ETSI TS 129 501 V18.5.0 (2024-07)

TECHNICAL SPECIFICATION

5G;
5G System;
Principles and Guidelines for Services Definition;
Stage 3
(3GPP TS 29.501 version 18.5.0 Release 18)

Reference

RTS/TSGC-0429501vi50

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 the ETSI Search & Browse Standards application.

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 on ETSI deliver.

Users should be aware that the present document may be revised or have its status changed, this information is available in the Milestones listing.

If you find errors in the present document, please send your comments to the relevant service listed under Committee Support Staff.

If you find a security vulnerability in the present document, please report it through our Coordinated Vulnerability Disclosure (CVD) program.

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

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 https://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 6

1 Scope 8

2 References 8

3 Definitions and abbreviations 9
3.1 Definitions 9
3.2 Abbreviations 10
3.3 Special characters, operators and delimiters 10
3.3.1 General 10
3.3.2 ABNF operators 10
3.3.3 URI reserved and special characters 10
3.3.4 SBI specific usage of delimiters 11

4 Design Principles for 5GC SBI APIs 11
4.1 General Principles 11
4.2 API Design Style and REST Implementation Levels 12
4.2.1 General 12
4.2.2 API Design Principles for Query Operation 12
4.2.3 API Design Principles for Delete Operation 12
4.3 Version Control 13
4.3.0 General 13
4.3.1 Structure of API version numbers 13
4.3.1.1 API version number format 13
4.3.1.2 Rules for incrementing field values 13
4.3.1.3 Visibility of the API version number fields 17
4.3.1.4 Relation to the Technical Specification version number 17
4.3.1.5 Discovery of the supported versions 17
4.3.1.6 Withdrawing API versions 18
4.4 URI Structure 18
4.4.1 Resource URI structure 18
4.4.2 Custom operations URI structure 19
4.4.3 Callback URI structure 19
4.5.1 Resource Representation 19
4.5.2 Content Format Negotiation 20
4.6 Use of HTTP Methods 20
4.6.1 Use of Request/Response Communication 20
4.6.1.1 CRUD 20
4.6.1.1.1 Creating a Resource 20
4.6.1.1.2 Reading a Resource 22
4.6.1.1.3 Updating a Resource 23
4.6.1.1.5 Query Parameters 25
4.6.1.2 Custom Operations 25
4.6.1.3 Use of Asynchronous Operations 27
4.6.1.4 Special provisions to support the seamless change of AMF as NF service producer 27
4.6.2 Use of Subscribe/Notify Communication 27
4.6.2.1 General 27
4.6.2.2 Management of Subscriptions 28
4.6.2.2.1 General 28
4.6.2.2.2 Creation of a Subscription 28
4.6.2.2.3 Modify a subscription 29
4.6.2.2.4 Delete a subscription 30
4.6.2.3 Notifications 31
4.6.2.4 Special provisions to support the seamless change of AMF as NF service consumer 32

4.7 HATEOAS 32
4.7.1 General....32
4.7.2 3GPP hypermedia format....32
4.7.2.1 Basic 3GPP hypermedia format 32
4.7.2.2 Extended 3GPP hypermedia format 33
4.7.3 Advertising legitimate application state transitions 34
4.7.4 Inferring link relation semantic....34
4.7.5 Common Relation Types 34
4.7.5.1 Introduction 34
4.7.5.2 Registered relation types 34
4.7.5.3 Extension relation types 35
4.7.6 Negotiating the support of optional HATEOAS features 35
4.8 Error Responses....35
4.8.1 Error Response Status Code....35
4.8.2 Error Response Body 35
4.8.3 Extending ProblemDetails for API Backward Compatibility 36
4.9 Transferring multiple resources to a NF Service Consumer....37
4.9.1 General....37
4.9.2 Direct Delivery 37
4.9.3 Direct Delivery with Iterations 37
4.9.4 Indirect Delivery....39
4.9.5 Indirect Delivery with HTTP/2 Server Push....39
4.9.6 Criteria for choosing the transfer method 41
4.10 Scopes definition for OAuth2.0 access token....41

