|  |
| --- |
|  |

|  |
| --- |
| CHANGE REQUEST |
| Meeting:\* | ARC#25.2 |
| Source:\* | C-DOT  |
| Date:\* | 2016-11-10 |
| Contact:\* | Poornima (poornima@cdot.in),Suman(ssheoran@cdot.in) ,Anupama(anupama@cdot.in)  |
| Reason for Change/s:\* | See the introduction  |
| CR against: Release\* | Release 2 |
| CR against: WI\* | [ ]  Active <Work Item number> [x]  MNT maintenace / < Work Item number(optional)>[ ]  STE Small Technical Enhancements / < Work Item number (optional)>Only ONE of the above shall be ticked |
| CR against: TS/TR\* | TS-0001 v2.10.1 |
| Clauses/Sub Clauses\* | 8.1.2 |
| 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 |
| 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 [x]  NO [ ] This CR is a mirror CR? YES [ ]  if YES, please indicate the document number of the original CR: : NO [x]   |
| Template Version:27 May 2015 (Dot 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 separated “mirror CR” should be posted at the same time of this CR

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 sections need not show surrounding clauses as long as the proposed section number clearly shows where the new section 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

### PRO-2016-0378R02 contribution was presented in PRO WG, this CR proposes a parameter in request to handle the fanout for only failed members.

For example: when a request is sent to fanoutpoint to create a <resource> in all the members then there is a possibility that CREATE operation is failed on some of the members of a group as highlighted in the table below:

**ResponsePrimitive:**

|  |  |
| --- | --- |
| X-M2M-RSC: 2001 | m2m:agr{m2m:rsp{rsc:2001}…....m2m:rsp{rsc:4103}m2m:rsp{rsc:4105}m2m:rsp{rsc:5106}} |

After processing the response, when Originator finds out that operation has failed for some of the members of group then in the current design either Originator has to send individual request to each member or fanout to complete group as originator doesn’t have option to send fanout request for only failed members.

The CR proposes to give option to the Originator to send fanout to selected members of group by adding a parameter in request.

INFO: The CR will lead to changes in <fanoutpoint> management procedures. The changes can only be initiated after we agree with the idea of sending fanout to the subset of members.

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

### 8.1.2 Request

Requests over the Mca and Mcc reference points, from an Originator to a Receiver, shall contain mandatory and may contain optional parameters. Certain parameters may be mandatory or optional depending upon the Requested operation. In this clause, the mandatory parameters are detailed first, followed by those that are operation dependent, and then by those that are optional:

* ***To*:** Address of the target resource or target attribute for the operation. The ***To*** parameter shall conform to clause 9.3.1.

NOTE 1: ***To*** parameter can be known either by pre-provisioning (clause 11.2) or by discovery (clause 10.2.6 for discovery). Discovery of *<CSEBase>* resource is not supported in this release of the document. It is assumed knowledge of *<CSEBase>* resource is by pre-provisioning only.

NOTE 2: The term target resource refers to the resource which is addressed for the specific operation. For example the ***To*** parameter of a Create operation for a resource *<example>* would be "/m2m.provider.com/exampleBase". The ***To*** parameter for the Retrieve operation of the same resource *<example>* is "/m2m.provider.com/exampleBase/example".

NOTE 3: For Retrieve operation (clause 10.1.2), the ***To*** parameter can be the URI of an attribute to be retrieved.

* ***From*:** Identifier representing the Originator.

The ***From*** parameter is used by the Receiver to check the Originator identity for access privilege verification.

* ***Operation*:** operation to be executed: Create (C), Retrieve (R), Update (U), Delete (D), Notify (N).

 The ***Operation*** parameter shall indicate the operation to be executed at the Receiver:

1. **Create (C): *To*** is the address of the target resource where the new resource (parent resource).
2. **Retrieve (R):** an existing ***To*** addressable resource is read and provided back to the Originator.
3. **Update (U):** the content of an existing ***To*** addressable resource is replaced with the new content as in ***Content*** parameter. If some attributes in the ***Content*** parameter do not exist at the target resource, such attributes are created with the assigned values. If some attributes in the ***Content*** parameter are set to NULL, such attributes are deleted from the addressed resource.
4. **Delete (D):** an existing ***To*** addressable resource and all its sub-resources are deleted from the Resource storage.
5. **Notify (N):** information to be sent to the Receiver, processing on the Receiver is not indicated by the Originator.
* ***Request Identifier*:** request Identifier (see clause 7.1.7).

