**3GPP TSG-CT WG4 Meeting #101eC4-205**

**E-Meeting, 3rd – 13th November 2020 was C4-205302**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *CR-Form-v12.0* | | | | | | | | |
| **CHANGE REQUEST** | | | | | | | | |
|  | | | | | | | | |
|  | **29.598** | **CR** | **0019** | **rev** | **1** | **Current version:** | **16.2.0** |  |
|  | | | | | | | | |
| *For* [*HE**LP*](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* | | | | | | | | |
|  | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME |  | Radio Access Network |  | Core Network | **X** |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | |
| ***Title:*** | Meta Schema | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Source to WG:*** | Nokia, Nokia Shanghai Bell | | | | | | | | | |
| ***Source to TSG:*** | CT4 | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Work item code:*** | SBIProtoc17 | | | | |  | ***Date:*** | | | 2020-11-03 |
|  |  | | | |  | |  | | |  |
| ***Category:*** | **B** |  | | | | | ***Release:*** | | | Rel-17 |
|  | *Use one of the following categories:* ***F*** *(correction)* ***A*** *(mirror corresponding to a change in an earlier release)* ***B*** *(addition of feature),* ***C*** *(functional modification of feature)* ***D*** *(editorial modification)*  Detailed explanations of the above categories can be found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | | | | | | | | *Use one of the following releases: Rel-8 (Release 8) Rel-9 (Release 9) Rel-10 (Release 10) Rel-11 (Release 11) Rel-12 (Release 12)* *Rel-13 (Release 13) Rel-14 (Release 14) Rel-15 (Release 15) Rel-16 (Release 16)* | |
|  |  | | | | | | | | | |
| ***Reason for change:*** | | When the consumer NF of the UDSF sends a request to the UDSF to search for records that match specific search criteria, this can lead to heavy processing load in the UDSF if a very large number of records need to be traversed. Indexing based on meta data tags can be used to make searches more efficient, however, indexes also make record write and update procedures more expensive. This cost can be reduced if the intended use of tags from a record's meta data is known by the UDSF based on standardized categories declared by the NF consumer.  It is therefore porposed to introduce a new optional feature "Meta Schema". When supported by the UDSF and the NF consumer, the NF consumer can store meta schemas in the UDSF and link a reocord's meta to a stored meta schema. The linked meta schema indicates for each tag how it is intended to be used, e.g. as unique search key, search key, count key or other key. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Summary of change:*** | | Add a new feature "Meta Schema"  Add new procedures to create, update and delete meta schemas in the UDSF.  Define resources and methods for meta schema access.  Define data types for meta schema.  Update type RecordMeta so it can be linked to a meta schema. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Consequences if not approved:*** | | Search operations in the UDSF are not optimized and can lead to heavy processing load. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Clauses affected:*** | | 5.2.2.2.1, 5.2.2.2.x (new), 5.2.2.3.1, 5.2.2.3.y (new), 5.2.2.4.1, 5.2.2.4.z (new), 5.2.2.5.1, 5.2.2.5.w (new), 6.1.3.1, 6.1.3.x (new), 6.1.6.1, 6.1.6.2.3, 6.1.6.2.8, 6.1.6.2.y1 (new), 6.1.6.2.y2 (new), 6.1.6.3.2, 6.1.6.3.y3 (new), 6.1.7.3, 6.1.8, A.2 | | | | | | | | |
|  | |  | | | | | | | | |
|  | | **Y** | **N** |  | | | |  | | |
| ***Other specs*** | |  | **X** | Other core specifications | | | | TS/TR ... CR ... | | |
| ***affected:*** | |  | **X** | Test specifications | | | | TS/TR ... CR ... | | |
| ***(show related CRs)*** | |  | **X** | O&M Specifications | | | | TS/TR ... CR ... | | |
|  | |  | | | | | | | | |
| ***Other comments:*** | | This CR introduces a backwards compatible new feature to the Nudsf\_DataRepository API. | | | | | | | | |
|  | |  | | | | | | | | |
| ***This CR's revision history:*** | |  | | | | | | | | |

\* \* \* First Change \* \* \* \*

##### 5.2.2.2.1 General

The following procedures using the Query service operation are supported:

- Record Retrieval

- Meta Retrieval

- Blocks Retrieval

- Block Retrieval

- Search

- Subscriptions Retrieval

- Individual Subscription Retrieval

- Meta Schema Retrieval

\* \* \* Next Change \* \* \* \*

##### 5.2.2.2.x Meta Schema Retrieval

Figure 5.2.2.2.x-1 shows a scenario where the NF service consumer sends a request to the UDSF to retrieve a meta schema that matches the provided schemaId and optionally includes the query parameter supported-features.



