|  |
| --- |
| CHANGE REQUEST |
| Meeting ID:\* | PRO 34.2 |
| Source:\* | Bob Flynn, Convida Wireless; Flynn.bob@convidawireless.com |
| Date:\* | 2018-04-12 |
| Reason for Change/s:\* | serviceSubscribedNode updates |
| CR against: Release\* | Release 3 |
| CR against: WI\* | [x]  Active <WI-0058> [ ]  MNT maintenance / < Work Item number(optional)>Is this a mirror CR? Yes [ ]  No [x] 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-0004 Version 3.7.0 |
| Clauses \* |  |
| 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 |
| Impacted other TS/TR(s) | <TS/TR number>, <Version Number>, and <Description on which aspect should be reflected in this TS/TR> |
| 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 [x]  |
| 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

R01 – fix lines in receiver operations. Add shortname.

Protocol contribution to reflect changes in ARC-2017-0380R01-TS-0001\_NIDD\_Support

*“3GPP Release 13 introduces the ability to send Non-IP data to and from the UE in 3GPP NAS control plane messages. Since no data plane set up is required when sending Non-IP data to / from a UE, this results in optimizations for the both the network and the UE.*

*3GPP Release 15 introduces the SCEF API that supports Non-IP Data Delivery. This API can be used to exchange Non-IP data between an IN-CSE and an MN-CSE, ADN-AE, or ASN-CSE hosted on a UE.*

*This contribution proposes functionality to support the NIDD feature. Specifically the following enhancements are proposed:*

1. *The addition of a niddRequired attribute to the <serviceSubscribedNode> resource to allow an IN-CSE to be provisioned with an indication of whether or not the IN-CSE should perform an NIDD Configuration request to the underlying network for a given UE.*
2. *An optimization to allow an Originator of a request to indicate that a response is not needed from the Receiver of the request. This feature is useful in use cases where an Originator does not care whether the request is successfully performed or not. For example, a IoT sensor that reports periodic sensor readings can use this feature. For NIDD this feature can be useful.”*

Email Comments

1.  The niddRequired bit can be flipped on or off by an Update.  Is there anything special that needs to happen if someone updates it, for example is the CSE required to do a NIDD configuration if the bit is set to true, and the converse (if there is one) when it is set to false? There's no special operation, so is it just a kind of "for info" bit?

A1 – according to TS-0001, table 9.6.20-2 for niddRequired:

*Controls whether the IN-CSE configures the underlying network to enable Non-IP Data Delivery for this node.Valid values are "TRUE" or "FALSE". If not configured, then IN-CSE default policy shall apply. See [15].*

NIDD is a 3GPP term, I am not sure if it applies to any other underlying network. So this attribute would not trigger an operation at the time of the CRUD operation on the attribute, but rather in support of other operations that occur later.

2. Is noResponse something that can be used more generally, i.e. on any request over any protocol?  If so, do we need to update the HTTP binding to say how it is implemented over HTTP?

Q2 – We believe that it can be useful beyond the nidd support that it was specifically added to address. However due to the R3 time constraints, we believe that it is best to restrict this for support of nidd (SCEF) in Release 3 and then address additional bindings in R4.

*If time is short for Rel-3 and we need to freeze stage 3 ASAP, then we can restrict usage of this to Mcn and defer Mca/Mcc to Rel-4.  I think Mca/Mcc support is possible but it will take some work for our existing protocol bindings such as HTTP and CoAP which always includes a oneM2M response.  MQTT is a bit easier I think.  I do see value in supporting this feature for both Mca/Mcc since it can help streamline the oneM2M protocol for “fire and forget” use cases such as periodic sensor reporting.*

If we agree with this, then change 3 and change 4 may be held for R4. And all of the question below get addressed there.

3. Are you allowed to include noResponse on a Retrieve request?  I realise that you can't send an error response back, but you might still want to discourage originators from trying it (e.g. via conformance test).

Q3 – I think this cannot apply to a Retrieve request. CUD seem reasonable though. For example a temp sensor that just pushes temp values periodically and ignores any responses.

Some further ones;

4. Should we disallow values of Result Content other than Nothing if Response Type is noResponse? For example failing a Delete

Q4 – I think that this should only be considered for the SUCCESSFUL responses. If a failure occurs then the CSE should send an appropriate response.