 Example usage of request identifier includes enabling the correlation between a Request and one of the many received Responses.

**Operation dependent Parameters:**

* ***Content*:** resource content to be transferred.

 The ***Content*** parameter shall be present in Request for the following operations:

* **Create (C):** ***Content*** is the content of the new resource with the resource type ***ResourceType.***
* **Update (U):** ***Content*** is the content to be replaced in an existing resource. For attributes to be updated at the resource, ***Content*** includes the names of such attributes with their new values. For attributes to be created at the resource, ***Content*** includes names of such attributes with their associated values. For attributes to be deleted at the resource, ***Content*** includes the names of such attributes with their value set to NULL.
* **Notify (N):** ***Content*** is the notification information.

 The ***Content*** parameter may be present in Request for the following operations:

* **Retrieve (R):** ***Content*** is the list of attribute names from the resource that needs to be retrieved. The values associated with the attribute names shall be returned.
* ***Resource Type:*** type of resource.

 The ***ResourceType*** parameter shall be present in Request for the following operations:

* **Create (C):** ***Resource Type*** is the type of the resource to be created.

**Optional Parameters:**

* ***Role IDs:*** optional, required when role based access control is applied. A list of Role-IDs that are allowed by the service subscription shall be provided otherwise the request is considered not valid.

The ***Role*** ***IDs*** parameter shall be used by the Receiver to check the Access Control privileges of the Originator.

* ***Originating Timestamp*:** optional originating timestamp of when the message was built.

 Example usage of the originating timestamp includes: to measure and enable operation (e.g. message logging, correlation, message prioritization/scheduling, accept performance requests, charging, etc.) and to measure performance (distribution and processing latency, closed loop latency, SLAs, analytics, etc.)

* ***Request Expiration Timestamp*:** optional request message expiration timestamp. The Receiver CSE should handle the request before the time expires. If a Receiver CSE receives a request with ***Request Expiration Timestamp*** with the value indicating a time in the past, then the request shall be rejected.

 Example usage of the request expiration timestamp is to indicate when request messages (including delay‑tolerant) should expire and to inform message scheduling/prioritization. When a request with set expiration timestamp demands an operation on a Hosting CSE different than the current Receiver CSE, then the current CSE shall keep trying to deliver the Request to the Hosting CSE until the request expiration timestamp time, in line with provisioned policies.

* ***Result Expiration Timestamp*:** optional result message expiration timestamp. The Receiver CSE should return the result of the request before the time expires.

 Example usage of the result expiration timestamp: An Originator indicates when result messages (including delay-tolerant) should expire and informs message scheduling/prioritization. It can be used to set the maximum allowed total request/result message sequence round trip deadline.

* ***Response Type*:** optional response message type: Indicates what type of response shall be sent to the issued request and when the response shall be sent to the Originator:
* **nonBlockingRequestSynch**: In case the request is accepted by the Receiver CSE, the Receiver CSE responds, after acceptance, with an Acknowledgement confirming that the Receiver CSE will further process the request. The Receiver CSE includes in the response to an accepted request a reference that can be used to access the status of the request and the result of the requested operation at a later time. Processing of Non-Blocking Requests is defined in clause 8.2.2 and in particular for the synchronous case in clause 8.2.2.2.
* **nonBlockingRequestAsynch {optional list of notification targets}:** In case the request is accepted by the Receiver CSE, the Receiver CSE shall respond, after acceptance, with an Acknowledgement confirming that the Receiver CSE will further process the request. The result of the requested operation needs to be sent as notification(s) to the notification target(s) provided optionally within this parameter as a list of entities or to the Originator when no notification target list is provided. When an empty notification target list is provided by the Originator, no notification with the result of the requested operation shall be sent at all. Processing of Non‑Blocking Requests is defined in clause 8.2.2 and in particular for the asynchronous case in clause 8.2.2.3.
* **blockingRequest:** In case the request is accepted by the Receiver CSE, the Receiver CSE responds with the result of the requested operation after completion of the requested operation. Processing of Blocking Requests is defined in clause 8.2.1. This is the default behaviour when the *Response Type* parameter is not given the request.
* **flexBlocking {optional list of notification targets}:** When Response Type in the request received by the Receiver CSE is set to flexBlocking, it means that the Originator of the request has the capability to accept the following types of responses: nonBlockingRequestSynch, nonBlockingRequestAsynch and blockingRequest.