Figure 5.2.2.2.x-1: Requesting a Meta Schema

1. The NF service consumer (any NF) sends a GET request to the resource indicated by schemaId.

2a. On success, the UDSF responds with "200 OK" with the message body containing the meta schema.

2b. If the meta schema for the given schemaId does not exist in the UDSF, the HTTP status code "404 Not Found" shall be returned optionally including additional error information in the response body (in the ProblemDetails element).

On failure, the appropriate HTTP status code indicating the error shall be returned and appropriate additional error information should be returned in the GET response body.

\* \* \* Next Change \* \* \* \*

##### 5.2.2.3.1 General

The following procedures using the Create service operation are supported:

- Record Create

- Block Create

- Meta Schema Create

\* \* \* Next Change \* \* \* \*

##### 5.2.2.3.y Meta Schema Create

Figure 5.2.2.3.y-1 shows a scenario where the NF service consumer sends a request to the UDSF to create a meta schema with the provided schemaId.

The request contains the schemaId and optionally the query parameters supported-features and get-previous.



Figure 5.2.2.3.y-1: Create a Meta Schema

1. The NF service consumer (any NF) sends a PUT request to create the resource indicated by schemaId. The request body contains the meta schema.

2a. On success, "201 Created" shall be returned, the payload body of the PUT response should contain the representation of the created resource, and the "Location" header shall be present and shall contain the URI of the created resource.

On failure, the appropriate HTTP status code indicating the error shall be returned and appropriate additional error information should be returned in the PUT response body.

\* \* \* Next Change \* \* \* \*

##### 5.2.2.4.1 General

The following procedures using the Update service operation are supported:

- Record Update

- Block Update

- Meta Update

- Subscription Notification Update

- Meta Schema Update

\* \* \* Next Change \* \* \* \*

##### 5.2.2.4.z Meta Schema Update

Figure 5.2.2.4.z-1 shows a scenario where the NF service consumer sends a request to the UDSF to update a meta schema with the provided schemaId.

The request contains the schemaId, the complete new meta schema and optionally the query parameters supported-features and get-previous.



Figure 5.2.2.4.z-1: Update a meta schema

1. The NF service consumer shall send a PUT request to the resource representing the record that is to be updated, the request body shall include the complete new meta schema.

2a. On success, the UDSF shall respond with "204 No Content" if no meta schema is returned, i.e. the get-previous query parameter was not included in the request.

2b. On success, the UDSF shall respond with "200 OK" if a meta schema is returned, i.e. the get-previous query parameter was included in the request.

2c. On failure, the UDSF shall respond with "404 Not Found" if realm or storage does not exist.

2d On failure, the UDSF shall respond with "501 Not Implemented" if meta schema update is not implemented by the UDSF.

On failure, the appropriate HTTP status code indicating the error shall be returned and appropriate additional error information should be returned in the PUT response body.

\* \* \* Next Change \* \* \* \*

##### 5.2.2.5.1 General

The following procedures using the Delete service operation are supported:

- Record Delete

- Block Delete

- Meta Schema Delete

\* \* \* Next Change \* \* \* \*

##### 5.2.2.5.w Meta Schema Delete

Figure 5.2.2.5.w-1 shows a scenario where the NF service consumer sends a request to the UDSF to Delete a meta schema with the provided schemaId.

The request contains the schemaId and optionally the query parameters supported-features and get-previous.



Figure 5.2.2.5.w-1: Delete a meta schema

1. The NF service consumer shall send a DELETE request to the resource representing the meta schema.

2a. On success, the UDSF shall respond with "204 No Content" if no meta schema is returned, i.e. the get-previous query parameter was not included in the request.

2b. On success, the UDSF shall respond with "200 OK" if a meta schema is returned, i.e. the get-previous query parameter was included in the request.

2c. On failure, the UDSF shall respond with "404 Not Found" if the meta schema does not exist or "403 Forbidden" if the meta schema is still referenced by existing records, and may include the ProblemDetails.

On failure, the appropriate HTTP status code indicating the error shall be returned and appropriate additional error information should be returned in the DELETE response body.

\* \* \* Next Change \* \* \* \*

#### 6.1.3.1 Overview



Figure 6.1.3.1-1: Resource URI structure of the nudsf-dr API

Table 6.1.3.1-1 provides an overview of the resources and applicable HTTP methods.

