

ETSI TS 129 598 V16.4.0 (2021-04)



**5G;
Unstructured data storage services
(3GPP TS 29.598 version 16.4.0 Release 16)**



Reference

RTS/TSGC-0429598vg40

Keywords

5G

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Important notice

The present document can be downloaded from:

<http://www.etsi.org/standards-search>

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at www.etsi.org/deliver.

Users of the present document should be aware that the document may be subject to revision or change of status.

Information on the current status of this and other ETSI documents is available at

<https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx>

If you find errors in the present document, please send your comment to one of the following services:

<https://portal.etsi.org/People/CommitteeSupportStaff.aspx>

Notice of disclaimer & limitation of liability

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of experience to understand and interpret its content in accordance with generally accepted engineering or other professional standard and applicable regulations.

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

No representation or warranty is made that this deliverable is technically accurate or sufficient or conforms to any law and/or governmental rule and/or regulation and further, no representation or warranty is made of merchantability or fitness for any particular purpose or against infringement of intellectual property rights.

In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the software.

Copyright Notification

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.

The content of the PDF version shall not be modified without the written authorization of ETSI.

The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI 2021.
All rights reserved.

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: "*Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards*", which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (<https://ipr.etsi.org/>).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM®** and the GSM logo are trademarks registered and owned by the GSM Association.

Legal Notice

This Technical Specification (TS) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities. These shall be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between 3GPP and ETSI identities can be found under <http://webapp.etsi.org/key/queryform.asp>.

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the [ETSI Drafting Rules](#) (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Contents

Intellectual Property Rights	2
Legal Notice	2
Modal verbs terminology.....	2
Foreword.....	7
1 Scope	9
2 References	9
3 Definitions of terms, symbols and abbreviations	10
3.1 Terms.....	10
3.2 Symbols.....	10
3.3 Abbreviations	10
4 Overview	10
5 Services offered by the UDSF.....	11
5.1 Introduction	11
5.2 Nudsf_DataRepository Service	11
5.2.1 Service Description.....	11
5.2.2 Service Operations.....	11
5.2.2.1 Introduction.....	11
5.2.2.2 Query.....	12
5.2.2.2.1 General	12
5.2.2.2.2 Record Retrieval.....	12
5.2.2.2.3 Meta Retrieval	12
5.2.2.2.4 Blocks Retrieval	13
5.2.2.2.5 Block Retrieval.....	13
5.2.2.2.6 Search.....	14
5.2.2.2.7 Subscriptions Retrieval.....	14
5.2.2.2.8 Individual Subscription Retrieval	15
5.2.2.3 Create	15
5.2.2.3.1 General	15
5.2.2.3.2 Record Create	16
5.2.2.3.3 Block Create	16
5.2.2.4 Update.....	17
5.2.2.4.1 General	17
5.2.2.4.2 Record Update.....	17
5.2.2.4.3 Block Update.....	18
5.2.2.4.4 Meta Update	18
5.2.2.4.5 Subscription Notification Update	19
5.2.2.4.6 Subscription Notification Update using PUT	19
5.2.2.5 Delete	20
5.2.2.5.1 General	20
5.2.2.5.2 Record Delete	20
5.2.2.5.3 Block Delete	21
5.2.2.6 Notify	21
5.2.2.6.1 General	21
5.2.2.6.2 Record Expiry Notify	21
5.2.2.6.3 Notification due to Data Change	22
5.2.2.7 Subscribe	22
5.2.2.7.1 General	22
5.2.2.7.2 Subscription to notifications of data change.....	22
5.2.2.8 Unsubscribe.....	23
5.2.2.8.1 General	23
5.2.2.8.2 Unsubscription to notifications of data change.....	23
6 API Definitions	24

6.1	Nudsf_DataRepository Service API.....	24
6.1.1	Introduction.....	24
6.1.2	Usage of HTTP.....	24
6.1.2.1	General.....	24
6.1.2.2	HTTP standard headers.....	25
6.1.2.2.1	General.....	25
6.1.2.2.2	Content type.....	25
6.1.2.2.3	Cache-Control.....	25
6.1.2.2.4	ETag.....	25
6.1.2.2.5	If-None-Match.....	25
6.1.2.2.6	If-Match.....	25
6.1.2.2.7	Last-Modified.....	25
6.1.2.2.8	If-Modified-Since.....	26
6.1.2.2.9	When to Use Entity-Tags and Last-Modified Dates.....	26
6.1.2.2.10	Content-Location.....	26
6.1.2.3	HTTP custom headers.....	26
6.1.2.4	HTTP multipart messages.....	26
6.1.2.4.1	General.....	26
6.1.2.4.2	Record.....	26
6.1.2.4.3	BlockCollection.....	26
6.1.2.4.4	RecordNotification.....	27
6.1.3	Resources.....	27
6.1.3.1	Overview.....	27
6.1.3.2	Resource: RecordCollection (Collection).....	28
6.1.3.2.1	Description.....	28
6.1.3.2.2	Resource Definition.....	28
6.1.3.2.3	Resource Standard Methods.....	29
6.1.3.2.3.1	GET.....	29
6.1.3.3	Resource: Record (Document).....	29
6.1.3.3.1	Description.....	29
6.1.3.3.2	Resource Definition.....	30
6.1.3.3.3	Resource Standard Methods.....	30
6.1.3.3.3.1	GET.....	30
6.1.3.3.3.2	PUT.....	30
6.1.3.3.3.3	DELETE.....	31
6.1.3.4	Resource: Meta (Document).....	32
6.1.3.4.1	Description.....	32
6.1.3.4.2	Resource Definition.....	32
6.1.3.4.3	Resource Standard Methods.....	33
6.1.3.4.3.1	GET.....	33
6.1.3.4.3.2	PATCH.....	33
6.1.3.5	Resource: BlockCollection (Collection).....	34
6.1.3.5.1	Description.....	34
6.1.3.5.2	Resource Definition.....	34
6.1.3.5.3	Resource Standard Methods.....	34
6.1.3.5.3.1	GET.....	34
6.1.3.6	Resource: Block (Document).....	35
6.1.3.6.1	Description.....	35
6.1.3.6.2	Resource Definition.....	35
6.1.3.6.3	Resource Standard Methods.....	35
6.1.3.6.3.1	GET.....	35
6.1.3.6.3.2	PUT.....	36
6.1.3.6.3.3	DELETE.....	37
6.1.3.7	Resource: NotificationSubscriptions.....	38
6.1.3.7.1	Description.....	38
6.1.3.7.2	Resource Definition.....	38
6.1.3.7.3	Standard Methods.....	38
6.1.3.7.3.2	GET.....	38
6.1.3.8	Resource: IndividualNotificationSubscription.....	39
6.1.3.8.1	Description.....	39
6.1.3.8.2	Resource Definition.....	39
6.1.3.8.3	Resource Standard Methods.....	39

6.1.3.8.3.1	DELETE	39
6.1.3.8.3.2	PATCH	40
6.1.3.8.3.3	GET	41
6.1.3.8.3.4	PUT	41
6.1.4	Custom Operations without associated resources	42
6.1.5	Notifications	42
6.1.5.1	General	42
6.1.5.2	Timer Expiry Notification	42
6.1.5.2.1	Description	42
6.1.5.2.2	Target URI	42
6.1.5.2.3	Standard Methods	43
6.1.5.2.3.1	POST	43
6.1.5.3	Notification due to Data Change	43
6.1.5.3.1	Description	43
6.1.5.3.2	Target URI	43
6.1.5.3.3	Standard Methods	43
6.1.5.3.3.1	POST	43
6.1.6	Data Model	44
6.1.6.1	General	44
6.1.6.2	Structured data types	44
6.1.6.2.1	Introduction	44
6.1.6.2.2	Type: RecordSearchResult	45
6.1.6.2.3	Type: RecordMeta	45
6.1.6.2.4	Type: RecordBody	45
6.1.6.2.5	Type: Record	45
6.1.6.2.6	Type: BlockBody	45
6.1.6.2.7	Type: Block	46
6.1.6.2.8	Type: SearchCondition	46
6.1.6.2.9	SearchComparison	46
6.1.6.2.10	Type: NotificationSubscription	47
6.1.6.2.11	Type: RecordNotification	47
6.1.6.2.12	Type: NotificationDescription	47
6.1.6.2.13	Type: SubscriptionFilter	48
6.1.6.2.14	Type: ClientId	48
6.1.6.3	Simple data types and enumerations	48
6.1.6.3.1	Introduction	48
6.1.6.3.2	Simple data types	49
6.1.6.3.3	Enumeration: ComparisonOperator	49
6.1.6.3.4	Enumeration: ConditionOperator	49
6.1.6.3.5	Enumeration: RecordOperation	49
6.1.6.4	Data types describing alternative data types or combinations of data types	50
6.1.6.4.1	Type: SearchExpression	50
6.1.7	Error Handling	50
6.1.7.1	General	50
6.1.7.2	Protocol Errors	50
6.1.7.3	Application Errors	50
6.1.8	Feature negotiation	50
6.1.9	Security	51
Annex A (normative):	OpenAPI specification	52
A.1	General	52
A.2	Nudsf_DataRepository API	52
Annex B (informative):	Search Examples	76
Annex C (informative):	HTTP Multipart Examples	78
C.1	General	78
C.2	Example HTTP multipart Record	78
C.3	Example HTTP multipart BlockCollection	78
C.4	Example HTTP multipart RecordNotification	79
Annex D (informative):	Change history	80

History81

Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
 - 1 presented to TSG for information;
 - 2 presented to TSG for approval;
 - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

In the present document, modal verbs have the following meanings:

- shall** indicates a mandatory requirement to do something
- shall not** indicates an interdiction (prohibition) to do something

The constructions "shall" and "shall not" are confined to the context of normative provisions, and do not appear in Technical Reports.

The constructions "must" and "must not" are not used as substitutes for "shall" and "shall not". Their use is avoided insofar as possible, and they are not used in a normative context except in a direct citation from an external, referenced, non-3GPP document, or so as to maintain continuity of style when extending or modifying the provisions of such a referenced document.

- should** indicates a recommendation to do something
- should not** indicates a recommendation not to do something
- may** indicates permission to do something
- need not** indicates permission not to do something

The construction "may not" is ambiguous and is not used in normative elements. The unambiguous constructions "might not" or "shall not" are used instead, depending upon the meaning intended.

- can** indicates that something is possible
- cannot** indicates that something is impossible

The constructions "can" and "cannot" are not substitutes for "may" and "need not".

- will** indicates that something is certain or expected to happen as a result of action taken by an agency the behaviour of which is outside the scope of the present document
- will not** indicates that something is certain or expected not to happen as a result of action taken by an agency the behaviour of which is outside the scope of the present document
- might** indicates a likelihood that something will happen as a result of action taken by some agency the behaviour of which is outside the scope of the present document

might not indicates a likelihood that something will not happen as a result of action taken by some agency the behaviour of which is outside the scope of the present document

In addition:

is (or any other verb in the indicative mood) indicates a statement of fact

is not (or any other negative verb in the indicative mood) indicates a statement of fact

The constructions "is" and "is not" do not indicate requirements.

1 Scope

The present document specifies the stage 3 protocol and data model for the Nudsf Service Based Interface. It provides stage 3 protocol definitions and message flows, and specifies the API for each service offered by the UDSF.

The 5G System stage 2 architecture and procedures are specified in 3GPP TS 23.501 [2] and 3GPP TS 23.502 [3].

The Technical Realization of the Service Based Architecture and the Principles and Guidelines for Services Definition are specified in 3GPP TS 29.500 [4] and 3GPP TS 29.501 [5].

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

- [1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".
- [2] 3GPP TS 23.501: "System Architecture for the 5G System; Stage 2".
- [3] 3GPP TS 23.502: "Procedures for the 5G System; Stage 2".
- [4] 3GPP TS 29.500: "5G System; Technical Realization of Service Based Architecture; Stage 3".
- [5] 3GPP TS 29.501: "5G System; Principles and Guidelines for Services Definition; Stage 3".
- [6] OpenAPI: "OpenAPI 3.0.0 Specification", <https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.0.md>.
- [7] 3GPP TR 21.900: "Technical Specification Group working methods".
- [8] 3GPP TS 33.501: "Security architecture and procedures for 5G system".
- [9] IETF RFC 6749: "The OAuth 2.0 Authorization Framework".
- [10] 3GPP TS 29.510: "5G System; Network Function Repository Services; Stage 3".
- [11] IETF RFC 7540: "Hypertext Transfer Protocol Version 2 (HTTP/2)".
- [12] IETF RFC 8259: "The JavaScript Object Notation (JSON) Data Interchange Format".
- [13] IETF RFC 7807: "Problem Details for HTTP APIs".
- [14] IETF RFC 6902: "JavaScript Object Notation (JSON) Patch".
- [15] IETF RFC 7231: "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content".
- [16] IETF RFC 7232: "Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests".
- [17] IETF RFC 7234: "Hypertext Transfer Protocol (HTTP/1.1): Caching".
- [18] ISO/IEC 14977: "Information technology – Syntactic metalanguage - Extended BNF".
- [19] 3GPP TS 29.571: "5G System; Common Data Types for Service Based Interfaces; Stage 3".

- [20] IETF RFC 2045: "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies".
- [21] IETF RFC 2046: "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types".
- [22] IETF RFC 3986: "Uniform Resource Identifier (URI): Generic Syntax".

3 Definitions of terms, symbols and abbreviations

3.1 Terms

void

3.2 Symbols

void

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in 3GPP TR 21.905 [1] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in 3GPP TR 21.905 [1].

5GC	5G Core Network
BNF	Backus–Naur Form
EBNF	Extended BNF
CP	Control Plane
MIME	Multipurpose Internet Mail Extensions
NF	Network Function
UDSF	Unstructured Data Storage Function

4 Overview

The UDSF, as depicted in Figure 4.1-1 below, is described in clause 4.2.5 of 3GPP TS 23.501 [2]. Any of the 5GS NFs can make use of the UDSF to store and retrieve unstructured data, i.e., data that is not defined in 3GPP specifications. The UDSF is deployed in the same network where the CP NF is located and the same UDSF may be shared by all the NFs in the PLMN to store/retrieve their respective data or an NF may have its own UDSF depending on operator configuration.

NOTE 1: Structured data in this specification refers to data for which the structure is defined in 3GPP specifications. Unstructured data refers to data for which the structure is not defined in 3GPP specifications.

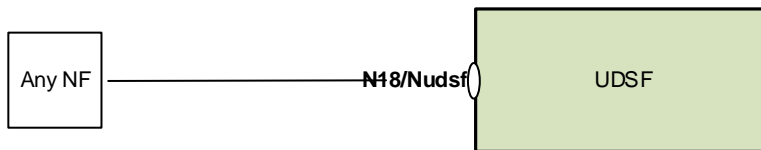


Figure 4.1-1: Reference model – UDSF

5 Services offered by the UDSF

5.1 Introduction

The UDSF offers the following services via the Nudsf service based interface:

- Nudsf_DataRepository Service

NOTE: This service corresponds to the Nudsf_UnstructuredDataManagement service in 3GPP TS 23.501 [2] and 3GPP TS 23.502 [3].

Table 5.1-1 summarizes the corresponding APIs defined for this specification.

Table 5.1-1: API Descriptions

Service Name	Clause	Description	OpenAPI Specification File	apiName	Annex
Nudsf_DataRepository	6.1	UDSF Data Repository Service	TS29598_Nudsf_DataRepository.yaml	nudsf-dr	A.2

5.2 Nudsf_DataRepository Service

5.2.1 Service Description

The UDSF is acting as an NF Service Producer. It provides UDSF data repository service to the NF service consumer. Any NF may use the UDSF to store unstructured data.

NOTE 1: Structured data in this specification refers to data for which the structure is defined in 3GPP specifications. Unstructured data refers to data for which the structure is not defined in 3GPP specifications.

5.2.2 Service Operations

5.2.2.1 Introduction

For the Nudsf_DataRepository service, the following service operations are defined:

- Query
- Create

- Update
- Delete
- Notify
- Subscribe
- Unsubscribe

5.2.2.2 Query

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

5.2.2.2.2 Record Retrieval

Figure 5.2.2.2.2-1 shows a scenario where the NF service consumer sends a request to the UDSF to retrieve a record that matches the provided recordId and optionally includes the query parameter supported-features.

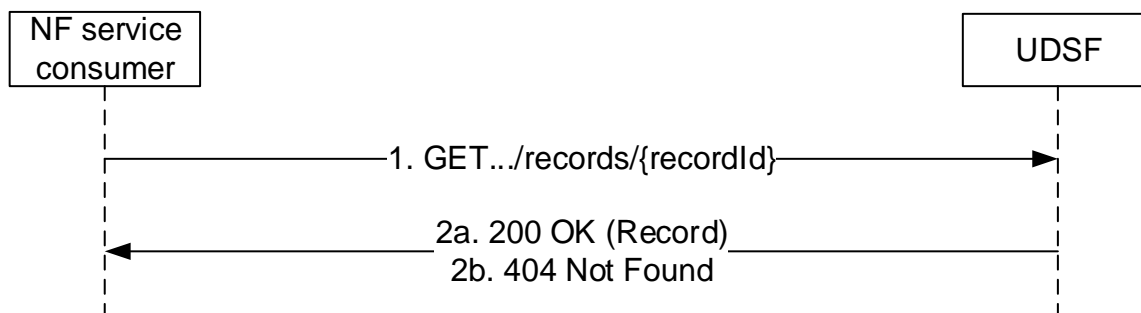


Figure 5.2.2.2.2-1: Requesting a Record

1. The NF service consumer (any NF) sends a GET request to the resource indicated by recordId.
- 2a. On success, the UDSF responds with "200 OK" with the message body containing the record.
- 2b. If the record for the given recordId 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.