 The Receiver CSE shall make the decision to respond using blocking or non-blocking based on its own local context (memory, processing capability, etc.) if not defined in the resource handling procedure.

 If the Receiver CSE choose to respond using non-blocking mode, based on the presence of notification targets in the request:

* If the notification targets are provided in the request and the Recerver CSE is responding, the Receiver CSE shall notify the result using nonBlockingRequestAsynch.
* If notification targets are not provided, the Receiver CSE shall respond with the address of <request> resource using nonBlockingRequestSynch.

 Example usage of the response type set to *nonBlockingRequestSynch*: An Originator that is optimized to minimize communication time and energy consumption wants to express a Request to the receiver CSE and get an acknowledgement on whether the Request got accepted. After that the Originator may switch into a less power consuming mode and retrieve a Result of the requested Operation at a later time.

 Further example usage of response type set to *nonBlockingRequestSynch:* When the result content is extremely large, or when the result consists of multiple content parts from a target group which are to be aggregated asynchronously over time.

* ***Result Content*:** optional result content: Indicates what are the expected components of the result of the requested operation. The Originator of a request may not need to get back a result of an operation at all. This shall be indicated in the ***Result Content*** parameter. Settings of ***Result Content*** depends on the requested operation specified in ***Operation***. Possible values of ***Result Content*** are:
* **attributes:** Representation of the requested resource shall be returned as content, without the address(es) of the child resource(s) or their descendants. For example, if the request is to retrieve a *<container>* resource, the address(es) of the *<contentInstance>* child-resource(s) is not provided. This setting shall be only valid for Create, Retrieve, Update, Delete operation. When this is used for Create operation, only assigned/modified attributes shall be included in the content. If the Originator does not set ***Result Content*** parameter in the request message, this setting shall be the default value when the Receiver processes the request message.
* **hierarchical-address:** Representation of the address of the created resource. This shall be only valid for a Create operation. The address shall be in hierarchical address scheme.
* **hierarchical-address+attributes:** Representation of the addresss in hierarchical address scheme and assigned/modified attributes of the created resource. This shall be only valid for a Create operation.

- **attributes+child-resources:** Representation of the requested resource, along with a nested representation of all of its child resource(s) , and their descendants, in line with any provided filter criteria as given in the ***Filter Criteria*** parameter shall be returned as content. If there is no filter criteria parameter in the request message then all children/descendants are returned along with their attributes. For example, if the request is to retrieve a *<container>* resource that only has *<contentInstance>* children, the attributes of that *<container>* resource and a representation of all of its *<contentInstance>* child-resource(s) , including their attributes, are provided.

 The originator may request to limit the maximum number of allowed nesting levels. The orginator may also include an offset that indicates the starting point of the direct child resource. The offset shall start at 1. The hosting CSE shall return all direct child resources and their descendants, or up to the maximum nesting level specififed in a request subject to maximum size limit that may be imposed by the hosting CSE. The offset, maximum number/size and maximum level shall be specified in ***Filter Criteria*** as *offset*, *limit*, and *level* condition, respectively, by the Originator.

 The hosting CSE shall list parent resources before their children. This means that the originator of the request will not receive a discovered resource without having received its parents. The hosting CSE shall also ensure that proper nesting representation of all the children is incorporated in its listing for parents and children.

 Nested processing is applicable at every level in the resource tree. If a direct child resource and all its descendants cannot be included in the returned content due to size limitations imposed by the hosting CSE then the direct child resource shall not be included in the response.

 An indication shall be included in the response signalling if the returned content is partial. If the indication is for partial content, the response shall include an offset for the direct child resource where processing can restart for the remaining direct child resources

 This shall be only valid for a Retrieve operation.