Table 6.1.3.1-1: Resources and methods overview

|  |  |  |  |
| --- | --- | --- | --- |
| Resource name | Resource URI | HTTP method or custom operation | Description |
| RecordCollection  (Collection) | /{realmId}/{storageId}/records | GET | Search for records |
| Record  (Document) | /{realmId}/{storageId}/records/{recordId} | GET | Retrieve a record |
| PUT | Create or update a record |
| DELETE | Delete a record |
| Meta  (Document) | /{realmId}/{storageId}/records/{recordId}/meta | GET | Retrieve the meta of a record |
| PATCH | Modify the meta of a record |
| BlockCollection  (Collection) | /{realmId}/{storageId}/records/{recordId}/blocks | GET | Retrieve all the blocks of a record |
| Block  (Document) | /{realmId}/{storageId}/records/{recordId}/blocks/{blockId} | GET | Retrieve a block |
| PUT | Create or update a block |
| DELETE | Delete a block |
| NotificationSubscriptions (Collection) | /{realmId}/{storageId}/subs-to-notify | GET | Retrieve existing subscriptions |
| Individual NotificationSubscription (Document) | /{realmId}/{storageId}/subs-to-notify/{subscriptionId} | DELETE | Delete the subscription identified by {subscriptionId}, i.e. unsubscribe to notification for change of data |
| PATCH | Update an individual Subscription to notification |
| PUT | Create or update a subscription to notification, |
| GET | Retrieve an individual Subscription to notification |
| Meta Schema  (Document) | /{realmId}/{storageId}/meta-schemas/{schemaId} | GET | Retrieve a Meta Schema |
| PUT | Create or update a Meta Schema |
| DELETE | Delete a Meta Schema |

\* \* \* Next Change \* \* \* \*

#### 6.1.3.x Resource: Meta Schema (Document)

##### 6.1.3.x.1 Description

This resource represents a meta schema within a storage.

##### 6.1.3.x.2 Resource Definition

Resource URI: **{apiRoot}/nudsf-dr/v1/{realmId}/{storageId}/meta-schemas/{schemaId}**

This resource shall support the resource URI variables defined in table 6.1.3.x.2-1.

Table 6.1.3.x.2-1: Resource URI variables for this resource

|  |  |
| --- | --- |
| Name | Definition |
| {apiRoot} | See clause 6.1.1 |
| {realmId} | Represents the realm Id where the schema is stored |
| {storageId} | Represents the storage Id where the schema is stored |
| {schemaId} | Represents the schema Id of the schema |

##### 6.1.3.x.3 Resource Standard Methods

###### 6.1.3.x.3.1 GET

This method shall support the URI query parameters specified in table 6.1.3.x.3.1-1.

Table 6.1.3.x.3.1-1: URI query parameters supported by the GET method on this resource

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Data type | P | Cardinality | Description | Applicability |
| supported-features | SupportedFeatures | O | 0..1 | see 3GPP TS 29.500 [4] clause 6.6. | Meta Schema |

This method shall support the request data structures specified in table 6.1.3.x.3.1-2 and the response data structures and response codes specified in table 6.1.3.x.3.1-3.

Table 6.1.3.x.3.1-2: Data structures supported by the GET Request Body on this resource

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | P | Cardinality | Description |
| n/a |  |  |  |

Table 6.1.3.x.3.1-3: Data structures supported by the GET Response Body on this resource

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data type | P | Cardinality | Response  codes | Description |
| MetaSchema | M | 1 | 200 OK | A response body containing the meta schema. |
| ProblemDetails | O | 0..1 | 404 Not Found | The "cause" attribute may be used to indicate one of the following application errors:  -REALM\_NOT\_FOUND  -STORAGE\_NOT\_FOUND  -SCHEMA\_NOT\_FOUND |
| NOTE: The mandatory HTTP error status code for the GET method listed in Table 5.2.7.1-1 of 3GPP TS 29.500 [4] also apply. | | | | |

###### 6.1.3.x.3.2 PUT

This method shall support the URI query parameters specified in table 6.1.3.x.3.2-1.

Table 6.1.3.x.3.2-1: URI query parameters supported by the PUT method on this resource

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Data type | P | Cardinality | Description | Applicability |
| supported-features | SupportedFeatures | O | 0..1 | see 3GPP TS 29.500 [4] clause 6.6 | Meta Schema |
| get-previous | boolean | O | 0..1 | Request to return the previous meta schema content if the meta schema already exists in the targeted storage for the same schema identifier. | Meta Schema |

This method shall support the request data structures specified in table 6.1.3.x.3.2-2 and the response data structures and response codes specified in table 6.1.3.x.3.2-3.