5.2.2.2.3 Meta Retrieval

Figure 5.2.2.2.3-1 shows a scenario where the NF service consumer sends a request to the UDSF to retrieve meta data associated with the provided recordId and optionally includes the query parameter supported-features.

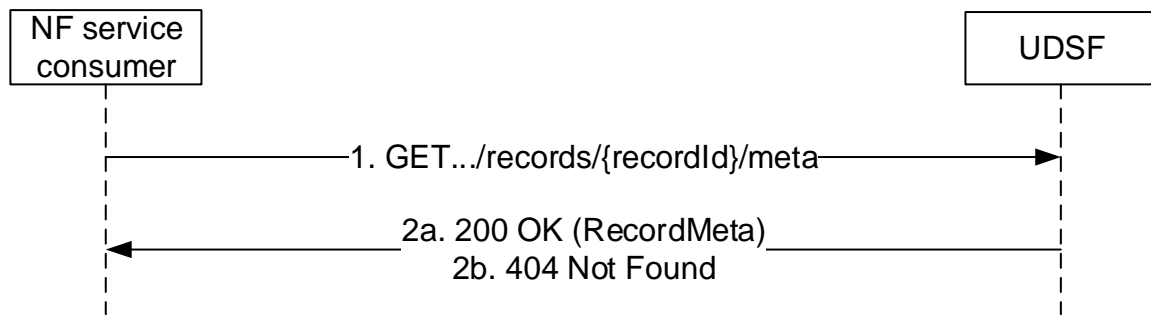


Figure 5.2.2.2.3-1: Requesting Meta for a Record

1. The NF service consumer (any NF) sends a GET request to the meta resource associated with the record indicated by recordId.
- 2a. On success, the UDSF responds with "200 OK" with the message body containing the RecordMeta.
- 2b. If the record for the given recordId and thus the RecordMeta 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.

5.2.2.2.4 Blocks Retrieval

Figure 5.2.2.2.4-1 shows a scenario where the NF service consumer sends a request to the UDSF to retrieve (all) the blocks associated with the provided recordId and optionally includes the query parameter supported-features.

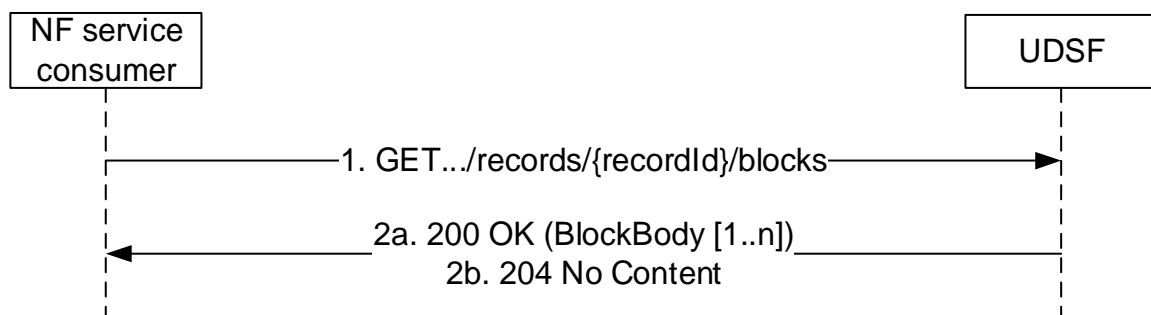


Figure 5.2.2.2.4-1: Requesting Blocks

1. The NF service consumer (any NF) sends a GET request to the resource indicated by recordId.
- 2a. On success, the UDSF responds with "200 OK" with the message body containing the Blocks associated with the record.
- 2b. If a Block for the given recordId does not exist in the UDSF, the HTTP status code "204 No Content" shall be returned.

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.

5.2.2.2.5 Block Retrieval

Figure 5.2.2.2.5-1 shows a scenario where the NF service consumer sends a request to the UDSF to retrieve a single block associated with the provided recordId and blockId and optionally includes the query parameter supported-features.

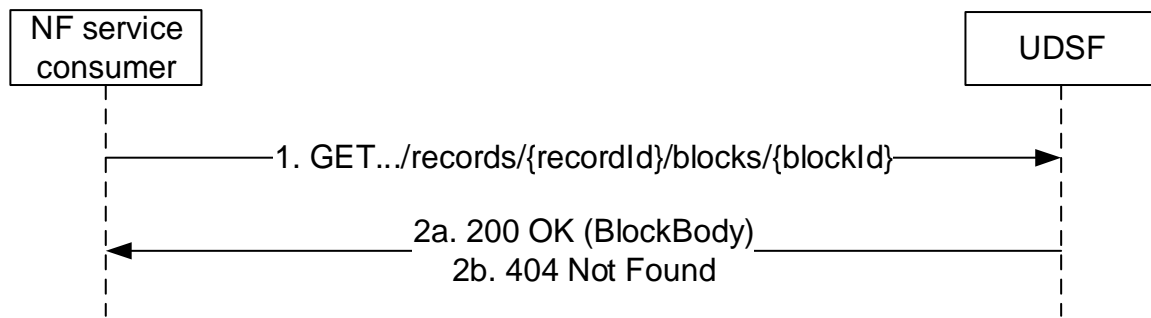


Figure 5.2.2.2.5-1: Requesting a Block

1. The NF service consumer (any NF) sends a GET request to the resource indicated by recordId and blockId.
- 2a. On success, the UDSF responds with "200 OK" with the message body containing the Block associated with the blockId.
- 2b. If the Block for the given recordId and blockId 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.

5.2.2.2.6 Search

Figure 5.2.2.2.6-1 shows a scenario where the NF service consumer sends a request to the UDSF to search a record that matches the provided search criteria.

The request contains the query parameters filter and optionally supported-features, limit-range, and count-indicator.

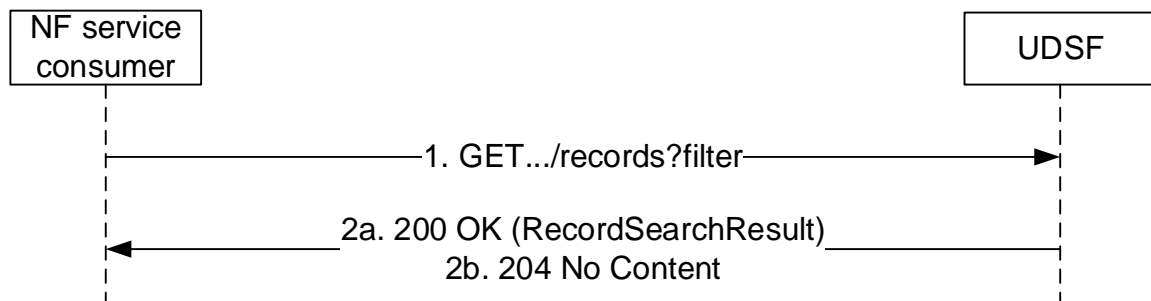


Figure 5.2.2.2.6-1: Searching for Records

1. The NF service consumer (any NF) sends a GET request to the Records resource with the filter query parameter indicating the search criteria.
- 2a. On success, the UDSF responds with "200 OK" with the message body containing the RecordSearchResult.
- 2b. If the UDSF is not able to return any record for the given search criteria, the HTTP status code "204 No Content" shall be returned.

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.

5.2.2.2.7 Subscriptions Retrieval

Figure 5.2.2.2.7-1 shows a scenario where the NF service consumer sends a request to the UDSF to retrieve all subscriptions associated with the provided storageId and optionally includes the query parameter supported-features and limit-range.

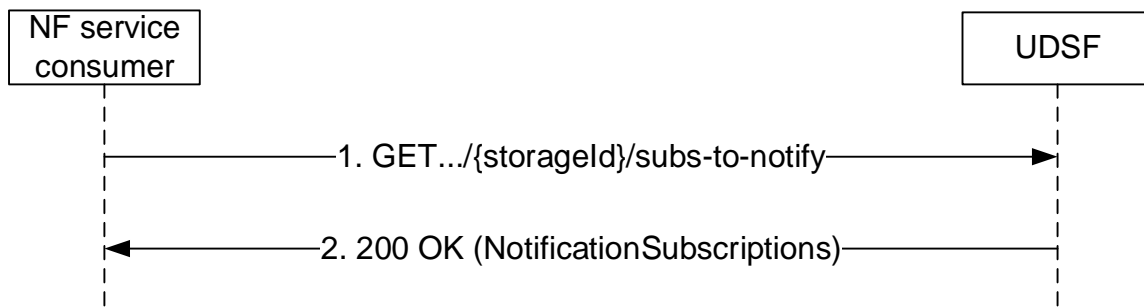


Figure 5.2.2.2.7-1: Requesting Subscriptions

1. The NF service consumer (any NF) sends a GET request to the resource indicated by the storageId.
2. On success, the UDSF responds with "200 OK" with the message body containing the NotificationSubscriptions associated with the storageId (if any).

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.

5.2.2.2.8 Individual Subscription Retrieval

Figure 5.2.2.2.8-1 shows a scenario where the NF service consumer sends a request to the UDSF to retrieve a subscription associated with the provided storageId and subscriptionId.

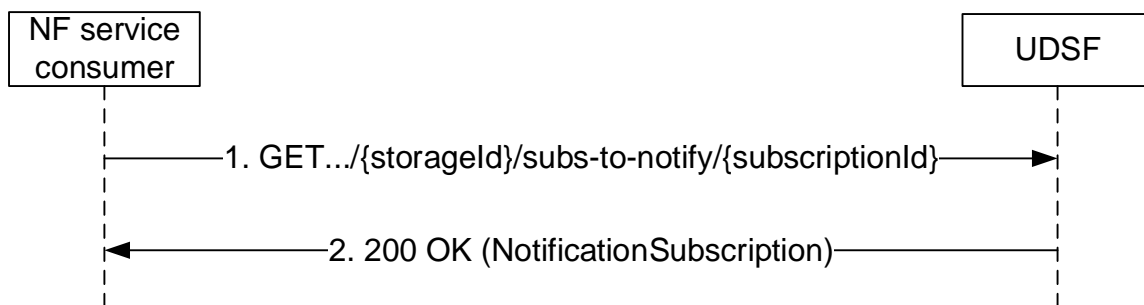


Figure 5.2.2.2.8-1: Requesting an Individual Subscription

1. The NF service consumer (any NF) sends a GET request to the resource indicated by the storageId and the subscriptionId.
- 2a. On success, the UDSF responds with "200 OK" with the message body containing the NotificationSubscription.

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.

5.2.2.3 Create

5.2.2.3.1 General

The following procedures using the Create service operation are supported:

- Record Create
- Block Create

5.2.2.3.2 Record Create

Figure 5.2.2.3.2-1 shows a scenario where the NF service consumer sends a request to the UDSF to create a record with the provided recordId.

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

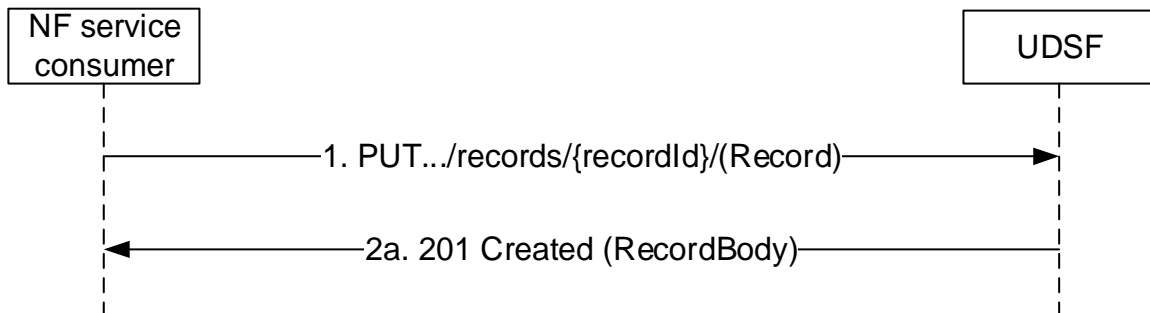


Figure 5.2.2.3.2-1: Create a Record

1. The NF service consumer (any NF) sends a PUT request to create the resource indicated by recordId. The request body contains the meta, zero or more blocks. The record meta information is mandatory and shall be the first part and the remaining parts of the request body (if any) shall be child blocks. If the record meta information is received with record expiry details, UDSF shall create an implicit subscription locally and notify the NF service consumer on record expiry.
- 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.

5.2.2.3.3 Block Create

Figure 5.2.2.3.3-1 shows a scenario where the NF service consumer sends a request to the UDSF to create a block with the provided blockId.

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

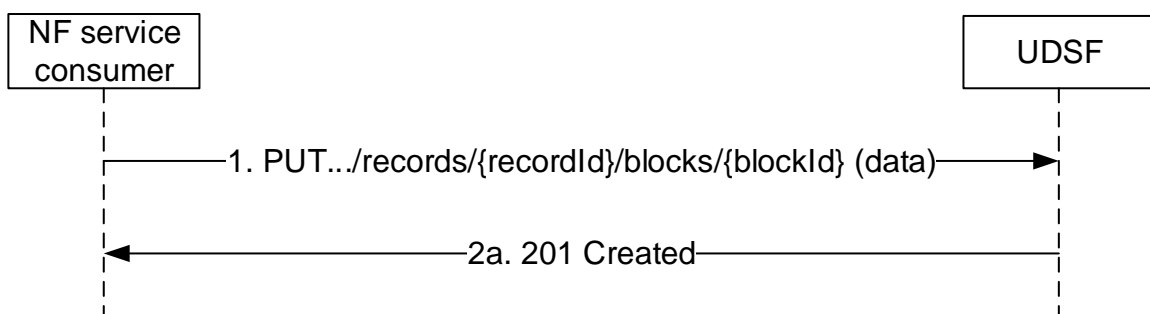


Figure 5.2.2.3.3-1: Create a Block

1. The NF service consumer (any NF) sends a PUT request to create the resource indicated by blockId.
- 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.

5.2.2.4 Update

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

5.2.2.4.2 Record Update

Figure 5.2.2.4.2-1 shows a scenario where the NF service consumer sends a request to the UDSF to update a record with the provided recordId.

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

The update shall include meta, zero or more blocks. The record meta information shall be the first part and is mandatory and the remaining parts of the body (if any) shall be interpreted as child blocks. Existing record, meta and blocks shall be discarded and the new record, meta and blocks (if any) shall be created.

NOTE: The order of the returned blocks in the response is not guaranteed and can be different from the order used to create them.

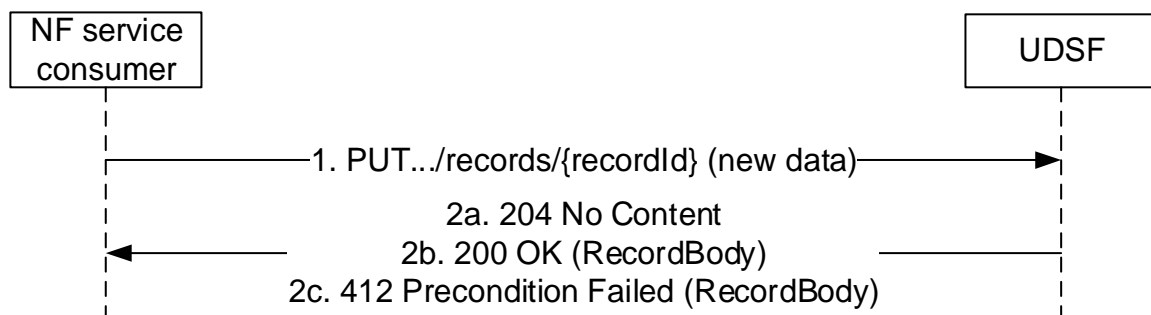


Figure 5.2.2.4.2-1: Update a record

1. The NF service consumer shall send a PUT request to the resource representing the record that is to be updated, and may include meta, zero or more blocks. The record meta information is mandatory and shall be the first part and the remaining parts of the request body (if any) shall be child blocks. An existing record, i.e., meta and blocks shall be discarded and the new record, meta and blocks (if any) shall be created.
- 2a. On success, the UDSF shall respond with "204 No Content" if no record 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 record is returned, i.e. the get-previous query parameter was included in the request, or due to operator's policy, the ttl value in the request exceeded the maximum value allowed.
- 2c. On failure, the UDSF shall respond with "412 Precondition Failed" if one or more conditions given in the request header fields evaluated to false. The RecordBody shall include the stored Record if the get-previous query parameter was included in the request.

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.

5.2.2.4.3 Block Update

Figure 5.2.2.4.3-1 shows a scenario where the NF service consumer sends a request to the UDSF to update a block with the provided blockId.

The request contains the recordId, blockId and the optional query parameters supported-features, get-previous and the data that is to be updated.