* **child-resources:** A nested representation of the resource's child resource(s) their descendants and their attributes shall be returned as content. The resources that are returned are subject to any filter criteria that are given in the ***Filter Criteria*** parameter (if there are no filter criteria then all children and their descendants are returned). The attributes of the parent resource are not returned, but all the attributes of the children are returned. For example, if the request is to retrieve a *<container>* resource that only has *<contentInstance>* children, only a representation of all of its *<contentInstance>* child-resource(s) is provided.

The offset, maximum number/size and maximum level shall be specified in ***Filter Criteria*** as *offset*, *limit*, and *level* condition, respectively, by the Originator. Processing of direct child resources, size limitations, maximum nesting level, and offset for the starting of direct child resource processing of **the attributes+child-resources** option shall apply to this option as well.

This shall be only valid for a Retrieve operation.

* **attributes+child-resource-references :** Representation of the requested resource, along with the address(es) of the child resource(s), and their descendants shall be returned as content. For example, if the request is to retrieve a *<container>* resource, the *<container>* resource and the address(es) of the *<contentInstance>* child-resource(s) are provided.

 The offset, maximum number/size and maximum level shall be specified in ***Filter Criteria*** as *offset*, *limit*, and *level* condition, respectively, by the Originator. Processing of child resources, size limitations, maximum nesting level, and offset for the starting of child resource processing of **the attributes+child-resources** option shall apply to this option as well.

 This shall be only valid for a Retrieve operation.

* **child-resource-references:** Address(es) of the child resources and their descendants, without any representation of the actual requested resource shall be returned as content. For example, if the request is to retrieve a *<container>* resource, only the address(es) of the *<contentInstance>* child-resource(s) is provided.

 The offset, maximum number/size and maximum level shall be specified in ***Filter Criteria*** as *offset*, *limit*, and *level* condition, respectively, by the Originator. Processing of child resources, size limitations, maximum nesting level, and offset for the starting of child resource processing of **the attributes+child-resources** option shall apply to this option as well.

 This shall be only valid for a Retrieve operation.

 This option can be used within the context of resource discovery mechanisms (see clause 10.2.6).

* **nothing:** Nothing shall be returned as operational result content. This setting shall be valid for a Create/Update/Delete/Notify operation.

EXAMPLE: If the request is to delete a resource, this setting indicates that the response shall not include any content.

* **original-resource:** Representation of the original resource pointed by the *link* attribute in the announced resource shall be returned as content, without the address(es) of the child resource(s). This shall be only valid for a Retrieve operation where the ***To*** parameter targets the announced resource.

 Note that for any of the above options, Discovery access control is applied against discovery related procedures, while Retrieve access control procedures is applied against non-discovery related Retrieve operations.

 Note that the fitter criteria usage governs the purpose of a Retrieve operation.

Table 8.1.2-1: Summary of Result Content Values

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Value** | **Create** | **Retrieve** | **Update** | **Delete** | **Notify** |
| attributes | default | default | default | default | n/a |
| hierarchical-address | valid | n/a | n/a | n/a | n/a |
| hierarchical-address+attributes | valid | n/a | n/a | n/a | n/a |
| attributes+child-resources | n/a | valid | n/a | n/a | n/a |
| child-resources | n/a | valid | n/a | n/a | n/a |
| attributes+child-resource-references | n/a | valid | n/a | n/a | n/a |
| child-resource-references | n/a | valid | n/a | n/a | n/a |
| nothing | valid | n/a | valid | valid | valid |
| original-resource | n/a | valid | n/a | n/a | n/a |

* ***Result Persistence*:** optional result persistence: indicates the time for which the response may persist to. The parameter is used in case of non-blocking request where the result attribute of the <request> resource should be kept at the CSE, for example, with the purpose of sharing, tracking and analytics.

 In the case the response of a request is required to be kept in the CSE, for example the procedures of <request> resource, <delivery> resource and <group> resource, the ***Result Persistence*** indicates the time duration for which the CSE keeps the response available after receiving it.

 Example usage of result persistence includes requesting sufficient persistence for analytics to process the response content aggregated asynchronously over time. If a result expiration time is specified then the result persistence lasts beyond the result expiration time.