Table 6.1.3.x.3.2-2: Data structures supported by the PUT Request Body on this resource

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | P | Cardinality | Description |
| MetaSchema | M | 1 | The schema that is to be created/updated. |

Table 6.1.3.x.3.2-3: Data structures supported by the PUT Response Body on this resource

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data type | P | Cardinality | Response  codes | Description |
| MetaSchema | M | 1 | 200 OK | Upon successful update of a schema, a response body containing the previous schema value (if get-previous was indicated in the request, and if one exists) will be returned |
| MetaSchema | M | 1 | 201 Created | Upon successful creation of a schema, a response body of the created schema schall be returned.  The HTTP response shall include a "Location" HTTP header that contains the resource URI of the created resource. |
| n/a |  |  | 204 No Content | Upon successful update of a schema, an empty response is returned if no previous schema value was requested. |
| ProblemDetails | O | 0..1 | 404 Not Found | The "cause" attribute may be used to indicate one of the following application errors:  -REALM\_NOT\_FOUND  -STORAGE\_NOT\_FOUND |
| NOTE: The mandatory HTTP error status code for the PUT method listed in Table 5.2.7.1-1 of 3GPP TS 29.500 [4] also apply. | | | | |

Table 6.1.3.x.3.2-4: Headers supported by the 201 Response Code on this resource

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Data type | P | Cardinality | Description |
| Location | string | M | 1 | Contains the URI of the newly created resource, according to the structure: {apiRoot}/nudsf-dr/<apiVersion>/{realmId}/{storageId}/meta-schemas/{schemaId} |

###### 6.1.3.x.3.3 DELETE

This method shall support the URI query parameters specified in table 6.1.3.x.3.3-1.

Table 6.1.3.x.3.3-1: URI query parameters supported by the DELETE method on this resource

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Data type | P | Cardinality | Description | Applicability |
| supported-features | SupportedFeatures | O | 0..1 | see 3GPP TS 29.500 [4] clause 6.6 | Meta Schema |
| get-previous | boolean | O | 0..1 | Request to return the schema content if a schema exists in the targeted storage for the same schema identifier. | Meta Schema |

This method shall support the request data structures specified in table 6.1.3.x.3.3-2 and the response data structures and response codes specified in table 6.1.3.x.3.3-3.

Table 6.1.3.x.3.3-2: Data structures supported by the DELETE Request Body on this resource

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | P | Cardinality | Description |
| n/a |  |  |  |

Table 6.1.3.x.3.3-3: Data structures supported by the DELETE Response Body on this resource

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data type | P | Cardinality | Response  codes | Description |
| MetaSchema | O | 0..1 | 200 OK | Upon success, a response body containing the schema value, if get-previous was indicated in the request. |
| n/a |  |  | 204 No Content | Upon success, if get-previous was not indicated in the request.. |
| ProblemDetails | O | 0..1 | 404 Not Found | The "cause" attribute may be used to indicate one of the following application errors:  -REALM\_NOT\_FOUND  -STORAGE\_NOT\_FOUND  -SCHEMA\_NOT\_FOUND |
| NOTE: The mandatory HTTP error status code for the DELETE method listed in Table 5.2.7.1-1 of 3GPP TS 29.500 [4] also apply. | | | | |

\* \* \* Next Change \* \* \* \*

#### 6.1.6.1 General

This clause specifies the application data model supported by the API.

Table 6.1.6.1-1 specifies the data types defined for the Nudsf service based interface protocol. For simple data types defined for the Nudsf\_DataRepository service API see table 6.1.6.3.2-1.

Table 6.1.6.1-1: Nudsf specific Data Types

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | Clause defined | Description | Applicability |
| RecordSearchResult | 6.1.6.2.2 | Record Search Result |  |
| RecordMeta | 6.1.6.2.3 | Record Meta |  |
| RecordBody | 6.1.6.2.4 | Record Body |  |
| Record | 6.1.6.2.5 | Record |  |
| BlockBody | 6.1.6.2.6 | Block Body |  |
| Block | 6.1.6.2.7 | Block |  |
| SearchCondition | 6.1.6.2.8 | Search Condition |  |
| SearchComparison | 6.1.6.2.9 | Search Comparison |  |
| ComparisonOperator | 6.1.6.3.3 | Comparison Operator |  |
| ConditionOperator | 6.1.6.3.4 | Condition Operator |  |
| SearchExpression | 6.1.6.4.1 | Search Expression |  |
| NotificationSubscription | 6.1.6.2.10 | Notification Subscription |  |
| RecordNotification | 6.1.6.2.11 | Record Notification |  |
| NotificationDescription | 6.1.6.2.12 | Notification Description |  |
| SubscriptionFilter | 6.1.6.2.13 | Subscription Filter |  |
| ClientId | 6.1.6.2.14 | Client Identity |  |
| MetaSchema | 6.1.6.2.y1 | Meta Schema | Meta Schema |
| TagType | 6.1.6.2.y2 | Tag Type | Meta Schema |
| RecordOperation | 6.1.6.3.15 | Record Operation |  |
| SchemaId | 6.1.6.3.2 | Identifier of a Meta Schema | Meta Schema |
| KeyType | 6.1.6.3.y3 | Key Type | Meta Schema |

Table 6.1.6.1-2 specifies data types re-used by the Nudsf service based interface protocol from other specifications, including a reference to their respective specifications and when needed, a short description of their use within the Nudsf service based interface.

