Default and allowed values of *notificationContentType*) the request shall be rejected with ***Response Status Code*** indicating “BAD\_REQUEST” error***.***

Recv-6.5. The following step is in addition to the procedures defined in clause 7.3.3.7:

1. If a <crossResourceSubscription> Hosting CSE ID is removed from *associatedCrossResourceSub*, the Hosting CSE shall send a Notify request for Subscription Deletion, using the procedures in clause 7.5.1.2.4, to the <crossResourceSubscription> Hosting CSE*.*
2. If the *notificationStatsEnable* attribute in the resource is true and the *notificationStatsEnable* attribute in the request is false, the Hosting CSE shall stop collecting notification statistics for the *<*subscription*>* resource. The Hosting CSE shall maintain the current value of the *notificationStatsInfo* attribute.
3. If the *notificationStatsEnable* attribute in the resource is false and the *notificationStatsEnable* attribute in the request is true, the Hosting CSE shall update the value of the *notificationStatsEnable* attribute in the resource to true, delete any values stored in the *notificationStatsInfo* attribute of the resource and then start recording notification statistics.

-------------------------------------------------- End of Change 6---------------------------------------

## ----------------------- Start of Change 7 --------------------------------------------

6.3.4.2.18 m2m:notificationContentType

**Table 6.3.4.2.18‑1: Interpretation of notificationContentType**

|  |  |  |
| --- | --- | --- |
| **Value** | **Interpretation** | **Note** |
| 1 | All Attributes |  |
| 2 | Modified Attributes |  |
| 3 | ResourceID |  |
| 4 | Trigger Payload |  |
| 5 | TimeSeries notification |  |
| NOTE: See clause 7.4.8 "Resource Type subscription". |

-------------------------------------------------- End of Change 7---------------------------------------

## ----------------------- Start of Change 8 --------------------------------------------

6.3.5.62 m2m:representation

Used for the *representation* element in the *notificationEvent* element of a notification. Table 6.3.5.62‑1 defines what shall be included in the representation element depending on the value of the *notificationContentType* of the <subscription> resource which triggered the notification.

**Table 6.3.5.62‑1: Elements used for representation element**

|  |  |  |
| --- | --- | --- |
| **Value of notificationContentType** | **Name of Global Element** | **Defined in**  |
| 1, 2 | m2m:<resourceType>{other namespace identifier}:<resourceType> | CDT-<resourceType>.xsd |
| 3 | m2m:URI | CDT-responsePrimitive.xsd |
| 4 | m2m:triggerPayload | CDT-triggerPayload.xsd |
| 5 | m2m:timeSeriesNotification | CDT-timeSeriesNotification-v3\_18\_0.xsd |

The XML representation element shall include a root element which is associated with an XSD Global Element. The root element shall be prefixed with a namespace prefix identifier (e.g. *m2m:*) specified in the associated XSD which defines the respective Global Element. The *representation*element allows the inclusion of namespaces other than m2m.

-------------------------------------------------- End of Change 8---------------------------------------

## ----------------------- Start of Change 9 --------------------------------------------

6.3.5.69 m2m:timeSeriesNotification

It defines the notification data object to be included in the *representation* element in the *notificationEvent* element of a notification for notifications generated for timeSeries. Table 6.3.5.69‑1 defines what shall be included in the timeSeriesNotification element:

**Table 6.3.5.69‑1: Elements of the timeSeriesNotification**

|  |  |  |  |
| --- | --- | --- | --- |
| **Element Path** | **Element Data Type**  | **Multiplicity** | **Note** |
| missingDataList | m2m:missingDataList | 1 | It indicates the missing data points detected since the last timeSeriesNotification |
| missingDataCurrentNr | xs:nonNegativeInteger | 1 | It indicates the number of missing data points detected since the start of the subscription’s timer |

-------------------------------------------------- End of Change 9---------------------------------------

##### ----------------------- Start of Change 10 -------------------------------------------7.5.1.2.2 Notification for <subscription> resources

When the notification message is forwarded or aggregated by transit CSEs, the Originator or a transit CSE shall check whether there are notification policies to enforce between subscription resource Hosting CSE and the notification target. In that case, the transit CSE as well as the Originator shall process Notify request primitive(s) by using the corresponding policy and send processed Notify request primitive(s) to the next CSE with notification policies related to the enforcement so that the transit CSE is able to enforce the policy defined by the subscriber. The notification policies related to the enforcement at this time is verified by using the subscription reference in the Notify request primitive. In the notification policies, the *latestNotify* attribute is only enforced in the transit CSE as well as the Originator.