* ***Operation Execution Time*:** optional operation execution time: indicates the time when the specified operation ***Operation*** is to be executed by the target CSE. A target CSE shall execute the specified operation of a Request having its operational execution time indicator set, starting at the operational execution time. If the execution time has already passed or if the indicator is not set, then the specified operation shall be immediately executed, unless the request expiration time, if set, has been reached.

 Example usage of operational execution time includes asynchronous distribution of flows, which are to be executed synchronously at the operational execution time.

NOTE 6: Time-based flows could not supported depending upon time services available at CSEs.

* ***Event Category*:** optional event category: Indicates the event category that should be used to handle this request. Event categories are impacting how Requests to access remotely hosted resources are processed in the CMDH CSF. Selection and scheduling of connections via CMDH are driven by policies that can differentiate event categories.

 Example usage of "event category" set to specific value X: When the request is demanding an operation to be executed on a Hosting CSE that is different from the current Receiver CSE, the request may be stored in the current Receiver CSE that is currently processing the request on the way to the Hosting CSE until it is allowed by provisioned policies for that event category X to use a communication link to reach the next CSE on a path to the Hosting CSE or until the request expiration timestamp is expired.

 The following values for ***Event Category*** shall have a specified pre-defined meaning:

* ***Event Category* = immediate:** Requests of this category shall be sent as soon as possible and shall not be subject to any further CMDH processing, i.e. the request will not be subject to storing in CMDH buffers when communication over an underlying network is possible. In particular, CMDH processing will respect values for ***Request Expiration Timestamp***, ***Result Expiration Timestamp*** given in the original request and not fill in any default values if they are missing.
* ***Event Category* = bestEffort:** Requests of this category can be stored in CMDH buffers at the discretion of the CSE that is processing the request for an arbitrary time and shall be forwarded via Mcc on a best effort basis. The CSE does not assume any responsibility to meet any time limits for delivering the information to the next CSE. Also the maximum amount of buffered requests for this category is at the discretion of the processing CSE.
* ***Event Category***= **latest:**
* If this category is used in a request asking for a CRUD operation on a resource, the following shall apply:
CRUD requests using this category shall undergo normal CMDH processing as outlined further below in the present document and in oneM2M TS-0004 [3] with a maximum buffer size of one pending request for a specific pair of ***From*** and ***To*** parameters that appear in the request. If a new request message is received by the CSE with a pair of parameters ***From*** and ***To*** that has already been buffered for a pending request, the newer request will replace the buffered older request.
* If this category is used in a notification request triggered by a subscription, the following shall apply:
Notification requests triggered by a subscription using this category shall undergo normal CMDH processing as outlined further below in the present document and in oneM2M TS-0004 [3] with a maximum buffer size of one pending notification request per subscription reference that appears in a notification request. If a new notification request is received by the CSE with a subscription reference that has already been buffered for a pending notification request, the newer request will replace the buffered older request.
* If no further CMDH policies are provisioned for this event category, the forwarding process shall follow the 'bestEffort' rules defined above.

 The M2M Service Provider shall be able to provision CMDH policies describing details for the usage of the specific Underlying Network(s) and the applicable rules as defined in the *[cmdhPolicy]* resource type for other ***Event Category*** values not listed above.

* ***Delivery Aggregation*:** optional delivery aggregation on/off: Use CRUD operations of *<delivery>* resources to express forwarding of one or more original requests to the same target CSE(s). When this parameter is not given in the request, the default behaviour is determined per the provisioned CMDH policy if available. If there is no such CMDH policy, then the default value is "aggregation off".

NOTE 7: Since ***Delivery Aggregation*** is optional, there could be a default value to be used when not present in the Request. This parameter could not be exposed to AEs via Mca.

 Example usage of delivery aggregation set on: The CSE processing a request shall use aggregation of requests to the same target CSE by requesting CREATE of a *<delivery>* resource on the next CSE on the path to the target CSE.

* ***Group Request Identifier*:** optional group request identifier: Identifier optionally added to the group request that is to be fanned out to each member of the group in order to detect loops and avoid duplicated handling of operation in case of loops of group and common members between groups that have parent-child relationship.
* ***Group Request Target Members*:** optional group request target members: Indicates subset of members of a group for which fanout is to be executed. This shall be set by the Originator only when originator wants to execute fanout for failed members of a previous fanout operation.***ilter Criteria*:** optional filter criteria: conditions for filtered retrieve operation are described in table 8.1.2-2. This is used for resource discovery (clause 10.2.6) and general retrieve, update, delete requests (clauses 10.1.2, 10.1.3 and 10.1.4).