Table 6.1.6.1-2: Nudsf re-used Data Types

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | Reference | Comments | Applicability |
| SupportedFeatures | 3GPP TS 29.571 [19] | see 3GPP TS 29.500 [4] clause 6.6. |  |
| PatchItem | 3GPP TS 29.571 [19] | Data structure used for JSON patch. |  |
| PatchResult | 3GPP TS 29.571 [19] |  |  |
| Uri | 3GPP TS 29.571 [19] |  |  |
| DateTime | 3GPP TS 29.571 [19] |  |  |
| NfInstanceId | 3GPP TS 29.571 [19] |  |  |
| NfSetId | 3GPP TS 29.571 [19] |  |  |

\* \* \* Next Change \* \* \* \*

##### 6.1.6.2.3 Type: RecordMeta

Table 6.1.6.2.3-1: Definition of type RecordMeta

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | Data type | P | Cardinality | Description | Applicability |
| tags | map(array(string)) | O | 1..N | A map of tag name/values pairs, where the tag name is a unique string name that is the primary key of the map and is paired with an array of string values. |  |
| ttl | DateTime | O | 0..1 | ttl refers to the lifetime of the record. After the expiry, the record shall be deleted. |  |
| callbackReference | Uri | O | 0..1 | The Uri where the NF Service Consumer shall receive notification on the expiry of the Record as indicated by the ttl attribute if desired. |  |
| schemaId | SchemaId | O | 0..1 | Id of the MetaSchema to which the tags comply to | Meta Schema |

\* \* \* Next Change \* \* \* \*

##### 6.1.6.2.8 Type: SearchCondition

Table 6.1.6.2.8-1: Definition of type SearchCondition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | Data type | P | Cardinality | Description | Applicability |
| cond | ConditionOperator | M | 1 | Logical operator ("AND", "OR" or "NOT") |  |
| units | array(SearchExpression) | M | 1..N | For the logical "NOT" operator indicated in the cond attribute, only one member shall be present in the array.  For the logical "AND" or "OR" operators indicated in the cond attribute, at least two members shall be present in the array and all the members in the array shall be interpreted as logically concatenated with the logical operator. |  |
| schemaId | SchemaId | O | 0..1 | When included, the search is limited to records which have the given schemaId value stored in their RecordMeta | Meta Schema |

\* \* \* Next Change \* \* \* \*

##### 6.1.6.2.y1 Type: MetaSchema

Table 6.1.6.2.y1-1: Definition of type MetaSchema

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | Data type | P | Cardinality | Description | Applicability |
| schemaId | SchemaId | M | 1 | Id of the schema. Used as reference. | Meta Schema |
| metaTags | array(TagType) | M | 1..N | Array of tag types that describe the schema | Meta Schema |

\* \* \* Next Change \* \* \* \*

##### 6.1.6.2.y2 Type: TagType

Table 6.1.6.2.y2-1: Definition of type TagType

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | Data type | P | Cardinality | Description | Applicability |
| tagName | string | M | 1 | Name of the tag. Used as reference. Refers to the unique tag string name that is the primary key of the map and is paired with an array of string values in the RecordMeta tags IE | Meta Schema |
| keyType | KeyType | M | 1 | Type of key | Meta Schema |
| sort | boolean | C | 0..1 | Shall be present if keyType is "SEARCH\_KEY" or "SEARCH\_AND\_COUNT\_KEY"  true: indicates that searches based on "GT", "GTE", "LT" and / or "LTE" are expected. false: indicates otherwise | Meta Schema |
| presence | boolean | O | 0..1 | true: indicates that presence of the tag is mandatory in the RecordMeta false (default): indicates that presence of the tag is optional in the RecordMeta | Meta Schema |

\* \* \* Next Change \* \* \* \*

##### 6.1.6.3.2 Simple data types

The simple data types defined in table 6.1.6.3.2-1 shall be supported.