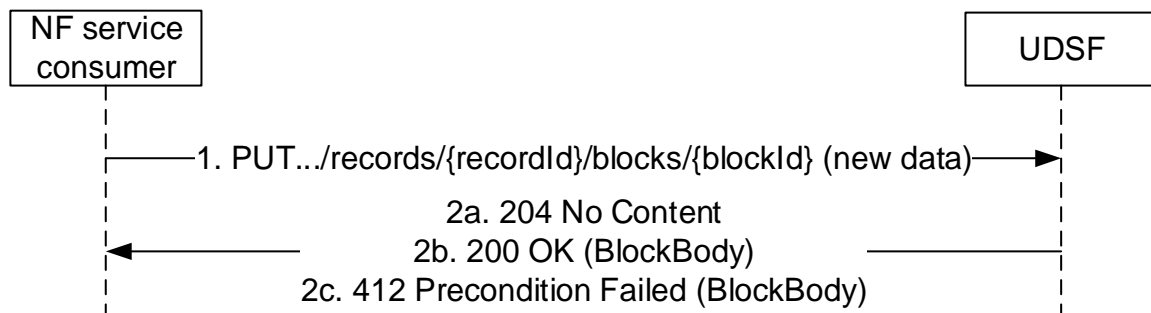


Figure 5.2.2.4.3-1: Update a block

1. The NF service consumer shall send a PUT request to the resource representing the block that is to be updated.
- 2a. On success, the UDSF shall respond with "204 No Content" if no record 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 record is returned, i.e. the get-previous query parameter was included in the request.
- 2c. On failure, the UDSF shall respond with "412 Precondition Failed" if one or more conditions given in the request header fields evaluated to false. The BlockBody shall be included with the stored Block if the get-previous query parameter was included in the request.

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.

5.2.2.4.4 Meta Update

Figure 5.2.2.4.4-1 shows a scenario where the NF service consumer sends a request to the UDSF to update the meta data associated with the provided recordId and optionally the query parameter supported-features and the data that is to be updated.

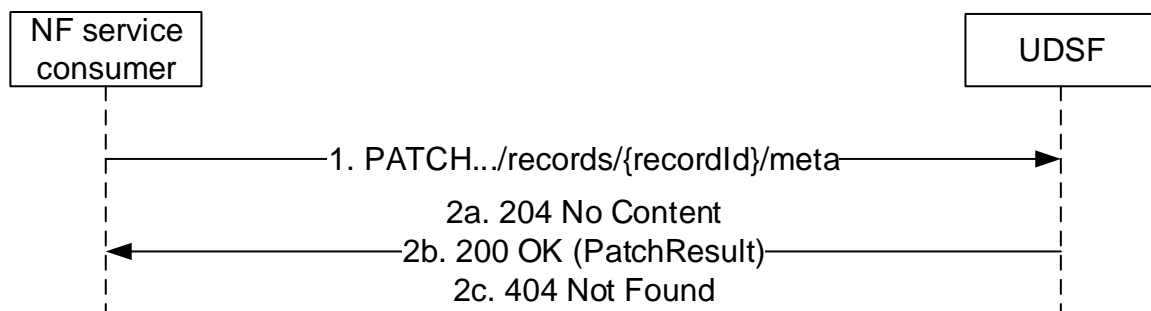


Figure 5.2.2.4.4-1: Update meta

1. The NF service consumer shall send a PATCH request to the resource representing the meta of the record.
- 2a. On success, if all the modification instructions in the PATCH request have been implemented, the UDSF shall respond with "204 No Content".
- 2b. On partial success, i.e. if one or more modification instructions have been discarded, "200 OK" with the execution report, shall be returned.

2c. On failure, the UDSF shall respond with "404 Not Found" if the record indicated by the recordId and thus the meta does not exist and may shall 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 PUT response body.

5.2.2.4.5 Subscription Notification Update

Figure 5.2.2.4.5-1 shows a scenario where the NF service consumer sends a request to the UDSF to update the Individual Subscription Notification identified with the storageId and subscriptionId and optionally the query parameter supported-features and the data that is to be updated.

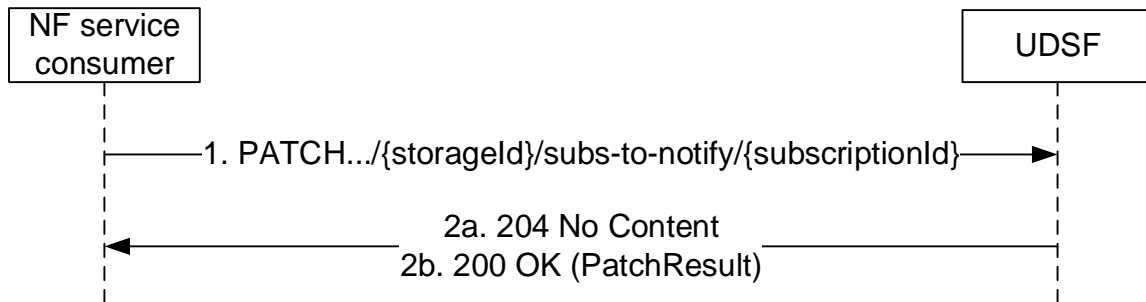


Figure 5.2.2.4.5-1: Update Subscription Notification

1. The NF service consumer shall send a PATCH request to the resource representing the subscriptionId.
- 2a. On success, if all the modification instructions in the PATCH request have been implemented, the UDSF shall respond with "204 No Content".
- 2b. On partial success, i.e. if one or more modification instructions have been discarded, "200 OK" with the execution report, shall be returned.

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

5.2.2.4.6 Subscription Notification Update using PUT

Figure 5.2.2.4.6-1 shows a scenario where the NF service consumer sends a request to the UDSF to update a subscription to notifications of data change using PUT. The request contains the subscriptionId and the NotificationSubscription and optionally the query parameter supported-features.

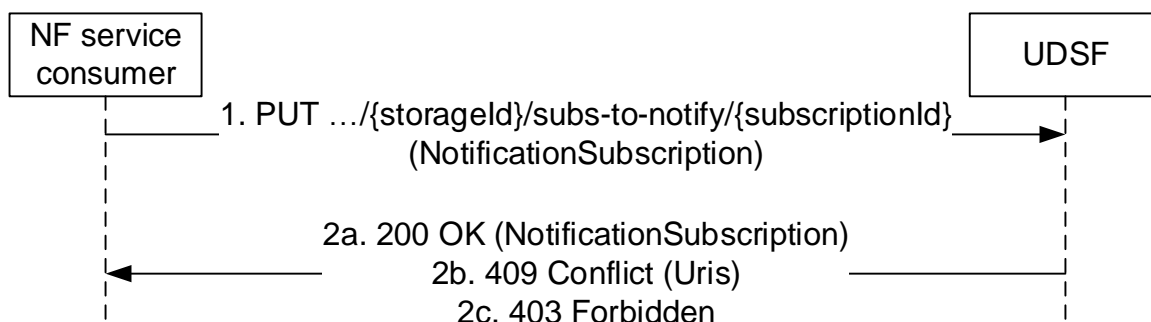


Figure 5.2.2.4.6-1: NF service consumer updates subscription to notifications

1. The NF service consumer sends a PUT request to the resource indicated by the storageId and the subscriptionId. The parameter clientId shall be included. If the resource indicated in URI exists and was created by the Client identified by the clientId, the UDSF shall apply the update of the subscription.

- 2a. On success, the UDSF shall respond with "200 OK" and include the updated NotificationSubscription. The expiry attribute of the received NotificationSubscription may indicate a value or a value different from the request, if due to an operator policy, an expiry time is enforced or if the value in the request exceeded a maximum allowed expiry time.
- 2b. On failure, if one or more monitoredResourceUris from the request don't exist in the UDSF, 409 Conflict shall be returned together with the non-existing monitoredResourceUris.
- 2c. On failure, if the service operation cannot be authorized due to e.g. the resource indicated in URI exists but the clientId in the PUT request does not match the clientId of the existing resource, the UDSF shall respond with "403 Forbidden" and optionally including additional error information in the response body (in "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 PUT response body.

5.2.2.5 Delete

5.2.2.5.1 General

The following procedures using the Delete service operation are supported:

- Record Delete
- Block Delete

5.2.2.5.2 Record Delete

Figure 5.2.2.5.2-1 shows a scenario where the NF service consumer sends a request to the UDSF to Delete a record with the provided recordId.

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

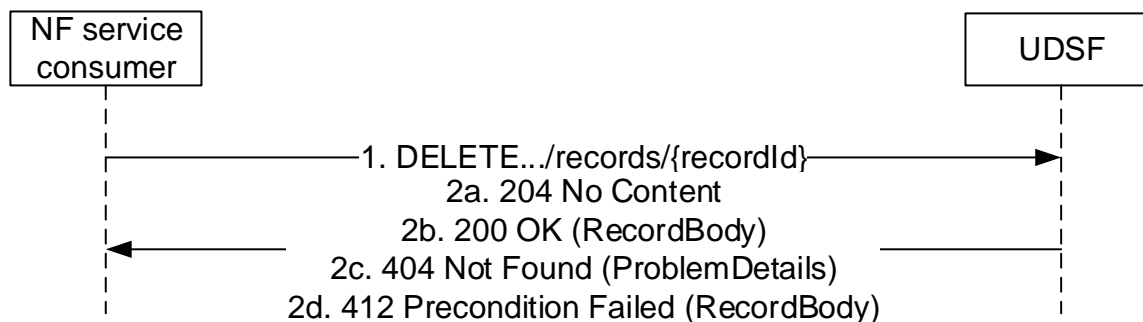


Figure 5.2.2.5.2-1: Delete a record

1. The NF service consumer shall send a DELETE request to the resource representing the record. The UDSF shall delete any resource associated with the resource (meta and block(s)).
- 2a. On success, the UDSF shall respond with "204 No Content" if no record 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 record 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 record does not exist and may include the ProblemDetails.
- 2d. On failure, the UDSF shall respond with "412 Precondition Failed" if one or more conditions given in the request header fields evaluated to false and the get-previous query parameter was included in the request. The RecordBody shall be included if the get-previous query parameter was included in the request.

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.

5.2.2.5.3 Block Delete

Figure 5.2.2.5.3-1 shows a scenario where the NF service consumer sends a request to the UDSF to Delete a block with the provided blockId.

The request contains the recordId, blockId and optionally the query parameters supported-features and get-previous.

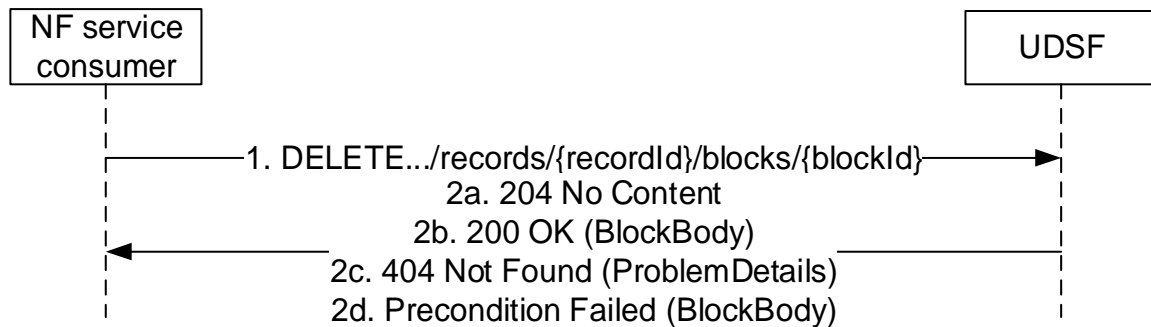


Figure 5.2.2.5.3-1: Delete a block

1. The NF service consumer shall send a DELETE request to the resource representing the block.
- 2a. On success, the UDSF shall respond with "204 No Content" if no block 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 block 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 block does not exist and may include the ProblemDetails.
- 2d. On failure, the UDSF shall respond with "412 Precondition Failed" if one or more conditions given in the request header fields evaluated to false. The BlockBody shall be included if the get-previous query parameter was included in the request.

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.

5.2.2.6 Notify

5.2.2.6.1 General

The following procedures using the Notify service operation are supported:

- Record Expiry Notify
- Notification due to Data Change

5.2.2.6.2 Record Expiry Notify

Figure 5.2.2.6.2-1 shows a scenario where the UDSF notifies the NF service consumer of the expired record.

The Notify is sent by the UDSF to the NF Service Consumer when the record expires as indicated by the time to live (ttl) attribute of RecordMeta.

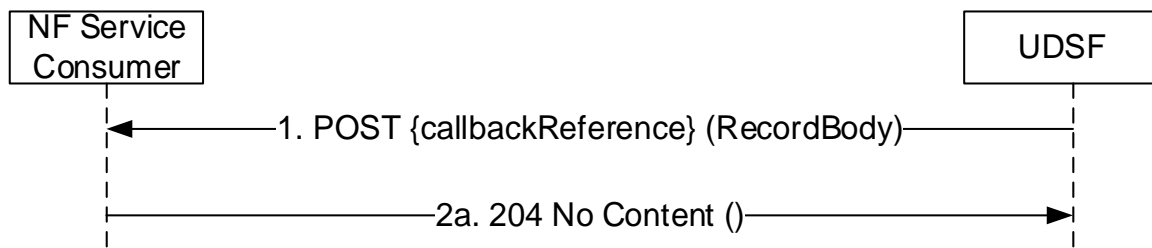


Figure 5.2.2.6.2-1: Record Expiry Notify

1. The UDSF shall send a POST request to the callback URI. The request shall contain the record details.

2a. On success, "204 No content" shall be returned by the NF Service Consumer to UDSF.

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

5.2.2.6.3 Notification due to Data Change

Figure 5.2.2.6.3-1 shows a scenario where the UDSF notifies the NF service consumer of a change to data associated with a block or a record triggered by one or more a Subscription to Notification.

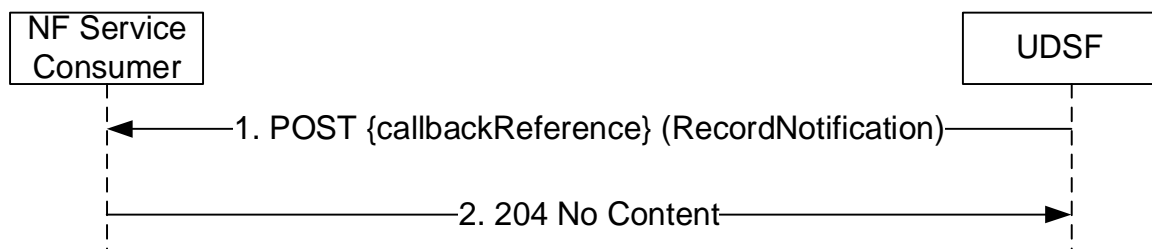


Figure 5.2.2.6.3-1: Notification due to Data Change

1. The UDSF shall send a POST request to the callback URI. The request shall contain the RecordNotification details.

2. On success, "204 No content" shall be returned by the NF Service Consumer to UDSF.

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

5.2.2.7 Subscribe

5.2.2.7.1 General

The following procedures using the Subscribe service operation are supported:

- Subscription to notification of data change

5.2.2.7.2 Subscription to notifications of data change

Figure 5.2.2.7.2-1 shows a scenario where the NF service consumer sends a request to the UDSF to subscribe to notifications of data change. The request contains the subscriptionId, the NotificationSubscription and optionally the query parameter supported-features.

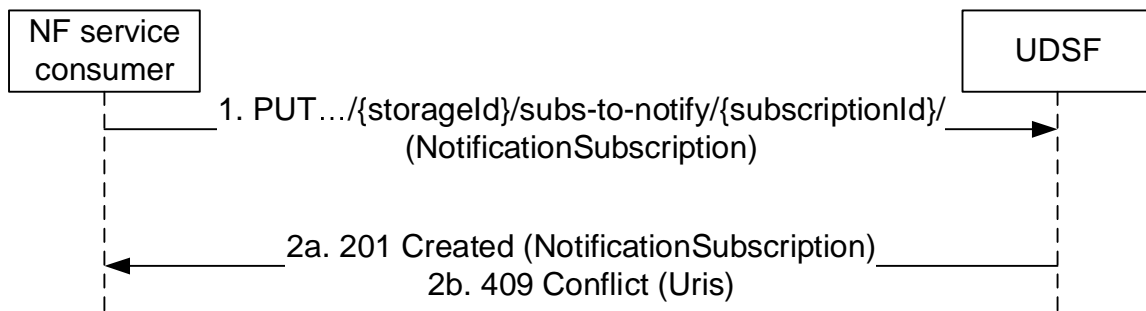


Figure 5.2.2.7.2-1: NF service consumer subscribes to notifications

1. The NF service consumer sends a PUT request to the resource indicated by the storageId and the subscriptionId. The parameter clientId shall be included. If the resource indicated in URI doesn't exist, the UDSF shall trigger the creation of the subscription.
- 2a. On success, the UDSF responds with "201 Created" with the message body containing the NotificationSubscription. The expiry attribute of the received NotificationSubscription may indicate a value or a value different from the request, if due to an operator policy, an expiry time is enforced or if the value in the request exceeded a maximum allowed expiry time.
- 2b. On failure, if one or more monitoredResourceUris from the request don't exist in the UDSF, 409 Conflict shall be returned together with the non-existing monitoredResourceUris.