If ***Event Category*** parameter is set to "latest" in the notification request primitive, the transit CSE as well as Originator shall cache the most recent Notify request. That is, if a new Notify request is received by the CSE with a subscription reference that has already been buffered for a pending Notify request, the newer Notify request will replace the buffered older Notify request.

***Originator:***

When an event is generated, the Originator shall execute the following steps in order:

Step 1.0 Check the *eventNotificationCriteria* attribute of the <subscription> resource associated with the modified resource:

If the *eventNotificationCriteria* attribute is set, then the Originator shall check whether the corresponding event matches with the event criteria. If multiple matching conditions of different types (i.e. different condition tags) are present in the *eventNotificationCriteria* attribute, then the combined condition shall be derived by applying the logical operation specified by the *filterOperation* condition. By default the logical AND operation shall be used if the *filterOperation* condition is not present.

If *notificationEventType* is not set within the *eventNotificationCriteria* attribute and the *operationMonitor* is also not present, the Originator shall use the default setting of "Update\_of\_Resource" to compare against the event.

If the *notificationEventType* has the value "Create\_of\_Direct\_Child\_Resource" or "Delete of Direct Child Resource" and the *childResourceType* condition is also present, then the matching event shall only be detected if one of the child resource types present in the list has been created or deleted, respectively. If the *childResourceType* condition is not present then a matching event is generated whenever any child resource is created or deleted.

If the *notificationEventType* has either an explicit or default value of "Update\_of\_Resource" and the *attribute* condition is also present then the matching event shall only be detected if one of the attributes in the list has been updated. If the *attribute* condition is not present then a matching event is generated whenever any attribute has been updated.

If the event matches, go to the step 2.0. Otherwise, the Originator shall discard the corresponding event.

If the *eventNotificationCriteria* attribute is not configured, the Originator shall use the default setting of "Update\_of\_Resource" for the *notificationEventType* and then continue with the step 2.0.

Step 2.0 The Originator shall check the notification policy as described in the below steps, but the notification policy may be checked in different order. After checking the notification policy in step 2.0 (i.e. from step 2.1 to step 2.6), then continue with step 3.0.

Step 2.1 The Originator shall determine the type of the notification per the *notificationContentType* attribute. The possible values of for *notificationContentType* attribute are "Modified Attributes", "All Attributes", "ResourceID", "Trigger Payload" or “TimeSeries notification”. This attribute may be used jointly with *eventType* attribute in the *eventNotificationCriteria* to determine if it is the attributes/resourceID of the subscribed-to resource or the attributes/resourceID of the child resource of the subscribed-to resource that shall be returned in the notification:

If the value of *notificationContentType* is set to "Modified Attributes", the Notify request primitive shall include the partial resource containing modified attribute(s) only (Refer to clause 7.2.1.2 for response content description).

If the value of *notificationContentType* is set to "All Attributes", the Notify request primitive shall include the complete resource with all attributes (Refer to clause 7.2.1.2 for response content description).

If the value of *notificationContentType* is set to "ResourceID", the Notify request primitive shall include the URI of the resource (Refer to clause 7.2.1.2 for response content description).

If the value of *notificationContentType* is set to "Trigger Payload", the Notify request primitive shall include the trigger payload (Refer to clause 9.2.1 for trigger payload description).

If the value of *notificationContentType* is set to "TimeSeries notification", the Notify request primitive shall include the timeSeriesNotification (Refer to clause 6.3.5.69 for timeSeriesNotification description).

Step 2.2 Check the *notificationEventCat* attribute:

If the *notificationEventCat* attribute is set, the Notify request primitive shall employ the ***Event Category*** parameter as given in the *notificationEventCat* attribute. Then continue with the step 2.3.

If the *notificationEventCat* attribute is not configured, then continue with step 2.3.

Step 2.3 Check the *latestNotify* attribute:

If the *latestNotify* attribute is set, the Originator shall assign ***Event Category*** parameter of value "latest" of the notifications generated pertaining to the subscription created.

Step 2.4 Check the batching notifications policy:

See details in oneM2M TS-0001 [6], clause 10.2.10.7.

NOTE: The use of some attributes such as *rateLimit* and *preSubscriptionNotify* is not supported in the present document.

Step 2.5 Check the *notificationURI* attribute:

The Originator shall fetch the *notificationURI* attribute and set the value to the ***To*** parameter of the Notify request. When the *notificationURI* attribute contains more than one target, the Originator shall generate each Notify request per target.