5. What happens if you set Response Type of noResponse and include a Result Expiration Time?

Q5. I think that RET should be ignored if provided with noResponse. Or in the spirit of making the request smaller, issue a bad request message. HOWEVER, we should do this in R4.

6. 7.3.2.8 (Check Hosting CSE of the targeted resource) would apply if Response Type is noResponse.  What would you expect to put in the Response Type of a forwarded request ?

Q6. Based on the above responses, I think we should forward the noResponse. HOWEVER, if an intermediate node forwards this request, then it MAY forget about the request, and therefore be unable to handle a response in the case of an error (as described above). So based on this, I think the receiving CSE should handle this in the following manner: Forward the request with another value. When the response comes back it should drop a successful response and forward any error response. HOWEVER, we should do this in R4.

7. What happens if you set Response Type of noResponse and send the request to a Group?  You haven't proposed any change to 7.4.14.2.5 in this CR.

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

#### 7.4.20.1 Introduction

The <serviceSubscribedNode> resource represents M2M Node information that is needed as part of the M2M Service Subscription resource. It shall contain information about the M2M Node as well as application identifiers of the Applications running on that Node.

The detailed description can be found in clause 9.6.20 in TS-0001 [6].

Table 7.4.20.1‑1: Data type definition of <serviceSubscribedNode> resource

|  |  |  |
| --- | --- | --- |
| Data Type ID | File Name | Note |
| serviceSubscribedNode | CDT-serviceSubscribedNode-v3\_5\_0.xsd |  |

Table 7.4.20.1‑2: Universal/Common Attributes of <serviceSubscribedNode> resource

|  |  |
| --- | --- |
| Attribute Name | Request Optionality  |
| Create | Update |
| @resourceName | O | NP |
| resourceType | NP | NP |
| resourceID | NP | NP |
| parentID | NP | NP |
| expirationTime | O | O |
| accessControlPolicyIDs | O | O |
| creationTime | NP | NP |
| labels | O | O |
| lastModifiedTime | NP | NP |
| *dynamicAuthorizationConsultationIDs* | O | O |

Table 7.4.20.1‑3: Resource Specific Attributes of <serviceSubscribedNode> resource

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Request Optionality  | Data Type | Default Value and Constraints |
| Create | Update |
| nodeID | M | NP | m2m:nodeID |  |
| CSE-ID | O | NP | m2m:ID |  |
| deviceIdentifier | O | NP | list of m2m:deviceID |  |
| ruleLinks | O | O | list of xs:anyURI |  |
| niddRequired | O | O | xs:boolean | No Default. If not configured, then IN-CSE default policy shall apply.  |

Table 7.4.20.1‑4: Child resources of <serviceSubscribedNode> resource

|  |  |  |  |
| --- | --- | --- | --- |
| Child Resource Type  | Child Resource Name | Multiplicity | Ref. to Resource Type Definition |
| <subscription> | [variable] | 0..n | 7.4.8 |

### -----------------------End of change 1-------------------------------------------

### -----------------------Start of change 2-------------------------------------------

* + 1.
1.
2.

##### m2m:responseType

Used for ***Response Type*** parameter (as a part of responseTypeInfo, see Clause 6.3.5.29) in request.

Table 6.3.4.2.6‑1: Interpretation of responseType

|  |  |  |
| --- | --- | --- |
| Value | Interpretation | Note |
| 1 | nonBlockingRequestSynch |  |
| 2 | nonBlockingRequestAsynch |  |
| 3 | blockingRequest |  |
| 4 | flexBlocking |  |
| 5 | noResponse |  |
| NOTE: See clause 6.4.1 "Request message parameter data types". |

### -----------------------End of change 2-------------------------------------------

### -----------------------Start of change 3-------------------------------------------

#### 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": Please refer to clause 7.3.1.1 for details.

Orig-2.0 "Send a Request primitive to the Receiver CSE": The Request primitive shall be included mandatory parameters which are ***Operation***, ***To***, ***From*** and ***Request*** ***Identifier***parameter. Please 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, flexBlocking or noResponse 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 it waits for Response primitive and goes to step Orig-4.0. If the Response Type is nonBlockingRequestSync, it waits for acknowledgement of the Response primitive and goes to step Orig-4.1. If the Response Type is nonBlockingRequestAsynch, it waits for acknowledgement of 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). If the Response Type is noResponse the procedure is complete.