Table 6.1.6.3.2-1: Simple data types

|  |  |  |  |
| --- | --- | --- | --- |
| Type Name | Type Definition | Description | Applicability |
| SchemaId | string | Id of the schema. Used as reference. | Meta Schema |

\* \* \* Next Change \* \* \* \*

##### 6.1.6.3.y3 Enumeration: KeyType

Table 6.1.6.3.y3-1: Enumeration KeyType

|  |  |  |
| --- | --- | --- |
| Enumeration value | Description | Applicability |
| "UNIQUE\_KEY" | Tags with this key type may be used for search and will result in no or one matching records. | Meta Schema |
| "SEARCH\_KEY" | Tags with this key type may be used for search and will result in no, one or more matching records. | Meta Schema |
| "COUNT\_KEY" | Tags with this key type may be used for count. | Meta Schema |
| "SEARCH\_AND\_COUNT\_KEY" | Tags with this key type may be used for seach and count. | Meta Schema |
| "OTHER\_TAG" | Tags with this key type may not be used for search or count. | Meta Schema |

\* \* \* Next Change \* \* \* \*

#### 6.1.7.3 Application Errors

The application errors defined for the Nudsf\_DataRepository service are listed in Table 6.1.7.3-1.

Table 6.1.7.3-1: Application errors

|  |  |  |
| --- | --- | --- |
| Application Error | HTTP status code | Description |
| TTL\_VALUE\_NOT\_ALLOWED | 403 Forbidden | The ttl value indicated in the request exceeds the maximum value allowed in the UDSF. |
| REALM\_NOT\_FOUND | 404 Not Found | The realm indicated in the HTTP/2 request is unavailable in the UDSF. |
| STORAGE\_NOT\_FOUND | 404 Not Found | The storage indicated in the HTTP/2 request is unavailable in the UDSF. |
| RECORD\_NOT\_FOUND | 404 Not Found | The record indicated in the HTTP/2 request is unavailable in the UDSF. |
| BLOCK\_NOT\_FOUND | 404 Not Found | The block indicated in the HTTP/2 request is unavailable in the UDSF. |
| SUBSCRIPTION\_NOT\_FOUND | 404 Not Found | The subscription indicated in the HTTP/2 request is unavailable in the UDSF. |
| SUBSCRIPTION\_EXISTS | 409 Conflict | The subscription indicated in the HTTP/2 request already exists in the UDSF. |
| SCHEMA\_NOT\_FOUND | 404 Not Found | The schema indicated in the HTTP/2 request is unavailable in the UDSF. |
| SCHEMA\_IN\_USE | 403 Forbidden | The schema cannot be deleted as it is still referenced by existing records. |

\* \* \* Next Change \* \* \* \*

## 6.1.8 Feature negotiation

The optional features in table 6.1.8-1 are defined for the Nudsf\_DataRepository API. They shall be negotiated using the extensibility mechanism defined in clause 6.6 of 3GPP TS 29.500 [4].

Table 6.1.8-1: Supported Features

|  |  |  |
| --- | --- | --- |
| Feature number | Feature Name | Description |
| 1 | AdvancedQuery | If an NF consumer detects that the UDSF supports the AdvancedQuery feature, it may use values of the ComparisonOperator besides "EQ" and may also use the cond attribute of the SearchCondition.  If an NF consumer detects that the UDSF does not support the AdvancedQuery feature, it shall only use a value of "EQ" of the ComparisonOperator and shall not use the cond attribute of the SearchCondition. |
| z | Meta Schema | This feature supports optimization of UDSF data storage and allows the UDSF to know in advance how data search access by the NF consumer is expected.  If the NF consumer detects that the UDSF does not support the Meta Schema feature, it shall not make use of the procedures for storing, updating and deleting Meta Schemas. |

\* \* \* Next Change \* \* \* \*

## A.2 Nudsf\_DataRepository API

openapi: 3.0.0

\*\*\*\*\*\*\*\*\*\*\*text not shown for clarity\*\*\*\*\*\*\*\*\*\*\*\*

'/{realmId}/{storageId}/meta-schemas/{schemaId}':

summary: Access to a specific Meta Schema, identified by its SchemaId

description: >-

Access to a specific Meta Schema

get:

summary: Meta Schema access

description: retrieve one specific Meta Schema

operationId: GetMetaSchema

tags:

- Meta Schema CRUD

parameters:

- name: realmId

in: path

description: Identifier of the Realm

required: true

schema:

type: string

example: Realm01

- name: storageId

in: path

description: Identifier of the Storage

required: true

schema:

type: string

example: Storage01

- name: schemaId

in: path

description: Identifier of the Meta Schema

required: true

schema:

$ref: '#/components/schemas/SchemaId'

example: 'UserSchemaValue000000001'

- name: If-None-Match

in: header

description: Validator for conditional requests, as described in RFC 7232, 3.2

schema:

type: string

- name: If-Modified-Since

in: header

description: Validator for conditional requests, as described in RFC 7232, 3.3

schema:

type: string

- name: supported-features

in: query

description: Features required to be supported by the target NF

schema:

$ref: 'TS29571\_CommonData.yaml#/components/schemas/SupportedFeatures'

responses:

'200' : #result ok

$ref: '#/components/responses/RecordBody'

'304':

$ref: '#/components/responses/304'

'400':

$ref: 'TS29571\_CommonData.yaml#/components/responses/400'

'401':

$ref: 'TS29571\_CommonData.yaml#/components/responses/401'

'403':

$ref: 'TS29571\_CommonData.yaml#/components/responses/403'

'404':

$ref: 'TS29571\_CommonData.yaml#/components/responses/404'

'500':

$ref: 'TS29571\_CommonData.yaml#/components/responses/500'

'503':

$ref: 'TS29571\_CommonData.yaml#/components/responses/503'

default:

$ref: 'TS29571\_CommonData.yaml#/components/responses/default'

put:

summary: Create/Modify Meta Schema

description: Create or Modify a Meta Schema with a user provided SchemaId

operationId: CreateOrModifyMetaSchema

tags:

- Meta Schema CRUD

parameters:

- name: realmId

in: path

description: Identifier(name) of the Realm

required: true

schema:

type: string

example: Realm01

- name: storageId

in: path

description: Identifier of the Storage

required: true

schema:

type: string

example: Storage01

- name: schemaId

in: path

description: Identifier of the Meta Schema

required: true

schema:

$ref: '#/components/schemas/SchemaId'

example: 'UserSchemaValue000000001'

- name: If-None-Match

in: header

description: Validator for conditional requests, as described in RFC 7232, 3.2

schema:

type: string

- name: If-Match

in: header

description: Validator for conditional requests, as described in RFC 7232, 3.2

schema:

type: string

- name: get-previous

in: query

description: Retrieve the Meta Schema before update

required: false

schema:

type: boolean

default: false

- name: supported-features

in: query

description: Features required to be supported by the target NF

schema:

$ref: 'TS29571\_CommonData.yaml#/components/schemas/SupportedFeatures'

requestBody:

content:

application/json:

schema:

$ref: '#/components/schemas/MetaSchema'

required: true

responses:

'200':

description: Update with return

content:

application/json:

schema:

$ref: '#/components/schemas/MetaSchema'

'201':

description: >-

Create case. The resource has been successfully created, location header indicates

the URI of the created Record.

$ref: '#/components/responses/RecordBody'

headers:

Location:

$ref: '#/components/headers/Location'

Cache-Control:

$ref: '#/components/headers/Cache-Control'

ETag:

$ref: '#/components/headers/ETag'

Last-Modified:

$ref: '#/components/headers/Last-Modified'

'204': # Update without return

description: >-

Update case. The resource has been successfully updated and no

additional content is included in the response message.

headers:

Cache-Control:

$ref: '#/components/headers/Cache-Control'

ETag:

$ref: '#/components/headers/ETag'

Last-Modified:

$ref: '#/components/headers/Last-Modified'

'304':

$ref: '#/components/responses/304'

'400':

$ref: 'TS29571\_CommonData.yaml#/components/responses/400'

'401':

$ref: 'TS29571\_CommonData.yaml#/components/responses/401'

'403':

$ref: 'TS29571\_CommonData.yaml#/components/responses/403'

'404':

$ref: 'TS29571\_CommonData.yaml#/components/responses/404'

'408':

$ref: 'TS29571\_CommonData.yaml#/components/responses/408'

'412':

description: Return Meta Schema value if get-previous=true

content:

application/json:

schema:

$ref: '#/components/schemas/MetaSchema'

'413':

$ref: 'TS29571\_CommonData.yaml#/components/responses/413'

'500':

$ref: 'TS29571\_CommonData.yaml#/components/responses/500'

'501':

$ref: 'TS29571\_CommonData.yaml#/components/responses/501'

'503':

$ref: 'TS29571\_CommonData.yaml#/components/responses/503'

default:

$ref: 'TS29571\_CommonData.yaml#/components/responses/default'

delete:

summary: Delete a Meta Schema with an user provided SchemaId

operationId: DeleteMetaSchema

tags:

- MetaSchema CRUD

parameters:

- name: realmId

in: path

description: Identifier(name) of the Realm

required: true

schema:

type: string

example: Realm01

- name: storageId

in: path

description: Identifier of the Storage

required: true

schema:

type: string

example: Storage01

- name: schemaId

in: path

description: Identifier of the Meta Schema

required: true

schema:

$ref: '#/components/schemas/SchemaId'

example: 'UserSchemaValue000000001'