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.

5.2.2.8 Unsubscribe

5.2.2.8.1 General

The following procedures using the Unsubscribe service operation are supported:

- Unsubscription to notification of data change

5.2.2.8.2 Unsubscription to notifications of data change

Figure 5.2.2.8.2-1 shows a scenario where the NF service consumer sends a request to the UDSF to Unsubscribe to notifications of data change. The request contains the subscriptionId, and query parameter client-id, optionally query parameters supported-features and get-previous.

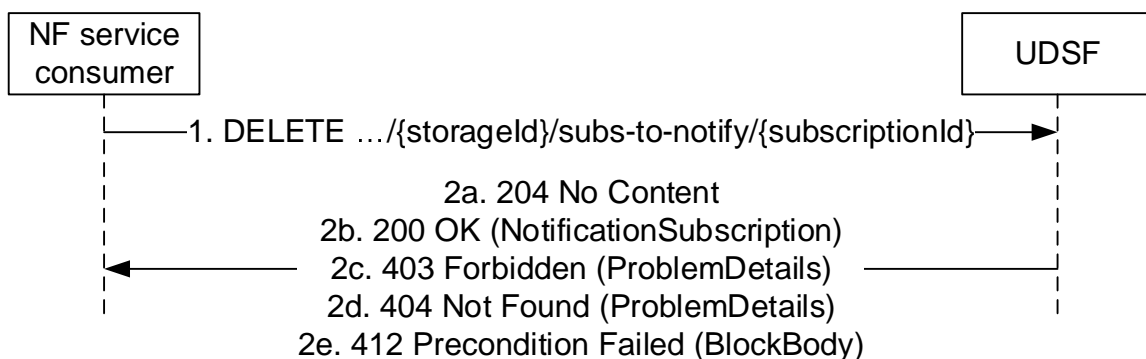


Figure 5.2.2.8.2-1: NF service consumer unsubscribes to notifications

1. The NF service consumer sends a DELETE request to the resource representing the subscription to notification of data change which is indicated by the subscriptionId.

- 2a. On success, the UDSF shall respond with "204 No Content" if an empty response body 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" with NotificationSubscription containing the NotificationSubscription value before the delete if get-previous was indicated in the request.
- 2c. If the service operation cannot be authorized due to e.g. the client-id query parameter does not match the clientId of the existing resource, the UDSF shall respond with "403 Forbidden" with including additional error information in the response body (in "ProblemDetails" element).
- 2d. If there is no valid subscription to notification of data change which is indicated by the request, the UDSF shall respond with "404 Not Found" with including additional error information in the response body (in "ProblemDetails" element).
- 2e. If one or more conditions given in the request header fields evaluated to false, the UDSF shall respond with "412 Precondition Failed". NotificationSubscription shall be included in the response if get-previous was indicated in the request.

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.

6 API Definitions

6.1 Nudsf_DataRepository Service API

6.1.1 Introduction

The Nudsf_DataRepository service shall use the Nudsf_DataRepository API.

The API URI of the Nudsf_DataRepository API shall be:

{apiRoot}/<apiName>/<apiVersion>/

The request URI used in HTTP requests from the NF service consumer towards the NF service producer shall have the Resource URI structure defined in clause 4.4.1 of 3GPP TS 29.501 [5], i.e.:

{apiRoot}/<apiName>/<apiVersion>/<apiSpecificResourceUriPart>

with the following components:

- The {apiRoot} shall be set as described in 3GPP TS 29.501 [5].
- The <apiName> shall be "nudsf-dr".
- The <apiVersion> shall be "v1".
- The <apiSpecificResourceUriPart> shall be set as described in clause 6.1.3.

6.1.2 Usage of HTTP

6.1.2.1 General

HTTP/2, IETF RFC 7540 [11], shall be used as specified in clause 5 of 3GPP TS 29.500 [4].

HTTP/2 shall be transported as specified in clause 5.3 of 3GPP TS 29.500 [4].

The OpenAPI [6] specification of HTTP messages and content bodies for the Nudsf_DataRepository API is contained in Annex A.

6.1.2.2 HTTP standard headers

6.1.2.2.1 General

See clause 5.2.2 of 3GPP TS 29.500 [4] for the usage of HTTP standard headers.

6.1.2.2.2 Content type

The following content types shall be supported:

- JSON, IETF RFC 8259 [12], shall be used as content type of the HTTP bodies specified in the present specification as specified in clause 5.4 of 3GPP TS 29.500 [4]. The use of the JSON format shall be signalled by the content type "application/json".
- "Problem Details" JSON object shall be used to indicate additional details of the error in a HTTP response body and shall be signalled by the content type "application/problem+json", as defined in IETF RFC 7807 [13].
- JSON Patch (IETF RFC 6902 [14]). The use of the JSON Patch format in a HTTP request body shall be signalled by the content type "application/json-patch+json".

Multipart messages shall also be supported possibly indicating other content-types as described in clause 6.1.2.4.

6.1.2.2.3 Cache-Control

As described in IETF RFC 7234 [17] clause 5.2, a "Cache-Control" header should be included in HTTP responses carrying a representation of cacheable resources. If it is included, it shall contain a "max-age" value, indicating the amount of time in seconds after which the received response is considered stale.

The "max-age" value shall be configurable by operator policy.

6.1.2.2.4 ETag

As described in IETF RFC 7232 [16] clause 2.3, an "ETag" (entity-tag) header should be included in HTTP responses carrying a representation of cacheable or modifiable resources to allow an NF Service Consumer performing a conditional GET request with an "If-None-Match" header or a conditional PUT/PATCH/DELETE request with an "If-Match" header. If it is included, it shall contain a server-generated strong validator, that allows further matching of this value (included in subsequent client requests) with a given resource representation stored in the server or in a cache.

6.1.2.2.5 If-None-Match

As described in IETF RFC 7232 [16] clause 3.2, an NF Service Consumer may issue conditional GET or PUT requests towards UDSF by including an "If-None-Match" header in HTTP requests containing one or several entity tags received in previous responses for the same resource.

If the If-None-Match header is included with the PUT method, it shall be used with a value of "*" to prevent the inadvertent modification of an existing representation of the target resource when the client believes that the resource does not have a current representation.

6.1.2.2.6 If-Match

As described in IETF RFC 7232 [16] clause 3.1, an NF Service Consumer may issue conditional PUT/PATCH/DELETE request towards UDSF by including an "If-Match" header in HTTP requests containing an entity tag received in previous responses for the same resource.

6.1.2.2.7 Last-Modified

As described in IETF RFC 7232 [16] clause 2.2, a "Last-Modified" header should be included in HTTP responses carrying a representation of cacheable resources to allow an NF Service Consumer performing a conditional request with "If-Modified-Since" header.

6.1.2.2.8 If-Modified-Since

As described in IETF RFC 7232 [16] clause 3.3, an NF Service Consumer may issue conditional GET request towards UDSF, by including an "If-Modified-Since" header in HTTP requests.

6.1.2.2.9 When to Use Entity-Tags and Last-Modified Dates

Both "ETag" and "Last-Modified" headers should be sent in the same HTTP response as stated in IETF RFC 7232 [16] clause 2.4.

NOTE: "ETag" is a stronger validator than the "Last-Modified" and is preferred.

If the UDSF included an "ETag" header with the resource then a conditional GET request for this resource shall be performed with the "If-None-Match" header, and a PUT/PATCH/DELETE request for this resource shall be performed with the "If-Match" header.

6.1.2.2.10 Content-Location

As described in IETF RFC 7231 [15] clause 3.1.4.2, the UDSF shall include the Content-Location header set to the URI of the expired Record when sending a Notification to an NF Consumer.

6.1.2.3 HTTP custom headers

The mandatory HTTP custom header fields specified in clause 5.2.3.2 of 3GPP TS 29.500 [4] shall be applicable.

6.1.2.4 HTTP multipart messages

6.1.2.4.1 General

HTTP multipart messages shall be supported to transfer the opaque Record and Block Information in the following service operations (and HTTP messages):

- Record (PUT/GET/DELETE)
- BlockCollection (GET)
- RecordNotification (POST)

6.1.2.4.2 Record

The Record is encoded as an HTTP multipart message with the multipart/mixed content-type as described in IETF RFC 2046 [21].

The boundary parameter is used to delimit each part (Start of parts to RecordMeta, RecordMeta to Block, Block to Block and Block to End of parts) and shall be set to a value as in accordance with IETF RFC 2046 [21].

The RecordMeta part shall be the first part, and shall always be present. It shall include a Content-ID header set to a "world-unique" value (see IETF RFC 2045 [20]), the Content-Type (see IETF RFC 2045 [20]) header set to "application/json" and include JSON content for the RecordMeta.

Zero or more Block parts may follow the RecordMeta Part. For each Block part in the HTTP multipart/mixed message, the Block part shall include a Content-ID header identifying the Block with the value of the blockId, which shall be a "world-unique" value (see IETF RFC 2045 [20]), a Content-Type (see IETF RFC 2045 [20]) header indicating the MIME type of the Block (any value), e.g. application/octet-stream, application/json or another applicable MIME type and the Content-Transfer-Encoding (see IETF RFC 2045 [20]) header with an appropriate value.

6.1.2.4.3 BlockCollection

The Record is encoded as an HTTP multipart message with the multipart/parallel content-type as described in IETF RFC 2046 [21].

The boundary parameter is used to delimit each part (Start of parts to Block, Block to Block and Block to End of parts) and shall be set to a value as in accordance with IETF RFC 2046 [21].

For the BlockCollection, zero or more Block parts may be included. For each Block part in the HTTP multipart/parallel message, the Block part shall include a Content-ID header identifying the Block with the value of the blockId, which shall be a "world-unique" value (see IETF RFC 2045 [20]), a Content-Type header indicating the MIME type of the Block (any value) e.g. application/octet-stream, application/json or another applicable MIME type and the Content-Transfer-Encoding (see IETF RFC 2045 [20]) header with an appropriate value.

6.1.2.4.4 RecordNotification

The RecordNotification is encoded as an HTTP multipart message with the multipart/mixed content-type as described in IETF RFC 2046 [21].

The boundary parameter is used to delimit each part (Start of parts to RecordNotification, RecordNotification to RecordMeta, RecordMeta to Block, Block to Block and Block to End of parts) and shall be set to a value in accordance with IETF RFC 2046 [21].

The RecordNotification shall be the first part, and shall always be present. It shall include a Content-ID header set to a "world-unique" value (see IETF RFC 2045 [20]), the Content-Type (see IETF RFC 2045 [20]) header set to "application/json" and include JSON content for the NotificationDescription.

The RecordMeta part shall be the second part, and shall always be present. It shall include a Content-ID header set to a "world-unique" value (see IETF RFC 2045 [20]), the Content-Type (see IETF RFC 2045 [20]) header set to "application/json" and include JSON content for the RecordMeta.

Zero or more Block parts may follow the RecordMeta Part. For each Block part in the HTTP multipart/mixed message, the Block part shall include a Content-ID header identifying the Block with the value of the blockId, which shall be a "world-unique" value (see IETF RFC 2045 [20]), a Content-Type (see IETF RFC 2045 [20]) header indicating the MIME type of the Block (any value), e.g. application/octet-stream, application/json or another applicable MIME type and the Content-Transfer-Encoding (see IETF RFC 2045 [20]) header with an appropriate value.

6.1.3 Resources

6.1.3.1 Overview

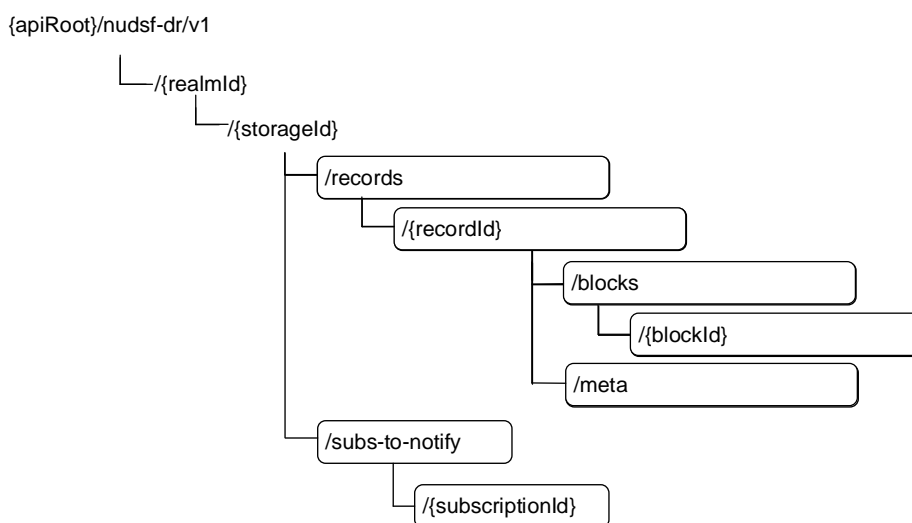


Figure 6.1.3.1-1: Resource URI structure of the nudsf-dr API

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

6.1.3.2 Resource: RecordCollection (Collection)

6.1.3.2.1 Description

This resource represents the collection of records within a storage. It can be used to search for specific records matching specific filter criteria.

6.1.3.2.2 Resource Definition

Resource URI: **{apiRoot}/nudsf-dr/<apiVersion>/{realmId}/{storageId}/records**

This resource shall support the resource URI variables defined in table 6.1.3.2.2-1.

Table 6.1.3.2.2-1: Resource URI variables for this resource

Name	Definition
apiRoot	See clause 6.1.1
realmId	Represents the realm Id.
storageId	Represents the storage Id.

6.1.3.2.3 Resource Standard Methods

6.1.3.2.3.1 GET

This method shall support the URI query parameters specified in table 6.1.3.2.3.1-1.

Table 6.1.3.2.3.1-1: 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	
filter	SearchExpression	O	0..1	The filter criteria for searching the records of the storage.	
limit-range	UInteger	C	0..1	When set, the returned response shall contain at the most the number of record references specified by the parameter value. If the count-indicator parameter is set in the request, this parameter shall be ignored.	
count-indicator	boolean	O	0..1	If this parameter is set, the number of records that matched the criteria shall be returned and no record references shall be returned.	

This method shall support the request data structures specified in table 6.1.3.2.3.1-2 and the response data structures and response codes specified in table 6.1.3.2.3.1-3.

Table 6.1.3.2.3.1-2: Data structures supported by the GET Request Body on this resource

Data type	P	Cardinality	Description
n/a			

Table 6.1.3.2.3.1-3: Data structures supported by the GET Response Body on this resource

Data type	P	Cardinality	Response codes	Description
RecordSearchResult	M	1	200 OK	The search result containing the record references matching the filter.
n/a			204 No Content	The search did not result in any matching record references.
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.				

6.1.3.3 Resource: Record (Document)

6.1.3.3.1 Description

This resource represents a record within a storage.

6.1.3.3.2 Resource Definition

Resource URI: {apiRoot}/nudsf-dr/<apiVersion>/{realmId}/{storageId}/records/{recordId}

This resource shall support the resource URI variables defined in table 6.1.3.3.2-1.

Table 6.1.3.3.2-1: Resource URI variables for this resource

Name	Definition
apiRoot	See clause 6.1.1
realmId	Represents the realm Id where the record is stored
storageId	Represents the storage Id where the record is stored
recordId	Represents the record Id of the record

6.1.3.3.3 Resource Standard Methods

6.1.3.3.3.1 GET

This method shall support the URI query parameters specified in table 6.1.3.3.3.1-1.

Table 6.1.3.3.3.1-1: 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.	

This method shall support the request data structures specified in table 6.1.3.3.3.1-2 and the response data structures and response codes specified in table 6.1.3.3.3.1-3.

Table 6.1.3.3.3.1-2: Data structures supported by the GET Request Body on this resource

Data type	P	Cardinality	Description
n/a			

Table 6.1.3.3.3.1-3: Data structures supported by the GET Response Body on this resource

Data type	P	Cardinality	Response codes	Description
RecordBody	M	1	200 OK	A response body containing the record.
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 -RECORD_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.				

6.1.3.3.3.2 PUT

This method shall support the URI query parameters specified in table 6.1.3.3.3.2-1.

Table 6.1.3.3.3.2-1: 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	
get-previous	boolean	O	0..1	Request to return the previous record content if a record already exists in the targeted storage for the same record identifier.	

When creating or replacing the record, meta and zero or more blocks shall be included. The record meta information shall be the first part and is mandatory. The remaining parts of the body (if any) shall be the blocks to be updated or created. See clause 6.1.2.4 for details on the encoding.

If the operation updates an existing record, then the existing record and its blocks shall be discarded and replaced by the meta and blocks supplied with the request. This also applies to the case when no new blocks are included, i.e. the old blocks (if any) shall be deleted from the record.

This method shall support the request data structures specified in table 6.1.3.3.3.2-2 and the response data structures and response codes specified in table 6.1.3.3.3.2-3.

Table 6.1.3.3.3.2-2: Data structures supported by the PUT Request Body on this resource

Data type	P	Cardinality	Description
Record	M	1	The record that is to be created including meta and zero or more blocks.