Orig-4.0 and Orig-4.1 "Wait for Response primitive": Please 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 be included mandatory parameters which are ***Operation***, ***To***, ***From*** and ***Request*** ***Identifier***parameter. 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 mandatory parameters which are ***Response*** ***Status*** ***Code***, ***Request*** ***Identifier*** and ***Content*** parameter. A ***Request*** ***Identifier*** shall be identical to the Orig-5.0. An information of ***Content*** parameter is the result of the Orig-2.0 when the Receiver completed handling of Request primitive of Orig-2.0.

Orig-5.2 "Completion of operation by ***Response*** ***Status*** ***Code*** parameter": When the ***Response*** ***Status*** ***Code*** is successful and ***Content*** parameter exists, it goes to Orig-5.3. When the ***Response*** ***Status*** ***Code*** is acknowledgment which indicates processing at the Receiver, it goes to Orig-5.0. When the ***Response*** ***Status*** *Code* is error such as Originator error (4XXX) or Receiver error (5XXX) or Network error (6XXX) or absence of *Content* parameter, it goes to finish with error.

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

Orig-6.0 "Process Response primitive": A ***Request*** ***Identifier*** shall be identical to the 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 Request primitive with mandatory parameters which are ***Operation***, ***To***, ***From***, ***Request*** ***Identifier*** and ***Content*** parameter. An ***Operation*** parameter shall be Notify. A ***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 which are ***Response*** ***Status*** ***Code*** and ***Request*** ***Identifier*** parameter. A ***Request*** ***Identifier*** shall be identical to the 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. Please 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 of operationResult attribute in <request> resource from Orig-7.0 in Response primitive is included ***Request*** ***Identifier***, ***Response*** ***Status*** ***Code*** and optional ***Content*** parameters. The <request> resource shall be included mandatory attributes as specified in clause 9.6.12 of oneM2M TS-0001 [6]. The ***Request*** ***Identifier*** in operationResult attribute shall be identical of Orig-2.0.

### -----------------------End of change 3-------------------------------------------

### -----------------------Start of change 4-------------------------------------------

* + 1.

#### 7.2.2.2 Generic procedure for handling a Request at a receiver

The Receiver shall execute the following steps in order. In case of error in any of the steps below, the Receiver shall execute "Create an error response" (refer to clause 7.3.3.13 for details) and then "Send Response primitive" (refer to clause 7.3.2.4 for details). The corresponding Response code shall be included in the Response primitive.



Figure 7.2.2.2‑1: Generic procedure of Receiver

Recv-1.0 "Check the validity of received request primitive": See clause 7.3.2.1 for details.

Recv-2.0 "Communication method?": The Receiver CSE checks whether a received request is blockingRequest, nonBlockingRequestSynch, nonBlockingRequestAsynch or noResponse by using ***Response Type*** parameter (see detail in clause 8.1.2 in TS-0001 [6]). If the request is blockingRequest or ***Response Type*** parameter is not included, it goes to step Recv-6.0 "Resource handling procedure". If the request is nonBlockingRequestSynch, it goes to step Recv-3.0 "Create <request> resource locally" If the request is nonBlockingRequestAsynch, it goes to step Recv-3.0 "Create <request> resource locally". If the request is flexBlocking, the Receiver CSE shall make the decision to respond using blocking or non-blocking based on its own local context (memory, processing capability, etc.) unless specified further in the resource-specific procedure. If the ***Response Type*** parameter is noResponse it goes to step Recv-6.0 "Resource handling procedure" and the ***Result Content*** parameter is ignored, if present.

Recv-3.0 "Create <request> resource locally": Please refer to clause 7.3.2.2 for details.

Recv-4.0 "Create a successResponse": Please refer to clause 7.3.3.12 for details.

Recv-5.0 "Send Response Primitive": Please refer to clause 7.3.2.4 for details.

Recv-6.0 "Resource handling procedure": Please refer to Figure 7.2.2.2‑2 for details.

Recv-7.0 "Update <request> resource": Please refer to clause 7.3.2.5 for details. This step is only valid when the request is non-blocking.

Recv-8.0 "Send Notification": Please refer to clause 7.5.1.2.5 for details.

Recv-9.0 "Wait for a Response primitive": Please refer to clause 7.3.1.3 for details.

Recv-10.0 "Send Response Primitive": Please refer to clause 7.3.3.16 for details.

Recv-6.10: “Queue request primitive and execute CMDH message forwarding procedure”

Recv-6.1: Hosting CSE of the targeted resource?