 Example usage of retrieve requests with filter criteria using *modifiedSince* condition tag: if a target resource is modified since 12:00 then the Hosting CSE will send a resource representation.

* ***Discovery Result Type:*** Optional Discovery result format. This parameter applies to discovery related requests (see *filterUsage* in table 8.1.2-2 and clause 10.2.6) to indicate the preference of the Originator for the format of returned information in the result of the operation. This parameter shall take on one of the following values reflecting the options in clause 9.3.1:
* *Hierarchical addressing* method.
* *Non-hierarchical addressing* method.

 For example if ***Discovery Result Type*** is set to *Non-hierarchical* addressing method, then the request Originator indicates that the discovered resources should be in the form of *Non-hierarchical* address.

 The absence of the parameter implies that the result shall be in the form of a *Hierarchical* address.

* ***Token Request Indicator:*** Optional parameter used to indicate that the Originator supports the Token Request procedure, and the Originator may attempt the Token Request procedure if the Receiver provides a ***Token Request Information*** parameter in the response.
* ***Tokens:*** Optional parameter used to transport ESData-protected *Tokens* applicable to the request for use in Indirect Dynamic Authorization.
* ***Token IDs:*** Optional parameter used to transport *Token-IDs* applicable to the request for use in Indirect Dynamic Authorization.
* ***Local Token IDs:*** Optional parameter used to transport Local-Token-IDs applicable to the request for use in Indirect Dynamic Authorization.

Table 8.1.2-2: *Filter Criteria* 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. |
| *labels* | 0..n | The *labels* attributes of the resource matches the specified value. |
| *resourceType* | 0..n | The *resourceType* attribute of the resource is the same as the specified value. It also allows differentiating between normal and announced resources. |
| *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. |
| *contentType* | 0..n | The *contentInfo* attribute of the *<contentInstance>* resource matches the specified value. |
| *limit* | 0..1 |  The maximum number of resources to be returned in the response. This may be modified by the Hosting CSE. When it is modified, then the new value shall be smaller than the suggested value by the Originator. |
| *attribute* | 0..n | This is an attribute of resource types (clause 9.6). Therefore, a real tag name is variable and depends on its usage and the value of the attribute can have wild card \*. E.g. *creator* of container resource type can be used as a filter criteria tag as "creator=Sam" , "creator=Sam\*" , "creator=\*Sam" . |
| *filterUsage* | 0..1 | Indicates how the filter criteria is used. If provided, possible values are 'discovery' and 'IPEOnDemandDiscovery'.If this parameter is not provided, the Retrieve operation is a generic retrieve operation and the content of the child resources fitting the filter criteria is returned.If filterUsage is'discovery', the Retrieve operation is for resource discovery (clause 10.2.6), i.e.only the addresses of the child resources are returned.If filterUsage is 'IPEOnDemandDiscovery', the other filter conditions are sent to the IPE as well as the discovery Originator ID. When the IPE successfully generates new resources matching with the conditions, then the resource address(es) shall be returned. This value shall only be valid for the Retrieve request targeting an <AE> resource that represents the IPE. |
| *semanticsFilter* | 0..n | The semantic description contained in one of the <semanticDescriptor> child resources matches the semanticFilter that shall be specified in the SPARQL query language [5]. Examples for matching semantic filters in SPARQL to semantic descriptions can be found in [i.28]. |
| *filterOperation* | 0..1 | Indicates the logical operation (AND/OR) to be used for different condition tags. The default value is logical AND. |
| *contentFilterSyntax* | 0..1 | Indicates the Identifier for syntax to be applied for content-based discovery. |
| *contentFilterQuery* | 0..1 | The query string shall be specified when *contentFilterSyntax* parameter is present. |
| *level* | 0..1 | The maximum level of resource tree that the Hosting CSE shall perform the operation starting from the target resource (i.e. ***To*** parameter). This shall only be appiled for Retrieve operation. The level of the target resource itself is zero and the level of the direct children of the target is one. |
| *offset* | 0..1 | The number of direct child and descendant resources that a Hosting CSE shall skip over and not include within a Retrieve response when processing a Retrieve request to a targeted resource.  |

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

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

 e.g. *createdBefore* = "time1" AND *unmodifiedSince* = "time2" if *filterOperation* = "AND" or "NULL", or *createdBefore* = "time1" OR *unmodifiedSince* = "time2" if *filterOperation* = "OR".