5 Documenting 5GC SBI APIs 42
5.1 Naming Conventions....42
5.1.1 Case Conventions 42
5.1.2 API Naming Conventions....43
5.1.3 Conventions for URI Parts....43
5.1.3.1 Introduction....43
5.1.3.2 URI Path Segment Naming Conventions....44
5.1.3.3 URI Query Naming Conventions....44
5.1.4 Conventions for Names in Data Structures....45
5.2 API Definition....45
5.2.1 Resource Structure....45
5.2.2 Resources and HTTP Methods 47
5.2.3 Representing RPC as Custom Operations on Resources 49
5.2.3A Notifications 50
5.2.4 Data Models....52
5.2.4.1 General 52
5.2.4.2 Structured data types....53
5.2.4.3 Simple data types and enumerations 54
5.2.4.4 Binary Data 55
5.2.4.5 Data types describing alternative data types or combinations of data types 55
5.2.5 Void 56
5.2.6 3GPP Custom HTTP Headers....56
5.3 OpenAPI specification files....56
5.3.1 General....56
5.3.2 Formatting of OpenAPI specification files 57
5.3.3 Info....57
5.3.4 externalDocs 57
5.3.5 Servers 58
5.3.6 References to other 3GPP-defined OpenAPI specification files....58
5.3.7 Server-initiated communication....59
5.3.8 Describing the body of HTTP PATCH requests....60
5.3.8.1 General 60
5.3.8.2 JSON Merge Patch....60
5.3.8.3 JSON PATCH....61
5.3.8.4 Multipart Messages....61
5.3.9 Structured data types....61
5.3.10 Data types describing alternative data types or combinations of data types 64

5.3.11 Error Responses 65
5.3.12 Enumerations 66
5.3.13 Formatting of structured data types in query parameters 67
5.3.14 Attribute Presence Conditions 68
5.3.15 Usage of the "tags" field 70
5.3.16 Security 70
5.3.17 Reuse of Structured Data Types 72
5.3.18 Operation identifiers 73
5.3.19 Formatting of description fields 73

6 Requirements for secure API design 74
6.1 Introduction 74
6.2 General 74
6.3 SBA-specific requirements 75

Annex A (informative): TS Skeleton Template 76

Annex B (informative): Backward Incompatible Changes 77

Annex C (Informative): Resource modelling 79

C.0 General 79

C.1 Document 79

C.2 Collection 79

C.3 Store 79

C.4 Custom operation 80

Annex D (informative): Example of an OpenAPI specification file for Patch 81

Annex E (Informative): Considerations for handling of JSON arrays 85

Annex F (informative): Change history 87

History 91

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 defines design principles and documentation guidelines for 5GC SBI APIs. These principles and guidelines should be followed when drafting the 5G System SBI Stage 3 specifications.

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 29.500: "5G System; Technical Realization of Service Based Architecture; Stage 3".

[3] IETF RFC 8259: "The JavaScript Object Notation (JSON) Data Interchange Format".

[4] OpenAPI: "OpenAPI Specification Version 3.0.0", https://spec.openapis.org/oas/v3.0.0.

[5] 3GPP TS 29.571: "5G System; Common Data Types for Service Based Interfaces Stage 3".

[6] IETF RFC 7231: "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content".

[7] IETF RFC 7396: "JSON Merge Patch".

[8] IETF RFC 6902: "JavaScript Object Notation (JSON) Patch".

[9] IETF RFC 3986: "Uniform Resource Identifier (URI): Generic Syntax".

[10] IETF RFC 5789: "PATCH Method for HTTP".

[11] IETF RFC 8288: "Web Linking".

[12] IANA: "HTTP Status Code Registry at IANA", http://www.iana.org/assignments/http-status-codes.

[13] IETF RFC 7540: "Hypertext Transfer Protocol Version 2 (HTTP/2)".

[14] Fielding, Roy Thomas. Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine, 2000.

[15] Erik Wilde, Cesare Pautasso, REST: From Research to Practice, Springer.

[16] YAML 1.2: "YAML Ain't Markup Language", http://yaml.org.

[17] Semantic Versioning Specification: https://semver.org.

[18] 3GPP TS 29.510: "5G System; Network Function Repository Services; Stage 3".