Start

Recv-6.3: “Check authorization of the Originator”

Recv-6.4: “Check validity of resource representation for the given resource type”

Recv-6.2: “Check existence of the addressed resource”

Recv-6.5: “Create/Update/Retrieve/Delete/Notify operation is performed”

Recv-6.6: “Announce/De-announce the resource”

Finish

Yes

No

Recv-6.7: “Create a success response”

Recv-6.9: CMDH processing supported?

Recv-6.11: “Forwarding”

No

Yes

Recv-6.0.1: Requested operation is an AE registration?

Recv-6.0.2: “Check Service Subscription Profile”

Yes

No

Recv-6.6.1: “Communication Method?”

Else

blockingRequest

Figure 7.2.2.2‑2: Resource handling procedure

The above figure describes the generic procedure to resource handling procedures.

Recv-6.0.1 "Requested operation is an AE registartion?": If the requested operation is an AE registartion, then it goes to Recv-6.0.2 "Check Service Subscription Profile". Otherwise, it goes to Recv-6.1.

Recv-6.0.2 "Check Service Subscription Profile": Please refer to clause 7.3.2.7 for details.

Recv-6.1 "Hosting CSE of the targeted resource?": The step checks if the receiver is a transit CSE or the Hosting CSE of the received Request by examining the ***To*** parameter of the Request primitive. If the receiver hosts the resource that the address in the ***To*** parameter represents, the receiver is the Hosting CSE (goes to Recv-6.2"Check existence of the addressed resource", Yes branch). Otherwise, the receiver is the Transit CSE (goes to Recv-6.9 "CMDH processing supported?", No branch). Please refer to clause 7.3.2.8 for details.

Recv-6.2 "Check existence of the addressed resource": Please refer to clause 7.3.3.1 for details.

Recv-6.3 "Check authorization of the Originator": Please refer to clause 7.3.3.15 for details.

Recv-6.4 "Check validity of resource representation": Please refer to clause 7.3.3.3 and clause 7.3.3.4 for details. Notify is not applicable for this step.

Recv-6.5 "Create/Update/Retrieve/Delete/Notify operation is performed": The step represents five common operations which are "Create the resource (clause 7.3.3.5)", "Retrieve the resource (clause 7.3.3.6)", "Update the resource (clause 7.3.3.7)", "Delete the resource (clause 7.3.3.8)" and "Notify processing (clause 7.3.3.9)".

Recv-6.6 "Announce/De-announce the resource": The step represents two common operations which are "Announce the resource" and "De-announce the resource". Please refer to clause 7.3.3.10 and clause 7.3.3.11 for details. Notify is not applicable for this step.

Recv-6.6.1 "Communication method?": The Receiver CSE checks whether a received request is blockingRequest or not by using ***Response Type*** parameter (see detail in clause 8.1.2 in TS-0001 [6]). If the request was blockingRequest or ***Response Type*** parameter was not included, it goes to step Recv-6.7 "Create a success response". Otherwise, it goes back to the generic procedure of the receiver (Figure 7.2.2.2‑1).

Recv-6.7 "Create a success response": Please refer to clause 7.3.3.12 for details.

Recv-6.9 "CMDH processing supported?": This step checks whether the Receiver supports the CMDH processing. If the receiver supports CMDH processing, it goes to Recv-6.10 "Queue request primitive and execute CMDH message forwarding procedure" otherwise, it goes to Recv-6.11 “Forwarding”.

Recv-6.10 "Queue request primitive and execute CMDH message forwarding procedure": the Receiver CSE shall queue the received request primitive and execute the "CMDH message forwarding procedure". Please refer to Annex H.2.4. for details.

Recv-6.11 "Forwarding": carry out message forwarding as defined in clause 7.3.2.6.

### -----------------------End of change 4-------------------------------------------

CHECK LIST

* Does this Change Request include an informative introduction containing the problem(s) being solved, and a summary list of proposals.?
* Does this CR contain changes related to only one particular issue/problem?
* Have any mirror CRs been posted?
* Does this Change Request make **all** the changes necessary to address the issue or problem? E.g. A change impacting 5 tables should not include a proposal to change only 3 tables?Does this Change Request follow the drafting rules?
* Are all pictures editable?
* Have you checked the spelling and grammar?
* Have you used change bars for all modifications?
* Does the change 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.)
* Are multiple changes in this CR 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.?