If the *notificationURI* attribute includes the notification serialization indication, in form of key-value pair, e.g. "ct=json", after the delimiter "?", the Originator shall serialize the notification for the notification target in that serialization type. The delimiter with the serialization indication shall be removed when the target is set to the ***To*** parameter of the Notify request. Then continue with step 3.0.

Step 3.0 The Originator shall check the notification and reachability schedules, but the notification schedules may be checked in different order:

If the <subscription> resource associated with the modified resource includes a <notificationSchedule> child resource, the Originator shall check the time periods given in the *scheduleElement* attribute of the *<notificationSchedule>* child resource.

Also, the Originator shall check the reachability schedule associated with the Receiver by exploring its <schedule> resource. If reachability schedules are not present in a Node then that Entity is considered to be always reachable.

If notificationSchedule and reachability schedule indicate that message transmission is allowed, then proceed with step 5.0. Otherwise, proceed with step 4.0.

In particular, if the *notificationEventCat* attribute is set to 'immediate' and the <notificationSchedule> resource does not allow transmission, then go to step 5.0 and send the corresponding Notify request primitive by temporarily ignoring the Originator's notification schedule.

Step 4.0 Check the *pendingNotification* attribute:

If the *pendingNotification* attribute is set, then the Originator shall cache pending Notify request primitives according to the *pendingNotification* attribute. The possible values are 'sendLatest' and 'sendAllPending'. If the value of pendingNotification is set to 'sendLatest', the most recent Notify request primitive shall be cached by the Originator and it shall set the ***Event Category*** parameter to "latest". If *pendingNotification* is set to 'sendAllPending', all Notify request primitives shall be cached by the Originator. If the *pendingNotification* attribute is not configured, the Originator shall discard the corresponding Notify request primitive. The processed Notify request primitive by the *pendingNotification* attribute is sent to the Receiver once message transmission becomes possible (see the step 6.0).

Step 5.0 Check the *expirationCounter* attribute:

If the *expirationCounter* attribute is set, then it shall be decreased by one when the Originator successfully sends the Notify request primitive. If the counter equals to zero('0'), the corresponding <subscription> resource shall be deleted. Then end the 'Compose Notify Request Primitive' procedure.

If the *expirationCounter* attribute is not configured, then end the 'Compose Notify Request Primitive' procedure.

When message transmission becomes possible, the Originator shall execute the following steps in order:

Step 6.0 If the *pendingNotification* attribute is set, the Originator shall send the processed Notify request primitive by the *pendingNotification* attribute and then continue with the step 7.0

Step 7.0 Check the *expirationCounter* attribute:

If the *expirationCounter* attribute is set, then its value shall be decreased by one when the Originator successfully sends the Notify request primitive. If the counter meets zero, the corresponding <subscription> resource shall be deleted. Then end the 'Compose Notify Request Primitive' procedure.

If the *expirationCounter* attribute is not configured, then end the 'Compose Notify Request Primitive' procedure.

***Receiver:***

When the Hosting CSE receives a Notify request primitive, the Hosting CSE shall check validity of the primitive parameters. In case the Receiver is a transit CSE which forwards or aggregates Notify request primitives before sending to the subscriber or other transit CSEs, upon receiving the Notify request primitive with the ***Event Category*** parameter set to "latest", the Receiver shall identify the latest Notify request primitive with the same subscription reference while storing Notify request primitives locally. When the Receiver as a transit CSE needs to send pending Notify request primitives, it shall send the latest Notify request primitive. When the Receiver as a transit CSE needs to send Notify request primitives, it shall use one of the serializations specified in the subscriber or other transit CSE *contentSerialization* attribute. If there is no *contentSerialization* value specified the transit CSE may use any serialization format.

-------------------------------------------------- End of Change 10---------------------------------------

## ----------------------- Start of Change 11 -------------------------------------------

8.2.7 TimeSeries notification fields

TimeSeries notification fields shall be translated into short names of Table 8.2.7‑1.