[19] IETF RFC 9457: "Problem Details for HTTP APIs".

[20] 3GPP TS 29.502: "5G System; Session Management Services; Stage 3".

[21] 3GPP TS 29.509: "5G System; Authentication Server Services; Stage 3".

[22] 3GPP TS 33.501: "Security architecture and procedures for 5G system".

[23] IETF RFC 6749: "The OAuth 2.0 Authorization Framework".

[24] 3GPP TS 29.573: "5G System; Public Land Mobile Network (PLMN) Interconnection;Stage 3".

[25] 3GPP TR 21.900: "Technical Specification Group working methods".

[26] IETF RFC 5234: "Augmented BNF for Syntax Specifications: ABNF".

[27] 3GPP TS 23.003: "Numbering, addressing and identification".

[28] 3GPP TS 29.503: "5G System; Unified Data Management Services; Stage 3".

[29] IETF RFC 2046: "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types".

3 Definitions and abbreviations

3.1 Definitions

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

Absolute URI: Absolute URI follows generic URI syntax and consists of a hierarchical sequence of the following components: the "scheme", "authority", "path" and "query", i.e. excluding the "fragment" component. See clause 4.3 in IETF RFC 3986 [9].

apiRoot: apiRoot follows an absolute URI syntax, but excludes the following absolute URI identifiers: the "query" and "fragment" components. The API root contains the "scheme" and the "authority" components and may also contain an API prefix subcomponent. See clause 4.4.1. In 3GPP APIs, the default apiRoot value is "https://example.com" (see e.g. Annex A.2 in 3GPP TS 29.502 [20]), where "example.com" is replaced by the operator specific FQDN (for FQDN examples see clause 28.3.2 in 3GPP TS 23.003 [27]).

API Prefix: API prefix is an optional, deployment-specific string, which is a subcomponent of the "apiRoot". API prefix format follows "path-absolute" syntax and that starts with a "/" reserved character (see clause 4.4.1and also clause 3.3 in IETF RFC 3986 [9]). In 3GPP APIs, the default apiRoot does not contain API Prefix (see e.g. Annex A.2 in 3GPP TS 29.502 [20]).

API URI: API URI has the following format: "{apiRoot}/<apiName>/<apiVersion>". For more details see clause 4.4.1. As an example, for the Nudm_SDM API defined in 3GPP TS 29.503 [28], the API URI is: "{apiRoot}/nudm-sdm/v2".

Base URI: Base URI is used as a reference against which the relative URI reference is applied. See clause 4.6.1.1.1.2 and also clause 5.1 in IETF RFC 3986 [9]. As an example, for the resource URI of the SM contexts collection that is defined in Nsmf_PDUSession API (see 3GPP TS 29.502 [20]), the base URI is "{apiRoot}/nsmf-pdusession/<apiVersion>".

Callback URI: Callback URI follows an absolute URI syntax, but excludes the following absolute URI identifiers: "userinfo" subcomponent of the "authority" component and also the "query" component ("fragment" component is already excluded from the absolute URI). Therefore, callback URI contains the "scheme", "authority" (excluding "userinfo" subcomponent) and the "path" components. See clause 4.4.3. As an example, for the Nsmf_PDUSession API defined in 3GPP TS 29.502 [20], "smContextStatusUri" is one of the callback references that can be provided by the NF Service Consumer. A complete callback URI for the "smContextStatusUri" has the following structure https://<comsumer-host>/<consumer-path-segment>/smContextStatus, where "consumer-host" is either an FQDN or an IP address (see clause 4.4.3).

Relative path after API URI: Refers to the sequence of path segments of a resource URI that are below the API URI (see the definition of the "Paths Object" in OpenAPI specification [4]). When appended to the API URI of the concerned API, it constitutes the resource URI. As an example, for the Nudm_SDM API defined in 3GPP TS 29.503 [28] for which the API URI is "{apiRoot}/nudm-sdm/v2", the "relative path after API URI" for the "AccessAndMobilitySubscriptionData" resource is: "/{supi}/am-data".

Relative URI: Relative URI is deprecated and shall not be use in SBI spcificationsSee clause 1.2.3 in IETF RFC 3986 [9].