Table 6.1.3.3.3.2-3: Data structures supported by the PUT Response Body on this resource

Data type	P	Cardinality	Response codes	Description
RecordBody	M	1	200 OK	Upon successful update of a record, a response body containing the previous record value (if get-previous was indicated in the request, and if one exists) will be returned
RecordBody	O	0..1	201 Created	Upon successful creation of a record, a response body of the created record shall be returned. If due to operator's policy the value of the ttl (if any) in the request exceeded a maximum allowed ttl value with the ttl set to the value applied by the UDSF.
n/a			204 No Content	Upon successful update of a record, an empty response is returned or if no previous record value was requested.
ProblemDetails	O	0..1	403 FORBIDDEN	If the UDSF (based on operator policy), determines to apply a ttl value of the recordMeta different from the ttl value (if any) of the request, the get-previous query-parameter is present in the request and the request applies to a record that already exists in the UDSF, the "cause" attribute shall be set to one of the following application errors: -TTL_VALUE_NOT_ALLOWED
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
RecordBody	O	0..1	412 Precondition Failed	If one or more conditions given in the request header fields evaluated to false and get-previous was indicated in the request, the UDSF shall include the RecordBody in the response.
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.				

6.1.3.3.3.3

DELETE

This method shall support the URI query parameters specified in table 6.1.3.3.3.3-1.

Table 6.1.3.3.3-1: 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	
get-previous	boolean	O	0..1	Request to return the record content if a record exists in the targeted storage for the same record identifier.	

This method shall support the request data structures specified in table 6.1.3.3.3-2 and the response data structures and response codes specified in table 6.1.3.3.3-3.

Table 6.1.3.3.3-2: Data structures supported by the DELETE Request Body on this resource

Data type	P	Cardinality	Description
n/a			

Table 6.1.3.3.3-3: Data structures supported by the DELETE Response Body on this resource

Data type	P	Cardinality	Response codes	Description
RecordBody	O	0..1	200 OK	Upon success, and a response body containing the record value with meta and associated blocks (if any), if get-previous was indicated in the request.
n/a			204 No Content	Upon success and no response body containing the record was requested.
RecordBody	O	0..1	412 Precondition Failed	If one or more conditions given in the request header fields evaluated to false and get-previous was indicated in the request, the UDSF shall include the RecordBody in the response.
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 -RECORD_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.				

6.1.3.4 Resource: Meta (Document)

6.1.3.4.1 Description

This resource represents the meta associated with a record.

6.1.3.4.2 Resource Definition

Resource URI: **{apiRoot}/nudsf-dr/<apiVersion>/{realmId}/{storageId}/records/{recordId}/meta**

This resource shall support the resource URI variables defined in table 6.1.3.4.2-1.

Table 6.1.3.4.2-1: Resource URI variables for this resource

Name	Definition
apiRoot	See clause 6.1.1
realmId	Represents the realm Id where the record is stored
storageId	Represents the storage Id where the record is stored
recordId	Represents the record Id of the record

6.1.3.4.3 Resource Standard Methods

6.1.3.4.3.1 GET

This method shall support the URI query parameters specified in table 6.1.3.4.3.1-1.

Table 6.1.3.4.3.1-1: 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	

This method shall support the request data structures specified in table 6.1.3.4.3.1-2 and the response data structures and response codes specified in table 6.1.3.4.3.1-3.

Table 6.1.3.4.3.1-2: Data structures supported by the GET Request Body on this resource

Data type	P	Cardinality	Description
n/a			

Table 6.1.3.4.3.1-3: Data structures supported by the GET Response Body on this resource

Data type	P	Cardinality	Response codes	Description
RecordMeta	M	1	200 OK	A response body containing the record meta will be returned.
ProblemDetails	O	0..1	404 Not Found	The "cause" attribute shall be set to one of the following application errors: -REALM_NOT_FOUND -STORAGE_NOT_FOUND -RECORD_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.				

6.1.3.4.3.2 PATCH

This method shall support the URI query parameters specified in table 6.1.3.4.3.2-1.

Table 6.1.3.4.3.2-1: URI query parameters supported by the PATCH 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	

This method shall support the request data structures specified in table 6.1.3.4.3.2-2 and the response data structures and response codes specified in table 6.1.3.4.3.2-3.

Table 6.1.3.4.3.2-2: Data structures supported by the PATCH Request Body on this resource

Data type	P	Cardinality	Description
array(patchItem)	M	1..N	A collection of patch items to apply on the record meta.

Table 6.1.3.4.3.2-3: Data structures supported by the PATCH Response Body on this resource

Data type	P	Cardinality	Response codes	Description
n/a			204 No Content	Upon successful modification, there is no body in the response message. (NOTE 2)
PatchResult	M	1	200 OK	If one or more modification instructions have been discarded, the execution report is returned. (NOTE 2)
ProblemDetails	O	0..1	404 Not Found	The "cause" attribute maybe used to indicate one of the following application errors: -REALM_NOT_FOUND -STORAGE_NOT_FOUND -RECORD_NOT_FOUND
NOTE 1: In addition common data structures as listed in table 6.1.7.3-1 are supported.				
NOTE 2: If all the modification instructions in the PATCH request have been implemented, the UDSF shall respond with 204 No Content response; if some of the modification instructions in the PATCH request have been discarded, the UDSF shall respond with PatchResult.				

6.1.3.5 Resource: BlockCollection (Collection)

6.1.3.5.1 Description

This resource represents the collection of blocks of associated with a record.

6.1.3.5.2 Resource Definition

Resource URI: {apiRoot}/nudsf-dr/<apiVersion>/{realmId}/{storageId}/records/{recordId}/blocks

This resource shall support the resource URI variables defined in table 6.1.3.5.2-1.

Table 6.1.3.5.2-1: Resource URI variables for this resource

Name	Definition
apiRoot	See clause 6.1.1
realmId	Represents the realm Id where the record is stored
storageId	Represents the storage Id where the record is stored
recordId	Represents the record Id of the record

6.1.3.5.3 Resource Standard Methods

6.1.3.5.3.1 GET

This method shall support the URI query parameters specified in table 6.1.3.5.3.1-1.

Table 6.1.3.5.3.1-1: 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	

This method shall support the request data structures specified in table 6.1.3.5.3.1-2 and the response data structures and response codes specified in table 6.1.3.5.3.1-3.

Table 6.1.3.5.3.1-2: Data structures supported by the GET Request Body on this resource

Data type	P	Cardinality	Description
n/a			

Table 6.1.3.5.3.1-3: Data structures supported by the GET Response Body on this resource

Data type	P	Cardinality	Response codes	Description
array(Block)	M	1..N	200 OK	A response body containing one or more blocks.
n/a			204 No Content	The BlockCollection did not contain any blocks.
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.				

6.1.3.6 Resource: Block (Document)

6.1.3.6.1 Description

This resource represents a record within a storage.

6.1.3.6.2 Resource Definition

Resource URI: {apiRoot}/nudsf-dr/<apiVersion>/{realmId}/{storageId}/records/{recordId}/blocks/{blockId}

This resource shall support the resource URI variables defined in table 6.1.3.6.2-1.

Table 6.1.3.6.2-1: Resource URI variables for this resource

Name	Definition
apiRoot	See clause 6.1.1
realmId	Represents the realm Id where the record is stored
storageId	Represents the storage Id where the record is stored
recordId	Represents the record Id of the record
blockId	Represents the block Id of the block

6.1.3.6.3 Resource Standard Methods

6.1.3.6.3.1 GET

This method shall support the URI query parameters specified in table 6.1.3.6.3.1-1.

Table 6.1.3.6.3.1-1: 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	

This method shall support the request data structures specified in table 6.1.3.6.3.1-2 and the response data structures and response codes specified in table 6.1.3.6.3.1-3.

Table 6.1.3.6.3.1-2: Data structures supported by the GET Request Body on this resource

Data type	P	Cardinality	Description
n/a			

Table 6.1.3.6.3.1-3: Data structures supported by the GET Response Body on this resource

Data type	P	Cardinality	Response codes	Description
BlockBody	M	1	200 OK	Upon success, a response body containing the requested block is returned.
ProblemDetails	O	0..1	404 Not Found	The "cause" attribute maybe used to indicate one of the following application errors: -REALM_NOT_FOUND -STORAGE_NOT_FOUND -RECORD_NOT_FOUND -BLOCK_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.				

6.1.3.6.3.2 PUT

This method shall support the URI query parameters specified in table 6.1.3.6.3.2-1.

Table 6.1.3.6.3.2-1: 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	
get-previous	boolean	O	0..1	Request to return the previous block content if a block already exists in the targeted storage for the same block identifier.	

If the operation updates an existing block, then the existing block shall be discarded and replaced by the block supplied with the request.

This method shall support the request data structures specified in table 6.1.3.6.3.2-2 and the response data structures and response codes specified in table 6.1.3.6.3.2-3.

Table 6.1.3.6.3.2-2: Data structures supported by the PUT Request Body on this resource

Data type	P	Cardinality	Description
Block	M	1	The block definition that shall be created. A Content-Type http header can be set to specify the media type of the block's opaque content. If the media-type is not included, the media-type shall be set to application/octet-stream.

Table 6.1.3.6.3.2-3: Data structures supported by the PUT Response Body on this resource

Data type	P	Cardinality	Response codes	Description
BlockBody	M	1	200 OK	Upon successful update of a block, a response body containing the previous record value ((if get-previous was indicated in the request and if one exists) will be returned
n/a			201 Created	Upon successful creation of a record, an empty response shall be returned.
n/a			204 No Content	Upon successful update of a record, an empty response is returned if no previous record 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 -RECORD_NOT_FOUND
BlockBody	O	0..1	412 Precondition Failed	If one or more conditions given in the request header fields evaluated to false and get-previous was indicated in the request, the UDSF shall include the BlockBody in the response.
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.				

6.1.3.6.3.3 DELETE

This method shall support the URI query parameters specified in table 6.1.3.6.3.3-1.

Table 6.1.3.6.3.3-1: 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	
get-previous	boolean	O	0..1	Request to return the previous block content (if any).	

This method shall support the request data structures specified in table 6.1.3.6.3.3-2 and the response data structures and response codes specified in table 6.1.3.6.3.3-3.

Table 6.1.3.6.3.3-2: Data structures supported by the DELETE Request Body on this resource

Data type	P	Cardinality	Description
n/a			

Table 6.1.3.6.3.3-3: Data structures supported by the DELETE Response Body on this resource

Data type	P	Cardinality	Response codes	Description
BlockBody	O	0..1	200 OK	Upon success, and a response body containing the block was requested.
n/a			204 No Content	Upon success, and no response body containing the block was requested.
BlockBody	O	0..1	412 Precondition Failed	If one or more conditions given in the request header fields evaluated to false and get-previous was indicated in the request, the UDSF shall include the BlockBody in the response.
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 -RECORD_NOT_FOUND -BLOCK_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.				

6.1.3.7 Resource: NotificationSubscriptions

6.1.3.7.1 Description

This resource is used to represent notification subscriptions.

6.1.3.7.2 Resource Definition

Resource URI: **{apiRoot}/nudsf-dr/<apiVersion>/{realmId}/{storageId}/subs-to-notify**

This resource shall support the resource URI variables defined in table 6.1.3.7.2-1.

Table 6.1.3.7.2-1: Resource URI variables for this resource

Name	Definition
apiRoot	See clause 6.1.1
realmId	Represents the realm Id.
storageId	Represents the storage Id.

6.1.3.7.3 Standard Methods

6.1.3.7.3.2 GET

This method shall support the URI query parameters specified in table 6.1.3.7.3.2-1.

Table 6.1.3.7.3.2-1: URI query parameters supported by the GET method on this resource

Name	Data type	P	Cardinality	Description
supported-features	SupportedFeatures	O	0..1	see 3GPP TS 29.500 [4] clause 6.6
limit-range	UInteger	C	0..1	When set, the returned response shall contain at the most the number of Notification Subscriptions specified by the parameter value.

This method shall support the request data structures specified in table 6.1.3.7.3.2-2 and the response data structures and response codes specified in table 6.1.3.7.3.2-3.

Table 6.1.3.7.3.2-2: Data structures supported by the GET Request Body on this resource

Data type	P	Cardinality	Description
n/a			

Table 6.1.3.7.3.2-3: Data structures supported by the GET Response Body on this resource

Data type	P	Cardinality	Response codes	Description
array(NotificationSubscription)	M	0..N	200 OK	Upon success, a response body containing the individual subscriptions shall be returned.
ProblemDetails	O	0..1	404 NOT FOUND	The "cause" attribute shall be set to one of the following application errors: -REALM_NOT_FOUND -STORAGE_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.				

6.1.3.8 Resource: IndividualNotificationSubscription

6.1.3.8.1 Description

This resource is used to represent an individual subscriber data subscriptions to notifications.

6.1.3.8.2 Resource Definition

Resource URI: **{apiRoot}/nudsf-dr/<apiVersion>/{realmId}/{storageId}/subs-to-notify/{subscriptionId}**

This resource shall support the resource URI variables defined in table 6.1.3.8.2-1.

Table 6.1.3.8.2-1: Resource URI variables for this resource

Name	Definition
apiRoot	See clause 6.1.1
realmId	Represents the realm Id.
storageId	Represents the storage Id.
subscriptionId	The subscriptionId identifies an individual NotificationSubscription.

6.1.3.8.3 Resource Standard Methods

6.1.3.8.3.1 DELETE

This method shall support the URI query parameters specified in table 6.1.3.8.3.1-1.

Table 6.1.3.8.3.1-1: URI query parameters supported by the DELETE method on this resource

Name	Data type	P	Cardinality	Description
client-id	ClientId	M	1	The client-id is used by the UDSF to guard against deletion of notification subscriptions that do not belong to the same NF or NFSet.
supported-features	SupportedFeatures	O	0..1	see 3GPP TS 29.500 [4] clause 6.6
get-previous	boolean	O	0..1	Request to return the associated NotificationSubscription if it exist.

This method shall support the request data structures specified in table 6.1.3.8.3.1-2 and the response data structures and response codes specified in table 6.1.3.8.3.1-3.

Table 6.1.3.8.3.1-2: Data structures supported by the DELETE Request Body on this resource

Data type	P	Cardinality	Description
n/a			The request body shall be empty.

Table 6.1.3.8.1-3: Data structures supported by the DELETE Response Body on this resource

Data type	P	Cardinality	Response codes	Description
NotificationSubscription	O	0..1	200 OK	Upon successful delete of a NotificationSubscription, a response body containing the NotificationSubscription value before the delete shall be returned if get-previous was indicated in the request.
n/a			204 NO CONTENT	Upon success, an empty response body shall be returned.
ProblemDetails	O	0..1	403 FORBIDDEN	If the client-id query parameter does not match the clientId of the existing resource, 403 FORBIDDEN shall be returned and the "cause" attribute may be used to indicate one of the following application errors: - SUBSCRIPTION_EXISTS
ProblemDetails	O	0..1	404 NOT FOUND	The "cause" attribute shall be set to one of the following application errors: -REALM_NOT_FOUND -STORAGE_NOT_FOUND -SUBSCRIPTION_NOT_FOUND
NotificationSubscription	O	0..1	412 PRECONDITION FAILED	412 PRECONDITION FAILED is returned if one or more conditions given in the request header fields evaluated to false. If get-previous was indicated in the request, the UDSF shall include the NotificationSubscription in the response.
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.				

6.1.3.8.3.2 PATCH

This method shall support the URI query parameters specified in table 6.1.3.8.3.2-1.

Table 6.1.3.8.3.2-1: URI query parameters supported by the PATCH method on this resource

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

This method shall support the request data structures specified in table 6.1.3.8.3.2-2 and the response data structures and response codes specified in table 6.1.3.8.3.2-3.

Table 6.1.3.8.3.2-2: Data structures supported by the PATCH Request Body on this resource

Data type	P	Cardinality	Description
array(PatchItem)	M	1..N	Contains the delta data of the Notification Subscription to be updated.

Table 6.1.3.8.3.2-3: Data structures supported by the PATCH Response Body on this resource

Data type	P	Cardinality	Response codes	Description
n/a			204 NO CONTENT	Upon successful modification, there is no body in the response message. (NOTE 2)
PatchResult	M	1	200 OK	Upon success, the execution report is returned. (NOTE 2)
ProblemDetails	O	0..1	404 NOT FOUND	The "cause" attribute shall be set to one of the following application errors: -REALM_NOT_FOUND -STORAGE_NOT_FOUND -SUBSCRIPTION_NOT_FOUND

NOTE 1: In addition common data structures as listed in table 6.1.7.3-1 are supported.
NOTE 2: If all the modification instructions in the PATCH request have been implemented, the UDSF shall respond with 204 No Content response; if some of the modification instructions in the PATCH request have been discarded, the UDSF shall respond with 200 OK and include the PatchResult.

6.1.3.8.3.3 GET

This method shall support the URI query parameters specified in table 6.1.3.8.3.3-1.

Table 6.1.3.8.3.3-1: URI query parameters supported by the GET method on this resource

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

This method shall support the request data structures specified in table 6.1.3.8.3.3-2 and the response data structures and response codes specified in table 6.1.3.8.3.3-3.

Table 6.1.3.8.3.3-2: Data structures supported by the GET Request Body on this resource

Data type	P	Cardinality	Description
n/a			The request body shall be empty.