- name: If-Match

in: header

description: Record validator for conditional requests, as described in RFC 7232, 3.2

schema:

type: string

- name: get-previous

in: query

description: Retrieve the Meta Schema before delete

required: false

schema:

type: boolean

default: false

- name: supported-features

in: query

description: Features required to be supported by the target NF

schema:

$ref: 'TS29571\_CommonData.yaml#/components/schemas/SupportedFeatures'

responses:

'200':

description: OK

content:

application/json:

schema:

$ref: '#/components/schemas/MetaSchema'

'204':

description: Successful case.

headers:

ETag:

$ref: '#/components/headers/ETag'

Last-Modified:

$ref: '#/components/headers/Last-Modified'

'304':

$ref: '#/components/responses/304'

'400':

$ref: 'TS29571\_CommonData.yaml#/components/responses/400'

'401':

$ref: 'TS29571\_CommonData.yaml#/components/responses/401'

'403':

$ref: 'TS29571\_CommonData.yaml#/components/responses/403'

'404':

$ref: 'TS29571\_CommonData.yaml#/components/responses/404'

'408':

$ref: 'TS29571\_CommonData.yaml#/components/responses/408'

'412':

description: Return value if get-previous=true

content:

application/json:

schema:

$ref: '#/components/schemas/MetaSchema'

'500':

$ref: 'TS29571\_CommonData.yaml#/components/responses/500'

'503':

$ref: 'TS29571\_CommonData.yaml#/components/responses/503'

default:

$ref: 'TS29571\_CommonData.yaml#/components/responses/default'

components:

securitySchemes:

oAuth2ClientCredentials:

type: oauth2

flows:

clientCredentials:

tokenUrl: '{nrfApiRoot}/oauth2/token'

scopes:

nudsf-dr: Access to the nudsf-dr API

\*\*\*\*\*\*\*\*\*\*\*text not shown for clarity\*\*\*\*\*\*\*\*\*\*\*\*

RecordMeta:

description: Meta data of a Record

type: object

properties:

ttl:

$ref: 'TS29571\_CommonData.yaml#/components/schemas/DateTime'

callbackReference:

$ref: 'TS29571\_CommonData.yaml#/components/schemas/Uri'

tags:

type: object # dictionary type

description: >-

A dictionary of {"tagName": [ "tagValue", ...] }. A tag name can be used to retrieve a Record. The tagValue are unique.

additionalProperties:

type: array

items:

type: string

uniqueItems: true

example: '{"ueId" : [ "455345", "455346" ], "recordId" : [ "1000106" ] }'

example: >-

{ "tags" : {"ueId" : [ "455345", "455346" ], "recordId" : [ "1000106" ] }}

schemaId:

$ref: '#/components/schemas/SchemaId'

\*\*\*\*\*\*\*\*\*\*\*text not shown for clarity\*\*\*\*\*\*\*\*\*\*\*\*

SearchCondition:

description: A logical condition

type: object

properties:

cond:

$ref: '#/components/schemas/ConditionOperator'

units:

type: array

items:

$ref: '#/components/schemas/SearchExpression'

minItems: 1

schemaId:

$ref: '#/components/schemas/SchemaId'

required:

- cond

- units

example:

{ "cond": "OR", "units": [ { "op": "EQ", "tag" : "ueId", "value" : "455345" }, { "op": "EQ", "tag" : "supi", "value" : "imsi-999559807001001" } ] }

SearchComparison:

description: A comparison to apply on tag/values pairs.

type: object

properties:

op:

$ref: '#/components/schemas/ComparisonOperator'

tag:

type: string

value:

type: string

required:

- tag

- value

example:

{ "op": "EQ", "tag" : "supi", "value" : "imsi-999559807001001" }

MetaSchema:

description: Defines the Meta Schema

type: object

required:

- schemaId

- metaTags

properties:

schemaId:

$ref: '#/components/schemas/SchemaId'

metaTags:

type: array

items:

$ref: '#/components/schemas/TagType'

TagType:

description: Defines the Tag Type

type: object

required:

- tagName

- keyType

properties:

tagName:

type: string

keyType:

$ref: '#/components/schemas/KeyType'

sort:

type: boolean

default: false

presence:

type: boolean

SchemaId:

type: string

KeyType:

anyOf:

- type: string

enum:

- UNIQUE\_KEY

- SEARCH\_KEY

- COUNT\_KEY

- SEARCH\_AND\_COUNT\_KEY

- OTHER\_TAG

- type: string

\*\*\*\*\*\*\*\*\*\*\*text not shown for clarity\*\*\*\*\*\*\*\*\*\*\*\*

\* \* \* End Of Change \* \* \* \*