* Same condition tags shall use the "OR" logical operation, i.e. *filterOperation* doesn't apply to same conditions.

No mixed AND/OR filter operation will be supported.

Once the Request is delivered, the Receiver shall analyze the Request to determine the target resource.

If the target resource is addressing another M2M Node, the Receiver shall route the request appropriately.

If the target resource is addressing the Receiver, it shall:

* Check the existence of***To*** addressed resource.
* Identify the resource type by ***Resource Type***.
* Check the privileges for ***From*** Originator to perform the requested operation.
* Perform the requested operation (using ***Content*** content when provided) according to the provided request parameters as described above.
* Depending on the request result content, respond to the Originator with indication of successful or unsuccessful operation results. In some specific cases (e.g. limitation in the binding protocol or based on application indications), the Response could be avoided.

Table 8.1.2-3 summarizes the parameters specified in this clause for the Request message, showing any differences as applied to C, R, U, D or N operations. "M" indicates mandatory, "O" indicates optional, "N/A" indicates "not applicable".

Table 8.1.2-3: Summary of Request Message Parameters

| Request message parameter | Operation |
| --- | --- |
| Create | Retrieve | Update | Delete | Notify |
| ***Mandatory*** | ***Operation*** - operation to be executed | M | M | M | M | M |
| ***To*** - the address of the target resource on the target CSE | M | M | M | M | M |
| ***From*** - the identifier of the message Originator | OSee note | M | M | M | M |
| ***Request Identifier*** - uniquely identifies a Request message | M | M | M | M | M |
| ***Operation dependent*** | ***Content*** - to be transferred | M | O | M | N/A | M |
| ***Resource Type*** - of resource to be created | M | N/A | N/A | N/A | N/A |
| ***Optional*** | ***Originating Timestamp*** - when the message was built | O | O | O | O | O |
| ***Request Expiration Timestamp*** - when the request message expires | O | O | O | O | O |
| ***Result Expiration Timestamp*** - when the result message expires | O | O | O | O | O |
| ***Operational Execution Time*** - the time when the specified operation is to be executed by the target CSE | O | O | O | O | O |
| ***Response Type*** - type of response that shall be sent to the Originator | O | O | O | O | O |
| ***Result Persistence*** - the duration for which the reference containing the responses is to persist | O | O | O | O | N/A |
| ***Result Content*** - the expected components of the result | O | O | O | O | N/A |
| ***Event Category*** - indicates how and when the system should deliver the message | O | O | O | O | O |
| ***Delivery Aggregation*** - aggregation of requests to the same target CSE is to be used | O | O | O | O | O |
| ***Group Request Identifier*** - Identifier added to the group request that is to be fanned out to each member of the group | O | O | O | O | O |
| ***Group Request Target Members-***indicates subset of members of a group | O | O | O | O | N/A |
| ***Filter Criteria*** - conditions for filtered retrieve operation | N/A | O | O | O | N/A |
| ***Discovery Result Type*** - format of information returned for Discovery operation | N/A | O | N/A | N/A | N/A |
| ***Token Request Indicator*** - indicating that the Originator may attempt Token Request procedure (for Dynamic Authorization) if initiated by the Receiver | O | O | O | O | O |
| ***Tokens*** - for use in dynamic authorization | O | O | O | O | O |
| ***Token IDs*** - for use in dynamic authorization | O | O | O | O | O |
| ***Role IDs*** - for use in role based access control | O | O | O | O | O |
| ***Local Token IDs*** - for use in dynamic authorization | O | O | O | O | O |
| NOTE: *From* parameter is optional in case of an AE CREATE request and mandatory for all other requests. |

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

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 only include a proposal to change only 3 tables. Includes any changes to references, definitions, and acronyms in the same deliverable?
* 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 sections need not show surrounding clauses as long as the proposed section number clearly shows where the new section 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.?