Table 6.1.3.8.3.3-3: Data structures supported by the GET Response Body on this resource

Data type	P	Cardinality	Response codes	Description
NotificationSubscription	M	1	200 OK	Upon success, a response body containing the individual subscriptions shall be returned.
ProblemDetails	O	0..1	404 NOT FOUND	The "cause" attribute shall be set to one of the following application errors: -REALM_NOT_FOUND -STORAGE_NOT_FOUND -SUBSCRIPTION_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.

6.1.3.8.3.4 PUT

This method shall support the URI query parameters specified in table 6.1.3.8.3.4-1.

Table 6.1.3.8.3.4-1: 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	

This method shall support the request data structures specified in table 6.1.3.8.3.4-2 and the response data structures and response codes specified in table 6.1.3.8.3.4-3.

Table 6.1.3.8.3.4-2: Data structures supported by the PUT Request Body on this resource

Data type	P	Cardinality	Description
NotificationSubscription	M	1	The Notification Subscription.

Table 6.1.3.8.3.4-3: Data structures supported by the PUT Response Body on this resource

Data type	P	Cardinality	Response codes	Description
NotificationSubscription	M	1	201 CREATED	Upon success, the created NotificationSubscription is returned. The HTTP response shall include a "Location" HTTP header that contains the resource URI of the created resource.
NotificationSubscription	M	1	200 OK	Upon success, the updated NotificationSubscription is returned.
ProblemDetails	O	0..1	404 NOT FOUND	The "cause" attribute shall be set to one of the following application errors: -REALM_NOT_FOUND -STORAGE_NOT_FOUND
ProblemDetails	O	0..1	403 FORBIDDEN	If the clientId of the PUT request does not match the clientId of the existing resource, 403 FORBIDDEN shall be returned and the "cause" attribute may be used to indicate one of the following application errors: - SUBSCRIPTION_EXISTS
array(Uri)	O	1..N	409 CONFLICT	If one or more monitoredResourceUris from the request don't exist in the UDSF, 409 CONFLICT shall be returned together with the non-existing monitoredResourceUris.
NOTE: The mandatory HTTP error status codes for the POST method listed in Table 5.2.7.1-1 of 3GPP TS 29.500 [4] also apply.				

6.1.4 Custom Operations without associated resources

None.

6.1.5 Notifications

6.1.5.1 General

Notifications shall comply to clause 6.2 of 3GPP TS 29.500 [4] and clause 4.6.2.3 of 3GPP TS 29.501 [5].

6.1.5.2 Timer Expiry Notification

6.1.5.2.1 Description

The Timer Expiry Notification is used by the NF service producer to report to an NF Consumer that the Record has expired as indicated by the ttl attribute (if set) of RecordMeta and if a callbackReference was set in the RecordMeta.

6.1.5.2.2 Target URI

The Callback URI "{callbackReference}" shall be used with the callback URI variables defined in table 6.1.5.2.2-1.

Table 6.1.5.2.2-1: Target URI variables for this resource

Name	Definition
callbackReference	string formatted as URI with the Callback Uri

6.1.5.2.3 Standard Methods

6.1.5.2.3.1 POST

This method shall support the request data structures specified in table 6.1.5.2.3.1-1 and the response data structures and response codes specified in table 6.1.5.2.3.1-1.

Table 6.1.5.2.3.1-2: Data structures supported by the POST Request Body on this resource

Data type	P	Cardinality	Description
RecordBody	M	1	The RecordBody of the record that was deleted.

Table 6.1.5.2.3.1-3: Data structures supported by the POST Response Body on this resource

Data type	P	Cardinality	Response codes	Description
n/a			204 No Content	Upon success, an empty response body shall be returned.
NOTE: The mandatory HTTP error status codes for the POST method listed in Table 5.2.7.1-1 of 3GPP TS 29.500 [4] also apply.				

6.1.5.3 Notification due to Data Change

6.1.5.3.1 Description

The Notification due to Data Change is used by the UDSF to report to an NF Consumer that a Record that is part of a NotificationSubscription has been updated or deleted.

6.1.5.3.2 Target URI

The Callback URI "{callbackReference}" shall be used with the callback URI variables defined in table 6.1.5.3.2-1.

Table 6.1.5.3.2-1: Callback URI variables for this resource

Name	Definition
callbackReference	String formatted as URI with the Callback Uri

6.1.5.3.3 Standard Methods

6.1.5.3.3.1 POST

This method shall support the request data structures specified in table 6.1.5.3.3.1-1 and the response data structures and response codes specified in table 6.1.5.3.3.1-1.

Table 6.1.5.3.3.1-2: Data structures supported by the POST Request Body on this resource

Data type	P	Cardinality	Description
RecordNotification	M	1	The notification with the record information.

Table 6.1.5.3.3.1-3: Data structures supported by the POST Response Body on this resource

Data type	P	Cardinality	Response codes	Description
n/a			204 NO CONTENT	Upon success, an empty response body shall be returned.
NOTE: The mandatory HTTP error status codes for the POST method listed in Table 5.2.7.1-1 of 3GPP TS 29.500 [4] also apply.				

6.1.6 Data Model

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 N_{udsf} 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: N_{udsf} 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	
RecordOperation	6.1.6.3.5	Record Operation	

Table 6.1.6.1-2 specifies data types re-used by the N_{udsf} 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 N_{udsf} service based interface.

Table 6.1.6.1-2: N_{udsf} 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]		

6.1.6.2 Structured data types

6.1.6.2.1 Introduction

This clause defines the structures to be used in resource representations.

6.1.6.2.2 Type: RecordSearchResult

Table 6.1.6.2.2-1: Definition of type RecordSearchResult

Attribute name	Data type	P	Cardinality	Description	Applicability
count	UInteger	M	1	The number of records found by the search.	
references	array(Uri)	O	1..N	The Record references found by the search.	
supportedFeatures	SupportedFeatures	O	0..1	See clause 6.1.8	

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.	

6.1.6.2.4 Type: RecordBody

Table 6.1.6.2.4-1: Definition of type RecordBody

Attribute name	Data type	P	Cardinality	Description	Applicability
record	Record	M	1	The record.	

6.1.6.2.5 Type: Record

Table 6.1.6.2.5-1: Definition of type Record

Attribute name	Data type	P	Cardinality	Description	Applicability
meta	RecordMeta	M	1	The meta of a record.	
blocks	array(Block)	O	1..N	The block(s) (if any) making up the record.	

6.1.6.2.6 Type: BlockBody

Table 6.1.6.2.6-1: Definition of type BlockBody

Attribute name	Data type	P	Cardinality	Description	Applicability
n/a	Block	M	1	The block.	

6.1.6.2.7 Type: Block

Table 6.1.6.2.7-1: Definition of type Block

Attribute name	Data type	P	Cardinality	Description	Applicability
value	Any Type	M	1	The block value of any data type.	

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.	

6.1.6.2.9 SearchComparison

Table 6.1.6.2.9-1: Definition of type SearchComparison

Attribute name	Data type	P	Cardinality	Description	Applicability
op	ComparisonOperator	M	1	Comparison operator	
tag	string	M	1	This attribute contains the tag name of an array of strings.	
value	string	M	1	The array of strings indicated in the tag attribute compares to the value of this attribute.	

6.1.6.2.10 Type: NotificationSubscription

Table 6.1.6.2.10-1: Definition of type NotificationSubscription

Attribute name	Data type	P	Cardinality	Description	Applicability
clientId	ClientId	M	1	Identity of the NF or NFSet for which the subscription applies.	
callbackReference	Uri	M	1	Identifies the NF or NF pool where the notification shall be sent.	
expiry	DateTime	C	0..1	This IE shall be included in a subscription response, if, based on operator policy and taking into account the expiry time included in the request, the UDSF needs to include an expiry time. The expiry time, based on operator policy, may indicate a value that is sooner than the NF consumer requested. The absence of this attribute in the subscription response indicates that the subscription does not have an expiry time.	
subFilter	SubscriptionFilter	O	0..1	If not included, the subscription applies to all Create, Delete and Update events of the storage.	
supportedFeatures	SupportedFeatures	O	0..1	Used to negotiate the applicability of optional features	

6.1.6.2.11 Type: RecordNotification

Table 6.1.6.2.11-1: Definition of type RecordNotification

Attribute name	Data type	P	Cardinality	Description	Applicability
descriptor	NotificationDescription	M	1	The block value of any data type.	
meta	RecordMeta	M	1	The meta of a record.	
blocks	array(Block)	O	1..n	The block(s) (if any) making up the record.	

6.1.6.2.12 Type: NotificationDescription

Table 6.1.6.2.12-1: Definition of type NotificationDescription

Attribute name	Data type	P	Cardinality	Description	Applicability
recordRef	Uri	M	1	The reference of the record triggering the Notification.	
operationType	RecordOperation	M	1	The operation type.	
subscriptionId	string	O	0..1	This IE shall contain the subscriptionId that uniquely identifies the subscription to notification within a storage when present.	

6.1.6.2.13 Type: SubscriptionFilter

Table 6.1.6.2.13-1: Definition of type SubscriptionFilter

Attribute name	Data type	P	Cardinality	Description	Applicability
monitoredResourceUris	array(Uri)	O	1..N	A set of URIs that identify the records for which a modification of the representation triggers a notification. The URI shall take the form of either an absolute URI or an absolute-path reference as defined in IETF RFC 3986 [22]. The monitored resource shall indicate an existing record. If not present, the subscription is applied to all records within the storage and a modification of the representation of any record within the storage triggers a notification.	
operations	array(RecordOperation)	O	0..3	The operations that shall generate a notification. If the monitoredResourceUris is present, only "UPDATED" and "DELETED" are allowed values, any other value shall be ignored. If the attribute is not present, all applicable operations shall apply to the subscription.	

6.1.6.2.14 Type: ClientId

Table 6.1.6.2.14-1: Definition of type ClientId

Attribute name	Data type	P	Cardinality	Description	Applicability
nfId	NfInstanceId	C	0..1	The NF Instance Id uniquely identifying the NF Consumer. Shall be present if the nfSetId is absent.	
nfSetId	NfSetId	C	0..1	The NF Set Id of the NF Consumer. Shall be present if the nfId is absent.	

6.1.6.3 Simple data types and enumerations

6.1.6.3.1 Introduction

This clause defines simple data types and enumerations that can be referenced from data structures defined in the previous clauses.

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

6.1.6.3.3 Enumeration: ComparisonOperator

The enumeration ComparisonOperator represents the comparison of an array of strings to a string value. The comparison shall be based on lexicographical order. It shall comply with the provisions defined in table 6.1.6.3.3-1.

Table 6.1.6.3.3-1: Enumeration ComparisonOperator

Enumeration value	Description	Applicability
"EQ"	The array contains the string value.	
"NEQ"	The array does not contain the string value.	AdvancedQuery
"GT"	The array contains a string that is greater than the string value.	AdvancedQuery
"GTE"	The array contains a string that is greater than or equal to the string value.	AdvancedQuery
"LT"	The array contains a string that is less than the string value.	AdvancedQuery
"LTE"	The array contains a string that is less than or equal to the string value.	AdvancedQuery

NOTE: It's recommended to use GT/GTE/LT/LTE on single value tags. If not, the logical operator "NOT" applied over the comparison operator "GT" evaluates to "true" if **there are no members** in the array that are greater than the value.

6.1.6.3.4 Enumeration: ConditionOperator

Table 6.1.6.3.4-1: Enumeration ConditionOperator

Enumeration value	Description	Applicability
"AND"	Logical "AND"	
"OR"	Logical "OR"	
"NOT"	Logical "NOT"	

6.1.6.3.5 Enumeration: RecordOperation

Table 6.1.6.3.5-1: Enumeration RecordOperation

Enumeration value	Description	Applicability
"CREATED"	Indicates a Create record operation	
"UPDATED"	Indicates an Update record operation	
"DELETED"	Indicates a Delete record operation	

6.1.6.4 Data types describing alternative data types or combinations of data types

6.1.6.4.1 Type: SearchExpression

Table 6.1.6.4.1-1: Definition of type SearchExpression as a list of mutually exclusive alternatives

Data type	Cardinality	Description	Applicability
SearchCondition	1	A search expression with logic operators	AdvancedQuery
SearchComparison	1	A minimum unit of the search expression	

6.1.7 Error Handling

6.1.7.1 General

For the Nudsf_DataRepository API, HTTP error responses shall be supported as specified in clause 4.8 of 3GPP TS 29.501 [5]. Protocol errors and application errors specified in table 5.2.7.2-1 of 3GPP TS 29.500 [4] shall be supported for an HTTP method if the corresponding HTTP status codes are specified as mandatory for that HTTP method in table 5.2.7.1-1 of 3GPP TS 29.500 [4].

In addition, the requirements in the following clauses are applicable for the Nudsf_DataRepository API.

6.1.7.2 Protocol Errors

Protocol errors handling shall be supported as specified in clause 5.2.7 of 3GPP TS 29.500 [4].

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.
SUBSCRIPTION_EXISTS	403 Forbidden	The subscription indicated in the HTTP/2 request is not authorized to be operated.
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.

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	<p>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.</p> <p>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.</p>

6.1.9 Security

As indicated in 3GPP TS 33.501 [8] and 3GPP TS 29.500 [4], the access to the Nudsf_DataRepository API may be authorized by means of the OAuth2 protocol (see IETF RFC 6749 [9]), based on local configuration, using the "Client Credentials" authorization grant, where the NRF (see 3GPP TS 29.510 [10]) plays the role of the authorization server.

If OAuth2 is used, an NF Service Consumer, prior to consuming services offered by the <API Name> API, shall obtain a "token" from the authorization server, by invoking the Access Token Request service, as described in 3GPP TS 29.510 [10], clause 5.4.2.2.

NOTE: When multiple NRFs are deployed in a network, the NRF used as authorization server is the same NRF that the NF Service Consumer used for discovering the Nudsf_DataRepository service.

The Nudsf_DataRepository API defines a single scope "nudsf-dr" for the entire service, and it does not define any additional scopes at resource or operation level.

Annex A (normative): OpenAPI specification

A.1 General

This Annex specifies the formal definition of the API(s) defined in the present specification. It consists of OpenAPI 3.0.0 specifications in YAML format.

This Annex takes precedence when being discrepant to other parts of the specification with respect to the encoding of information elements and methods within the API(s).

NOTE: The semantics and procedures, as well as conditions, e.g. for the applicability and allowed combinations of attributes or values, not expressed in the OpenAPI definitions but defined in other parts of the specification also apply.

Informative copies of the OpenAPI specification files contained in this 3GPP Technical Specification are available on a Git-based repository that uses the GitLab software version control system (see 3GPP TS 29.501 [5] clause 5.3.1 and 3GPP TR 21.900 [7] clause 5B).