**Table 8.2.7‑1: TimeSeries notification field short names**

| **Member Name** | **Short Name** |
| --- | --- |
| *missingDataList* | ***mdlt\**** |
| *missingDataCurrentNr* | ***mdc\**** |
| NOTE: \* marked short names have been already assigned in attribute Table 8.2.3-5. |

-------------------------------------------------- End of Change 11---------------------------------------

## ----------------------- Start of Change 12 -------------------------------------------

<xs:simpleType name="notificationEventType">

 <xs:annotation>

 <xs:documentation>Used in the notificationEventType element of eventNotificationCriteria attribute of subscription resource

 and in the Notification data object</xs:documentation>

 </xs:annotation>

 <xs:restriction base="xs:integer">

 <!-- Update\_of\_Resource -->

 <xs:enumeration value="1" />

 <!-- Delete\_of\_Resource -->

 <xs:enumeration value="2" />

 <!-- Create\_of\_Direct\_Child\_Resource -->

 <xs:enumeration value="3" />

 <!-- Delete\_of\_Direct\_Child\_Resource -->

 <xs:enumeration value="4" />

 <!-- Retrieve\_of\_Container\_Resource\_With\_No\_Child\_Resource -->

 <xs:enumeration value="5" />

 <!-- Trigger\_Received\_For\_AE\_Resource -->

 <xs:enumeration value="6" />

 <!-- Blocking\_Update -->

 <xs:enumeration value="7" />

 <!—- TimeSeries notification -->

 <xs:enumeration value="8" />

 </xs:restriction>

 </xs:simpleType>

<xs:simpleType name="notificationContentType">

 <xs:annotation>

 <xs:documentation></xs:documentation>

 </xs:annotation>

 <xs:restriction base="xs:integer">

 <!-- All attributes -->

 <xs:enumeration value="1" />

 <!-- Modified attributes -->

 <xs:enumeration value="2" />

 <!-- ResourceID -->

 <xs:enumeration value="3" />

 <!-- Trigger Payload -->

 <xs:enumeration value="4" />

 <!—- TimeSeries notification -->

 <xs:enumeration value="5" />

 </xs:restriction>

 </xs:simpleType>

-------------------------------------------------- End of Change 12---------------------------------------

## ----------------------- Start of Change 13 -------------------------------------------

Code for new XSD file CDT-timeSeriesNotification-v3\_18\_0.xsd:

<?xml version="1.0" encoding="UTF-8"?>

<!--

Copyright Notification

The oneM2M Partners authorize you to copy this document, or any components thereof, provided that you retain all copyright and other proprietary notices

contained in the original materials on any copies of the materials and that you comply strictly with these terms.

This copyright permission does not constitute an endorsement of the products or services, nor does it encompass the granting of

any patent rights. The oneM2M Partners assume no responsibility for errors or omissions in this document.

© 2020, oneM2M Partners Type 1 (ARIB, ATIS, CCSA, ETSI, TIA, TSDSI, TTA, TTC). All rights reserved.

Notice of Disclaimer & Limitation of Liability

The information provided in this document is directed solely to professionals who have the appropriate degree of experience to understand

and interpret its contents in accordance with generally accepted engineering or other professional standards and applicable regulations.

No recommendation as to products or vendors is made or should be implied.

NO REPRESENTATION OR WARRANTY IS MADE THAT THE INFORMATION IS TECHNICALLY ACCURATE OR SUFFICIENT OR CONFORMS TO ANY STATUTE,

GOVERNMENTAL RULE OR REGULATION, AND FURTHER, NO REPRESENTATION OR WARRANTY IS MADE OF MERCHANTABILITY OR FITNESS FOR ANY

PARTICULAR PURPOSE OR AGAINST INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS.

NO oneM2M PARTNER TYPE 1 SHALL BE LIABLE, BEYOND THE AMOUNT OF ANY SUM RECEIVED IN PAYMENT BY THAT PARTNER FOR THIS DOCUMENT, WITH RESPECT TO

ANY CLAIM, AND IN NO EVENT SHALL oneM2M BE LIABLE FOR LOST PROFITS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES.

oneM2M EXPRESSLY ADVISES ANY AND ALL USE OF OR RELIANCE UPON THIS INFORMATION PROVIDED IN THIS DOCUMENT IS AT THE RISK OF THE USER.

-->

<xs:schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace="http://www.onem2m.org/xml/protocols"

 xmlns:m2m="http://www.onem2m.org/xml/protocols" elementFormDefault="unqualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">

 <xs:include schemaLocation="CDT-commonTypes-v3\_18\_0.xsd" />

 <xs:element name="timeSeriesNotification">

 <xs:complexType>

 <xs:sequence>

 <xs:element name="missingDataList" type="m2m:missingDataList"/>

 <xs:element name="missingDataCurrentNr" type="xs:nonNegativeInteger"/>

 </xs:sequence>

 </xs:complexType>

 </xs:element>

</xs:schema>

-------------------------------------------------- End of Change 13---------------------------------------