Resource URI: Resource URI identifies an abstract or a physical resource. See the Abstract of the IETF RFC 3986 [9]. In this specification this generally means "an URI of a resource". Resource URI structure is defined in clause 4.4.1. As an example, for the resource URI of the SM contexts collection that is defined in Nsmf_PDUSession API (see 3GPP TS 29.502 [20]), the resource URI is "{apiRoot}/nsmf-pdusession/<apiVersion>/sm-contexts".

3.2 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
CNF	Conjunctive Normal Form
CRUD	Create, Read, Update, Delete
DNF	Disjunctive Normal Form
HAL	Hypertext Application Language
HATEOAS	Hypermedia as the Engine of Application State
REST	REpresentational State Transfer
RPC	Remote-Procedure-Call
SBI	Service Based Interface
YAML	YAML Ain't Markup Language

3.3 Special characters, operators and delimiters

3.3.1 General

A number of characters have special meaning and are used as delimiters in this document and also in other stage 3 SBI specifications. Below clauses specify the usage of a selected set of the special characters. Full set of these special characters are specified in the respective IETF specifications.

3.3.2 ABNF operators

Special characters that are used as delimiters in ABNF syntax have somewhat different purpose from the same characters when used by URI syntax. See clause 3.3.2 in 3GPP TS 29.500 [2].

3.3.3 URI reserved and special characters

/ Reserved character. The forward slash character is a delimiter, which precedes an URI path component and also separates a sequence of path segments. See clauses 2.2 and 3.3 in IETF RFC 3986 [9].

# Reserved character. The number sign (hash) character is a delimiter, which terminates either an URI path component, or an authority component, or a query component and starts a fragment component. See clauses 2.2, 3.2, 3.3 and 3.5 in IETF RFC 3986 [9].

? Reserved character. The question mark character is a delimiter, which terminates either an URI path component, or an authority component and starts a query component/parameter. See clauses 2.2, 3.2, 3.3 and 3.4 in IETF RFC 3986 [9].

= Reserved character. The equal sign character is a delimiter, which separates a parameter name and its value, e.g. in an URI query. See clauses 1.2.3 and 3.3 in IETF RFC 3986 [9].

[ ] Reserved character. The square bracket characters enclose an IPv6 literal address. This is the only place where square bracket characters are allowed in the URI syntax. See clause 3.2.2 in IETF RFC 3986 [9].

< > Special characters. The angle bracket characters are delimiters and enclose a complete URI or an URI component or a subcomponent. See Appendix C in IETF RFC 3986 [9].

; Reserved character. The semicolon character is a delimiter, which is often used to separates a parameter name and its value in a path segment. See clause 3.3 in IETF RFC 3986 [9].

: Reserved character. The colon character is a delimiter, which either terminates an URI scheme component, or userinfo subcomponent, or separates host subcomponent from the port subcomponent, an authority component, or a query component and starts a fragment component. See clauses 2.2, and 3.2 in IETF RFC 3986 [9].

NOTE: The same characters, like "/", "#", etc. lead to different processing in ABNF and URI grammars. For instance, in URI syntax, ";" character separates parameter and its value, while in ABNF ";" starts a comment.

3.3.4 SBI specific usage of delimiters

The braces (curly brackets) are used as delimiters across the SBI specifications, but have somewhat different purpose as specified below.

{ }	Delimiters. The braces (curly brackets) characters enclose a name of a variable in an URI path segment (see clause 5.1.3.2). Example: …/subscriber-data/{supi}.
{ }	Delimiters. The braces (curly brackets) characters enclose a parameter value in a query (see clause 5.1.3.3). Example: ?nf-id={chooseAValue}

NOTE: In SBI specifications, "< >" is a generic placeholder, while "{ }" enclose specifically a variable.