A.2 Nudsf_DataRepository API

```

openapi: 3.0.0
info:
  title: Nudsf_DataRepository
  version: 1.0.2
  description: |
    Nudsf Data Repository Service.
    © 2021, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TSDSI, TTA, TTC).
    All rights reserved.

externalDocs:
  description: 3GPP TS 29.598 UDSF Services, V16.4.0.
  url: 'http://www.3gpp.org/ftp/Specs/archive/29_series/29.598/'

servers:
  - url: '{apiRoot}/nudsf-dr/v1'
    variables:
      apiRoot:
        default: https://example.com
        description: apiRoot as defined in clause 4.4 of 3GPP TS 29.501

security:
  - {}
  - oAuth2ClientCredentials:
    - nudsf-dr

paths:
  /{realmId}/{storageId}/records:
    summary: Access to all Records of a Storage
    description: >-
      root of all Records of a Storage
    get:
      summary: Records search with get
      description: Retrieve one or multiple Records based on filter
      operationId: SearchRecord
      tags:
        - Record 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: limit-range
    in: query
    description: The most number of record references to fetch
    schema:
      $ref: 'TS29571_CommonData.yaml#/components/schemas/UInteger'
  - name: filter
    in: query
    description: Query filter using conditions on tags
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/SearchExpression'
  - name: count-indicator
    in: query
    description: Indicates whether the number of records that matched the criteria shall be
returned.
    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: Successful case. Response contains result of the search.
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/RecordSearchResult'
  '204':
    description: >-
      The search condition does not match any Record.
  '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'
  '406':
    $ref: 'TS29571_CommonData.yaml#/components/responses/406'
  '429':
    $ref: 'TS29571_CommonData.yaml#/components/responses/429'
  '500':
    $ref: 'TS29571_CommonData.yaml#/components/responses/500'
  '503':
    $ref: 'TS29571_CommonData.yaml#/components/responses/503'
  default:
    $ref: 'TS29571_CommonData.yaml#/components/responses/default'

/{realmId}/{storageId}/records/{recordId}:
summary: Access to a specific Record, identified by its RecordId
description: >-
  Access to a specific Record
get:
  summary: Record access
  description: retrieve one specific Record
  operationId: GetRecord
  tags:
  - Record 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: recordId
    in: path
    description: Identifier of the Record
    required: true
    schema:
      type: string
      example: 'UserRecordValue000000001'
  - 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 Record
  description: Create or Modify a Record with a user provided RecordId
  operationId: CreateOrModifyRecord
  tags:
    - Record 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: recordId
      in: path
      description: Identifier of the Record
      required: true
      schema:
        type: string
        example: UserRecordValue000000001
    - 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: Record validator for conditional requests, as described in RFC 7232, 3.2
  schema:
    type: string
- name: get-previous
  in: query
  description: Retrieve the Record 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:
  $ref: '#/components/requestBodies/RecordBody'
callbacks:
  recordExpired:
    '{$request.body#/callbackReference}':
      post:
        parameters:
          - name: Content-Location
            in: header
            description: The expired record URI
            schema:
              $ref: 'TS29571_CommonData.yaml#/components/schemas/Uri'
        requestBody:
          $ref: '#/components/requestBodies/RecordBody'
        responses:
          '204':
            description: Callback executed successfully
          '400':
            $ref: 'TS29571_CommonData.yaml#/components/responses/400'
          '401':
            $ref: 'TS29571_CommonData.yaml#/components/responses/401'
          '403':
            $ref: 'TS29571_CommonData.yaml#/components/responses/403'
          '500':
            $ref: 'TS29571_CommonData.yaml#/components/responses/500'
          '503':
            $ref: 'TS29571_CommonData.yaml#/components/responses/503'
          default:
            $ref: 'TS29571_CommonData.yaml#/components/responses/default'
responses:
  '200' : # Update with return
    $ref: '#/components/responses/RecordBody'
  '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': # Return Record value if get-previous=true
  $ref: '#/components/responses/RecordBody'
'413':
  $ref: 'TS29571_CommonData.yaml#/components/responses/413'
'500':
  $ref: 'TS29571_CommonData.yaml#/components/responses/500'
'503':
  $ref: 'TS29571_CommonData.yaml#/components/responses/503'
default:
  $ref: 'TS29571_CommonData.yaml#/components/responses/default'
delete:
  summary: Delete a Record with an user provided RecordId
  operationId: DeleteRecord
  tags:
    - Record 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: recordId
      in: path
      description: Identifier of the Record
      required: true
      schema:
        type: string
        example: UserRecordValue000000001
    - 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 Record 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':
      $ref: '#/components/responses/RecordBodyDelete'
    '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': # Return return value if get-previous=true
  $ref: '#/components/responses/RecordBody'
'500':
  $ref: 'TS29571_CommonData.yaml#/components/responses/500'
'503':
  $ref: 'TS29571_CommonData.yaml#/components/responses/503'
default:
  $ref: 'TS29571_CommonData.yaml#/components/responses/default'

/{realmId}/{storageId}/records/{recordId}/meta:
  summary: Access to the meta of a specific Record, identified by its RecordId
  description: >-
    Access to the meta of a specific Record
  get:
    summary: Record's meta access
    description: retrieve meta of a specific Record
    operationId: GetMeta
    tags:
      - Record 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: recordId
        in: path
        description: Identifier of the Record
        required: true
        schema:
          type: string
          example: 'UserRecordValue000000001'
      - 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':
      description: Expected response to a valid request
      headers:
        Cache-Control:
          $ref: '#/components/headers/Cache-Control'
        ETag:
          $ref: '#/components/headers/ETag'
        Last-Modified:
          $ref: '#/components/headers/Last-Modified'
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/RecordMeta'
    '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'
patch: # patch meta data
  summary: Record's meta update
  description: update meta of a specific Record
  operationId: UpdateMeta
  tags:
  - Record 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: recordId
    in: path
    description: Identifier of the Record
    required: true
    schema:
      type: string
      example: 'UserRecordValue000000001'
  - name: If-Match
    in: header
    description: Record validator for conditional requests, as described in RFC 7232, 3.2
    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'
  requestBody:
    description: Meta data to patch
    content:
      application/json-patch+json:
        example: ' [{ "op": "replace", "path": "/tags/ueId", "value": "450005" }, { "op":
"remove", "path": "/tags/recordId" } ]'
        schema:
          type: array
          items:
            $ref: 'TS29571_CommonData.yaml#/components/schemas/PatchItem'
          minItems: 1
    required: true
  responses:
    '200':
      description: >-
        One or more modification instructions have been discarded, the execution report is
returned in response PatchResult.
      content:
        application/json:
          example:
            schema:
              $ref: 'TS29571_CommonData.yaml#/components/schemas/PatchResult'
  headers:
    Cache-Control:
      $ref: '#/components/headers/Cache-Control'
    ETag:

```

```

    $ref: '#/components/headers/ETag'
  Last-Modified:
    $ref: '#/components/headers/Last-Modified'
'204':
  description: >-
    Successful case. The meta has been successfully updated and no return is expected.
  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'
'500':
  $ref: 'TS29571_CommonData.yaml#/components/responses/500'
'503':
  $ref: 'TS29571_CommonData.yaml#/components/responses/503'
default:
  $ref: 'TS29571_CommonData.yaml#/components/responses/default'

```

```

/{realmId}/{storageId}/records/{recordId}/blocks:
  summary: Access to the Blocks of a specific Record, identified by its RecordId
  description: >-
    Access to the Blocks of a specific Record
  get:
    summary: Record's Blocks access
    description: retrieve all Blocks of a specific Record
    operationId: GetBlockList
    tags:
      - Block 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: recordId
        in: path
        description: Identifier of the Record
        required: true
        schema:
          type: string
          example: 'UserRecordValue000000001'
      - 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: Expected response to a successful request
      headers:
        Cache-Control:
          $ref: '#/components/headers/Cache-Control'
        ETag:
          $ref: '#/components/headers/ETag'
        Last-Modified:

```

```

    $ref: '#/components/headers/Last-Modified'
  content:
    multipart/parallel:
      schema:
        type: object
        properties:
          blocks:
            type: array
            description: >-
              an array of Block parts, can be empty
            items:
              $ref: '#/components/schemas/Block'
        encoding:
          blocks:
            contentType: '*/*' # Block content type can be of any type.
          headers:
            Content-ID: # Block identifier is defined by the Content-Id header.
              schema:
                type: string
                required: true
            Content-Transfer-Encoding:
              schema:
                type: string
                required: true
'204':
  description: Successful response, the record contains no blocks
'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'

/{realmId}/{storageId}/records/{recordId}/blocks/{blockId}:
summary: Access to a Block of a specific Record, identified by its BlockId
description: >-
  Access to a specific Block of a specific Record
get:
  summary: Retrieve a specific Block
  description: retrieve a specific Block
  operationId: GetBlock
  tags:
    - Block 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: recordId
      in: path
      description: Identifier of the Record
      required: true
      schema:
        type: string
        example: 'UserRecordValue000000001'
    - name: blockId
      in: path
      description: Id of the Block
      required: true
      schema:

```

```

    type: string
    example: 'userDefjson01'
  - 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':
    $ref: '#/components/responses/BlockBody'
  '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 or Update a specific Block in a Record.
  description: Create or update a specific Block, related to a Record
  operationId: CreateOrModifyBlock
  tags:
  - Block 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: recordId
    in: path
    description: Identifier of the Record
    required: true
    schema:
      type: string
      example: 'UserRecordValue000000001'
  - name: blockId
    in: path
    description: Id of the Block
    required: true
    schema:
      type: string
      example: 'userDefjson01'
  - name: get-previous
    in: query
    description: Retrieve the Block before update
    required: false
    schema:
      type: boolean
      default: false

```

```

- 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: Record validator for conditional requests, as described in RFC 7232, 3.2
  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'
requestBody:
  description: information on the Block to create
  required: true
  content:
    '*/*':
      schema:
        $ref: '#/components/schemas/Block'
responses:
  '200':
    $ref: '#/components/responses/BlockBody'
  '201':
    description: >-
      Creation case. The Block has been successfully created. Location header indicates the
URI of the created Block.
    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':
    description: >-
      Successful case. The resource has been successfully updated.
    headers:
      Cache-Control:
        $ref: '#/components/headers/Cache-Control'
      ETag:
        $ref: '#/components/headers/ETag'
      Last-Modified:
        $ref: '#/components/headers/Last-Modified'
  '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': # Return previous Block value if get-previous=true
    $ref: '#/components/responses/BlockBody'
  '413':
    $ref: 'TS29571_CommonData.yaml#/components/responses/413'
  '500':
    $ref: 'TS29571_CommonData.yaml#/components/responses/500'
  '503':
    $ref: 'TS29571_CommonData.yaml#/components/responses/503'
  default:
    $ref: 'TS29571_CommonData.yaml#/components/responses/default'
delete:
  summary: Delete a specific Block. Then update the Record
  description: delete a specific Block, related to a Record
  operationId: DeleteBlock
  tags:
  - Block 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: recordId
    in: path
    description: Identifier of the Record
    required: true
    schema:
      type: string
      example: 'UserRecordValue000000001'
  - name: blockId
    in: path
    description: Id of the Block
    required: true
    schema:
      type: string
      example: 'userDefjson01'
  - name: get-previous
    in: query
    description: Retrieve the Block before delete
    required: false
    schema:
      type: boolean
      default: false
  - name: If-Match
    in: header
    description: Record validator for conditional requests, as described in RFC 7232, 3.2
    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':
    $ref: '#/components/responses/BlockBodyDelete'
  '204':
    description: >-
      Successful case. The Block has been successfully deleted.
    headers:
      ETag:
        $ref: '#/components/headers/ETag'
      Last-Modified:
        $ref: '#/components/headers/Last-Modified'
  '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': # Return previous Block value if get-previous=true
    $ref: '#/components/responses/BlockBody'
  '500':
    $ref: 'TS29571_CommonData.yaml#/components/responses/500'
  '503':
    $ref: 'TS29571_CommonData.yaml#/components/responses/503'
  default:
    $ref: 'TS29571_CommonData.yaml#/components/responses/default'

/{realmId}/{storageId}/subs-to-notify:
  summary: The notification subscription collection resource
  description: >-
    Access to the subscription resource
  get:
    summary: Notification subscription retrieval

```



```

description: retrieve all notification subscriptions of the storage
operationId: GetNotificationSubscriptions
tags:
- NotificationSubscriptions 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: limit-range
  in: query
  description: The maximum number of NotificationSubscriptions to fetch
  schema:
    $ref: 'TS29571_CommonData.yaml#/components/schemas/Uinteger'
- 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: Expected response to a valid request
  content:
    application/json:
      schema:
        type: array
        items:
          $ref: '#/components/schemas/NotificationSubscription'
'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'

/{realmId}/{storageId}/subs-to-notify/{subscriptionId}:
summary: The notification subscription resource
description: >-
  Access to the subscription resource
get:
summary: Notification subscription retrieval
description: retrieve a single notification subscription of the storage
operationId: GetNotificationSubscription
tags:
- NotificationSubscription 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: subscriptionId
    in: path
    description: Identifier of the NotificationSubscription
    required: true
    schema:
      type: string
      example: Subscription01
  - name: supported-features
    in: query
    description: Features required to be supported by the target NF
    schema:
      $ref: 'TS29571_CommonData.yaml#/components/schemas/SupportedFeatures'
  - 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
responses:
  '200':
    description: Expected response to a valid request
    headers:
      Cache-Control:
        $ref: '#/components/headers/Cache-Control'
      ETag:
        $ref: '#/components/headers/ETag'
      Last-Modified:
        $ref: '#/components/headers/Last-Modified'
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/NotificationSubscription'
  '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'

delete:
  summary: Delete a Notification Subscription of the storage
  description: delete a single subscriptions of the storage
  operationId: DeleteNotificationSubscription
  tags:
  - NotificationSubscription 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: subscriptionId
    in: path

```

```

description: Identifier of the NotificationSubscription
required: true
schema:
  type: string
  example: Subscription01
- name: client-id
  in: query
  description: Identifies the NF or NFSet
  required: true
  schema:
    $ref: '#/components/schemas/ClientId'
- name: get-previous
  in: query
  description: Retrieve the NotificationSubscription before delete
  required: false
  schema:
    type: boolean
    default: false
- name: If-Match
  in: header
  description: Record validator for conditional requests, as described in RFC 7232, 3.2
  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':
    description: Deleted NotificationSubscription if requested with get-previous
    content:
      application/json:
        schema:
          type: array
          items:
            $ref: '#/components/schemas/NotificationSubscription'
  '204':
    description: >-
      Successful case. The SubscriptionNotification has been successfully deleted.
  '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 previous NotificationSubscription value if get-previous=true
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/NotificationSubscription'
  '500':
    $ref: 'TS29571_CommonData.yaml#/components/responses/500'
  '503':
    $ref: 'TS29571_CommonData.yaml#/components/responses/503'
  default:
    $ref: 'TS29571_CommonData.yaml#/components/responses/default'

patch: # patch NotificationSubscription data
summary: NotificationSubscription update
description: update a specific NotificationSubscription
operationId: UpdateNotificationSubscription
tags:
- NotificationSubscription 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: subscriptionId
    in: path
    description: Identifier of the NotificationSubscription
    required: true
    schema:
      type: string
      example: Subscription01
  - name: If-Match
    in: header
    description: Validator for conditional requests, as described in RFC 7232, 3.2
    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'
requestBody:
  description: data to patch
  content:
    application/json-patch+json:
      example: 'TBD'
      schema:
        type: array
        items:
          $ref: 'TS29571_CommonData.yaml#/components/schemas/PatchItem'
        minItems: 1
      required: true
  responses:
    '200':
      description: >-
        One or more modification instructions have been discarded, the execution report is
returned in response PatchResult.
      content:
        application/json:
          example:
          schema:
            $ref: 'TS29571_CommonData.yaml#/components/schemas/PatchResult'
      headers:
        Cache-Control:
          $ref: '#/components/headers/Cache-Control'
        ETag:
          $ref: '#/components/headers/ETag'
        Last-Modified:
          $ref: '#/components/headers/Last-Modified'
    '204':
      description: >-
        Successful case. The meta has been successfully updated and no return is expected.
      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'
    '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: NotificationSubscription Create/Update
  operationId: CreateAndUpdateNotificationSubscription
  tags:
    - NotificationSubscription 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: subscriptionId
      in: path
      description: Identifier of the NotificationSubscription
      required: true
      schema:
        type: string
        example: Subscription01
    - name: supported-features
      in: query
      description: Features required to be supported by the target NF
      schema:
        $ref: 'TS29571_CommonData.yaml#/components/schemas/SupportedFeatures'
    - 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: Record validator for conditional requests, as described in RFC 7232, 3.2
      schema:
        type: string
  requestBody:
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/NotificationSubscription'
        required: true
  responses:
    '200': # Update
      description: Expected response to a valid update request
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/NotificationSubscription'
    '201':
      description: Expected response to a valid create request
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/NotificationSubscription'
  headers:
    Location:
      description: 'Contains the URI of the newly created resource according to the
structure: {apiRoot}/nudsf-dr/<apiVersion>/{realmId}/{storageId}/subs-to-notify/{subscriptionId}'
      required: true
      schema:
        type: string
    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'
'409':
  description: Conflict
  content:
    application/json:
      schema:
        type: array
        items:
          $ref: 'TS29571_CommonData.yaml#/components/schemas/Uri'
'412':
  $ref: 'TS29571_CommonData.yaml#/components/responses/412'
'500':
  $ref: 'TS29571_CommonData.yaml#/components/responses/500'
'503':
  $ref: 'TS29571_CommonData.yaml#/components/responses/503'
default:
  description: Unexpected error
  content:
    application/problem+json:
      schema:
        $ref: 'TS29571_CommonData.yaml#/components/schemas/ProblemDetails'
callbacks:
  onDataChange:
    '{request.body#/callbackReference}':
      post:
        requestBody:
          $ref: '#/components/requestBodies/RecordNotificationBody'
        responses:
          '204':
            description: Callback executed successfully
          '400':
            $ref: 'TS29571_CommonData.yaml#/components/responses/400'
          '401':
            $ref: 'TS29571_CommonData.yaml#/components/responses/401'
          '403':
            $ref: 'TS29571_CommonData.yaml#/components/responses/403'
          '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

schemas:
  RecordSearchResult:
    description: Count and collection of Record references matching the providing filter.
    type: object
    properties:
      count:
        # The total number of elements found.
        $ref: 'TS29571_CommonData.yaml#/components/schemas/UInteger'
      references:
        # The Record references found. If count-indicator is true, no references are
sent back.
        type: array
        items:
          $ref: 'TS29571_CommonData.yaml#/components/schemas/Uri'
        minItems: 1
      supportedFeatures:
        $ref: 'TS29571_CommonData.yaml#/components/schemas/SupportedFeatures'
    required:

```

```

- count
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" ] }}

Record:
  description: Definition of a Record
  type: object
  properties:
    meta:
      # json representation of the Meta Data
      $ref: '#/components/schemas/RecordMeta'
    blocks:
      # List of multipart data
      type: array
      description: list of opaque Block's in this Record
      items:
        $ref: '#/components/schemas/Block'
      minItems: 1
  required:
    - meta
  example: >-
    { "meta": { "tags" : { "tag1" : ["value1"], "tag2" : ["value2"] } }, "blocks": [{"Content-ID":
"userDefBinaryBlob", "Content-Type": "text/plain", "content": "QmxvY2sgY29udGVudA=="}, {"Content-
Id": "userDefJsonBlob", "Content-Type": "application/json", "content": "{\"key\": \"ftsimpletype-
999550000000002\", \"value\": \"A3E71A78377179B5B91A;imsi-999550000000123\"}"}]}

Block:
  description: A Block can be of any type
  example: >-
    "QmxvY2sgY29udGVudA=="
NotificationSubscription:
  description: Definition of a notification subscription
  type: object
  properties:
    clientId:
      $ref: '#/components/schemas/ClientId'
    callbackReference:
      $ref: 'TS29571_CommonData.yaml#/components/schemas/Uri'
    expiry:
      $ref: 'TS29571_CommonData.yaml#/components/schemas/DateTime'
    subFilter:
      $ref: '#/components/schemas/SubscriptionFilter'
    supportedFeatures:
      $ref: 'TS29571_CommonData.yaml#/components/schemas/SupportedFeatures'
  required:
    - clientId
    - callbackReference

RecordNotification:
  description: Definition of a notification on a record
  type: object
  properties:
    descriptor:
      # json representation of the notification description
      $ref: '#/components/schemas/NotificationDescription'
    meta:
      # json representation of the Meta Data
      $ref: '#/components/schemas/RecordMeta'
    blocks:

```

```

# List of multipart data
type: array
description: list of opaque Block's in this Record
items:
  $ref: '#/components/schemas/Block'
required:
- descriptor
- meta
example: >-
  {"descriptor": { "recordRef" : "...", "operationType" : "DELETED"}, "meta": { "tags" :
{"tag1" : ["value1"], "tag2" :["value2"] } }, "blocks": [{"Content-ID": "userDefBinaryBlob",
"Content-Type": "text/plain", "content": "QmxvY2sgY29udGVudA=="}, {"Content-Id": "userDefJsonBlob",
"Content-Type": "application/json", "content": "{\"key\": \"ftsimplotype-99955000000002\", \"value\":
\"A3E71A78377179B5B91A;imsi-999550000000123\"}"]}

```

```

NotificationDescription:
description: Description of a record notification
type: object
properties:
  recordRef:
    $ref: 'TS29571_CommonData.yaml#/components/schemas/Uri'
  operationType:
    $ref: '#/components/schemas/RecordOperation'
  subscriptionId:
    # unique identifier of the NotificationSubscription
    type: string
required:
- recordRef
- operationType
example: >-
  { "record" : "...", "operationType" : "DELETED"}

```

```

SubscriptionFilter:
description: A subscription filter
type: object
properties:
  monitoredResourceUris:
    type: array
    description: list of resources applicable to the subscription
    items:
      $ref: 'TS29571_CommonData.yaml#/components/schemas/Uri'
    minItems: 1
  operations:
    type: array
    description: list of resources applicable to the subscription
    items:
      $ref: '#/components/schemas/RecordOperation'
    maxItems: 3

```

```

ClientId:
description: Defines the identity of the NF Consumer
type: object
properties:
  nfId:
    $ref: 'TS29571_CommonData.yaml#/components/schemas/NfInstanceId'
  nfSetId:
    $ref: 'TS29571_CommonData.yaml#/components/schemas/NfSetId'

```

```

RecordOperation:
description: Indicate operation made on a record
anyOf:
- type: string
  enum:
    - CREATED
    - UPDATED
    - DELETED
- type: string

```

```

ConditionOperator:
description: TBD
anyOf:
- type: string
  enum:
    - AND
    - OR
    - NOT
- type: string

```



```

ComparisonOperator:
  description: TBD
  anyOf:
    - type: string
      enum:
        # Equals
        - EQ
        # Not Equal
        - NEQ
        # Greater Than
        - GT
        # Greater Than or Equal
        - GTE
        # Less Than
        - LT
        # Less Than or Equal
        - LTE
    - type: string

SearchExpression:
  description: A logical expression element
  type: object
  oneOf:
    - $ref: '#/components/schemas/SearchCondition'
    - $ref: '#/components/schemas/SearchComparison'
  example:
    { "cond": "OR", "units": [ { "op": "EQ", "tag": "ueId", "value": "455345" }, { "op": "EQ",
"tag": "supi", "value": "imsi-999559807001001" } ] }

SearchCondition:
  description: A logical condition
  type: object
  properties:
    cond:
      $ref: '#/components/schemas/ConditionOperator'
    units:
      type: array
      items:
        $ref: '#/components/schemas/SearchExpression'
      minItems: 1
  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:
    - op
    - tag
    - value
  example:
    { "op": "EQ", "tag": "supi", "value": "imsi-999559807001001" }

headers:
  Cache-Control:
    description: Cache-Control containing max-age, as described in RFC 7234, 5.2
    schema:
      type: string
  ETag:
    description: Entity Tag, containing a strong validator, as described in RFC 7232, 2.3
    schema:
      type: string
  Last-Modified:
    description: Timestamp for last modification of the resource, as described in RFC 7232, 2.2
    schema:
      type: string

```

```

Location:
  description: Contains the URI of the newly created resource
  required: true
  schema:
    type: string
Retry-After:
  description: 'Indicates the time the NF Consumer has to wait before making a new request. It
can be a non-negative integer (decimal number) indicating the number of seconds the NF Consumer has
to wait before making a new request or an HTTP-date after which the AF can retry a new request.'
  schema:
    anyOf:
      - type: integer
      - type: string

requestBodies:
  RecordBody:
    description: The record multipart request body. The meta part shall be the first part and is
mandatory but can be empty and zero or more block parts may follow the meta part.
    required: true
    content:
      multipart/mixed:
        schema:
          $ref: '#/components/schemas/Record'
        encoding:
          meta: # The meta part shall be the first part and is mandatory but can be empty
            contentType: application/json
            headers:
              Content-ID:
                schema:
                  type: string
                  required: true
              blocks: # 0 or more block parts may follow the meta part
                contentType: '*/*' # Block part can be of any type
                headers:
                  Content-ID: # Block identifier is defined by the Content-ID header.
                    schema:
                      type: string
                      required: true
                  Content-Transfer-Encoding:
                    schema:
                      type: string
                      required: true

  RecordNotificationBody:
    description: The record notification multipart request body. The descriptor part shall be the
first one, followed by record meta part and by zero or more block parts.
    required: true
    content:
      multipart/mixed:
        schema:
          $ref: '#/components/schemas/RecordNotification'
        encoding:
          descriptor: # The descriptor part shall be the first part and is mandatory
            contentType: application/json
            headers:
              Content-ID:
                schema:
                  type: string
                  required: true
          meta: # The meta part shall be the second part and is mandatory but can be empty
            contentType: application/json
            headers:
              Content-ID:
                schema:
                  type: string
                  required: true
          blocks: # 0 or more block parts may follow the meta part
            contentType: '*/*' # Block part can be of any type
            headers:
              Content-ID: # Block identifier is defined by the Content-ID header.
                schema:
                  type: string
                  required: true
              Content-Transfer-Encoding:
                schema:
                  type: string
                  required: true

```

```

responses:
  '304': # Etag response if the value might differ from that sent
    description: Not Modified
    content:
      application/problem+json:
        schema:
          $ref: 'TS29571_CommonData.yaml#/components/schemas/ProblemDetails'
    headers:
      Cache-Control:
        $ref: '#/components/headers/Cache-Control'
      ETag:
        $ref: '#/components/headers/ETag'
      Retry-After:
        $ref: '#/components/headers/Retry-After'

'RecordBody': # Record value with associated headers
  description: >-
    - 200 Update. The resource has been successfully updated and previous value must be sent in
the response message if requested.
    - 200 Get. The resource exists, its value must be sent in the response message
    - 412 Precondition Failed, the previous value must be sent in response message if
requested.
  content:
    multipart/mixed:
      schema:
        $ref: '#/components/schemas/Record'
      encoding:
        meta: # The meta part shall be the first part and is mandatory but can be empty.
          contentType: application/json
          headers:
            Content-Id: # The meta part is identified by the 'meta' Content-Id header.
              schema:
                type: string
                required: true
            blocks: # Zero or more block parts may follow the meta part
              contentType: '*/*' # Block parts can be of any type.
              headers:
                Content-Id: # Block identifier is defined by the Content-Id header.
                  schema:
                    type: string
                    required: true
                Content-Transfer-Encoding:
                  schema:
                    type: string
                    required: true
          headers:
            Cache-Control:
              $ref: '#/components/headers/Cache-Control'
            ETag:
              $ref: '#/components/headers/ETag'
            Last-Modified:
              $ref: '#/components/headers/Last-Modified'

'RecordBodyDelete': # Record value with associated headers
  description: >-
    - 200 Delete. The resource has been successfully delete and previous value must be sent in
the response message if requested.
  content:
    multipart/mixed:
      schema:
        $ref: '#/components/schemas/Record'
      encoding:
        meta: # The meta part shall be the first par and is mandatory but can be empty.
          contentType: application/json
          headers:
            Content-ID: # The meta part is identified by the 'meta' Content-Id header.
              schema:
                type: string
                enum:
                  - meta
                required: true
            blocks: # Zero or more block parts may follow the meta part.
              contentType: '*/*' # Block parts can be of any type.
              headers:
                Content-ID: # Block identifier is defined by the Content-Id header.
                  schema:
                    type: string
                    required: true

```

```
        Content-Transfer-Encoding:
          schema:
            type: string
            required: true
headers:
  ETag:
    $ref: '#/components/headers/ETag'
  Last-Modified:
    $ref: '#/components/headers/Last-Modified'

'BlockBody': # Block value with associated headers
description: >-
  - 200 Update: The resource has been successfully updated and previous value must be sent in
the response message if requested.
  - 200 Get: The resource exists, its value must be sent in the response message
  - 412 Precondition Failed: the previous value must be sent in response message if
requested.
content:
  '*/*':
    schema:
      $ref: '#/components/schemas/Block'
headers:
  Cache-Control:
    $ref: '#/components/headers/Cache-Control'
  ETag:
    $ref: '#/components/headers/ETag'
  Last-Modified:
    $ref: '#/components/headers/Last-Modified'

'BlockBodyDelete': # Block value with associated headers
description: >-
  - 200 Delete: The resource has been successfully delete and previous value must be sent in
the response message if requested.
content:
  '*/*':
    schema:
      $ref: '#/components/schemas/Block'
headers:
  ETag:
    $ref: '#/components/headers/ETag'
  Last-Modified:
    $ref: '#/components/headers/Last-Modified'
```

Annex B (informative): Search Examples

The conditional expression is defined by the following Extended Backus-Naur Form (EBNF) [18].



Figure B-1: Search Expression

$\text{Search_Expression} ::= \text{Search_Comparison} \mid \text{Search_Condition}$

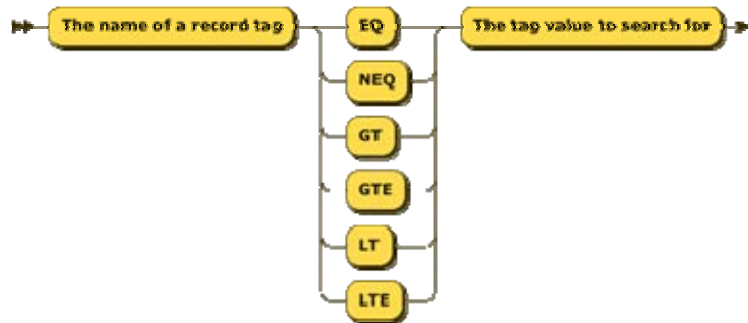


Figure B-2: Search Comparison

$\text{Search_Comparison} ::= \text{Tag} (\text{'EQ'} \mid \text{'NEQ'} \mid \text{'GT'} \mid \text{'GTE'} \mid \text{'LT'} \mid \text{'LTE'}) \text{Value}$

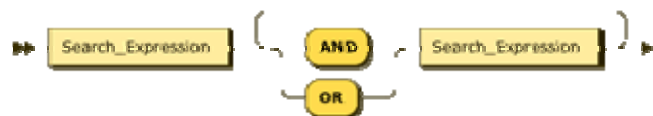


Figure B-3: Search Condition

$\text{Search_Condition} ::= \text{Search_Expression} ((\text{'AND'} \mid \text{'OR'}) \text{Search_Expression})_+$



Figure B-4: Search ConditionNot

$\text{Search_ConditionNot} ::= \text{'NOT'} \text{Search_Expression}$

Example:

Find all records where the tag "ueId" is equal to "455345" OR the tag "supi" is equal to "imsi-999559807001001":

```
{
  "cond": "OR",
  "items": [
    {
      "op": "EQ",
      "tag": "ueId",
      "value": "455345"
    },
  ],
}
```

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



```
sAU6YCGJRgMpt7F6gTNdRBpLo1oh94UvQJdrJiJVlJzCwNphiYSQh+KI0Npw9SQGcF0TYeVSuNpvi2B7IClMBSfabwVu9WtbuUq
NifngaEtFqP54Saw9DavnAtuz61ELegMf8PgEjCxGdvdtxjvDZVmdhOsagNrLjdg5gJzjuleDR/DdLC0sdr1u5glUPTHfnjdPT6p
qFxiukfyAl3sFO+BVTZgNeqHAHdjIW+0c35jsiQxETloPxG5c/S6+4+h4PV2sGGCVfmFbrJCVrzero0GsBuyIstx3g07L14ER1GfT
/rgbdLAOzpLHax67wXLvb7s+48eQsjWsiXxhKl/sw7AjvsiLRR1j416j4qvgd7vueisims+K1Hqx7f99q2WGlN+xPviOeVA1D2
jXqslLwTlVGQ4+/C82oAAAAASUVORK5CYII=
--partboundary--'
```

C.4 Example HTTP multipart RecordNotification

```
Content-Type: multipart/mixed; boundary=partboundary

--partboundary
Content-ID: 860013c6-240e-498a-8110-5a708a96aa54
Content-Type: application/json; charset=UTF-8

{"recordRef" : "...", "operationType" : "DELETED" }

--partboundary

Content-ID: 76elf8c1-e8ba-47af-8980-e9292bcffd6b
Content-Type: application/json; charset=UTF-8

{"tags": { "ueId" : ["455345"], "supi" : ["imsi-999559807001001"] } }

--partboundary

Content-ID: 5cda2686-efbb-47e0-a749-a6f92aaa58fb
Content-Type: application/json; charset=UTF-8
Content-Transfer-Encoding: binary

{"firstName": "John", "lastName": "Doe"}

--partboundary

Content-ID: 25d16458-019d-46a0-af25-92cc1adf2277
Content-Type: image/png
Content-Transfer-Encoding: binary

<binary representation of png>

--partboundary--'
```


Annex D (informative): Change history

Change history							
Date	Meeting	TDoc	CR	Rev	Cat	Subject/Comment	New version
2019-11	CT4#94	C4-195489				Initial Draft.	0.1.0
2020-03	CT4#96	C4-200359 C4-200360 C4-200584 C4-200585 C4-200586 C4-200654 C4-200905 C4-200920 C4-201226 C4-201230 C4-201235 C4-201236 C4-201237				Implementation of pCRs agreed at CT4#96.	0.2.0
2020-03	CT#87e	CP-200066				TS presented for information and approval	1.0.0
2020-03	CT#87e					Approved at CT#87e	16.0.0
2020-06	CT#88e	CP-201175	0001	4	B	Subscribe To Notify	16.1.0
2020-06	CT#88e	CP-201041	0002		C	Removal of Editor's Note	16.1.0
2020-06	CT#88e	CP-201041	0003		C	Add supportedFeatures to RecordSearchResult	16.1.0
2020-06	CT#88e	CP-201041	0004		F	SearchExpression	16.1.0
2020-06	CT#88e	CP-201041	0006	1	F	Miscellaneous Corrections	16.1.0
2020-06	CT#88e	CP-201041	0007	1	B	Storage of YAML files in ETSI Forge	16.1.0
2020-06	CT#88e	CP-201041	0008		F	204 missing in OpenAPI	16.1.0
2020-06	CT#88e	CP-201073	0009		F	Rel-16 API version and External doc update	16.1.0
2020-09	CT#89e	CP-202116	0010		F	Optionality of ProblemDetails in TS29.598 cleanup	16.2.0
2020-12	CT#90e	CP-203052	0011	1	F	Misc corrections	16.3.0
2020-12	CT#90e	CP-203035	0012		F	Removal of the reference to ETSI Forge	16.3.0
2020-12	CT#90e	CP-203052	0013	1	F	Corrections on Subscription and Notification	16.3.0
2020-12	CT#90e	CP-203052	0014	1	F	Corrections on yaml of Nudsf_DataRepository OpenAPI	16.3.0
2020-12	CT#90e	CP-203052	0015	1	F	Define Unsubscription to notifications service operation	16.3.0
2020-12	CT#90e	CP-203052	0016	1	F	Incorrect data type	16.3.0
2020-12	CT#90e	CP-203052	0018		F	Resource URI problems clean up	16.3.0
2020-12	CT#90e	CP-203036	0021		F	29.598 Rel-16 API version and External doc update	16.3.0
2021-03	CT#91e	CP-210039	0023	1	F	Removal of paging parameters	16.4.0
2021-03	CT#91e	CP-210039	0025		F	BlockId clarification	16.4.0
2021-03	CT#91e	CP-210054	0030		F	29.598 Rel-16 API version and External doc update	16.4.0

History

Document history		
V16.1.0	July 2020	Publication
V16.2.0	November 2020	Publication
V16.3.0	January 2021	Publication
V16.4.0	April 2021	Publication