4 Design Principles for 5GC SBI APIs

4.1 General Principles

Each 5GC SBI API specification should include the following information for each specified service:

- Purpose of the API;
- URIs of resources;
- Supported HTTP methods for a given resource;
- Supported representations (e.g. JSON, see IETF RFC 8259 [3]);
- Request body schema(s) (where applicable);
- Response body schema(s) (where applicable);
- Supported response status codes;
- Relation types supported if HATEOAS is implemented by the API;
- A reference in the resource description clause to one of the archetypes defined in Annex C if the resource design matches one of them; and
- A list defining identifiers of optional features (see clause 6.6 of 3GPP TS 29.500 [2] for related procedures).

For each specified service a clause to a normative Annex should be provided containing the OpenAPI definitions according to OpenAPI Specification [4] for the service. The specifications should state that content of this normative annex takes precedence when being discrepant to other parts of the specification with respect to the encoding of information elements and methods.

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.

The TS Skeleton Template as provided in Annex A should be used as a starting point when drafting 5GC SBI API specifications.

Common procedures, HTTP extensions and error handling applicable to several 5GC SBI API specifications should be defined in 3GPP TS 29.500 [2] and should be referenced from individual 5GC SBI API specifications.

Common data types applicable to several 5GC SBI API specifications should be defined in 3GPP TS 29.571 [5] and should be referenced from individual 5GC SBI API specifications.

4.2 API Design Style and REST Implementation Levels

4.2.1 General

5GC SBI API specifications should apply a protocol design framework as follows:

a) REST-style service operations should implement the Level 2 of the Richardson maturity model, with standard HTTP methods, whenever it is a good match for the style of interaction to model, e.g. service operations that can naturally map to one of the standard methods (CRUD operations), this should be the preferred modelling attempt;

b) service operations may use custom API operations (RPC-style interaction), when it is seen a better fit for the style of interaction to model, e.g. non-CRUD service operations;

c) it is possible to mix REST-style operations and RPC-style operations in the same API.

NOTE: Level 3 (HATEOAS) of the Richardson maturity model in the 5G Service-Based Architecture can be implemented by an API but is optional. Hypermedia usage guidelines are provided in clause 4.7 of the present specification.

4.2.2 API Design Principles for Query Operation

When designing a query operation API, i.e. the NF service consumer invokes the API aiming to retrieve certain information from the NF service producer, the following principles should be applied:

a) if the query operation does not require any input parameter for the NF service producer, then the REST-style service operation with standard HTTP GET method should be used (see clause 4.6.1.1.2);

b) if

- the query operation requires input parameter(s) for the NF service producer; and
- all the required input parameter(s) are used to identify a particular resource and/or control the content of the result of the query operation;

then the REST-style service operation with standard HTTP GET method should be used (see clause 4.6.1.1.2);

c) standard HTTP GET method shall not be used for non-safe operations and non-idempotent operations.

4.2.3 API Design Principles for Delete Operation

When designing a delete operation API, i.e. the NF service consumer invokes the API aiming to delete certain resource on the NF service producer, the following principles should be applied:

a) if the delete operation does not require any input parameter for the NF service producer, then the REST-style service operation with standard HTTP DELETE method should be used (see clause 4.6.1.1.4);

b) if

- the delete operation requires input parameter(s) for the NF service producer; and
- all the required input parameter(s) are used to identify a particular resource and/or control the content of the result of the delete operation;

then the REST-style service operation with standard HTTP DELETE method should be used (see clause 4.6.1.1.4);

c) standard HTTP DELETE method shall not be used for non-idempotent operations.

4.3 Version Control

4.3.0 General

The version control mechanism in the present clause allows the management of changes to an API and provides a version number that is incremented whenever changes to the API are applied.

NOTE: The version number does not reflect the usage of optional features. A mechanism to negotiate the usage of optional features is defined in clause 6.6 of 3GPP TS 29.500 [2].

4.3.1 Structure of API version numbers

4.3.1.1 API version number format

API version numbers shall consist of at least 3 fields, following a MAJOR.MINOR.PATCH pattern according to the Semantic Versioning Specification [17].

The 1st Field (MAJOR), the 2nd Field (MINOR), and the 3rd Field (PATCH) shall contain unsigned integer numbers, and they shall not contain leading zeroes.

An additional field (called "pre-release version" in Semantic Versioning Specification [17]) is added to denote an OpenAPI version under development, i.e. prior to the freeze of the corresponding OpenAPI description for a given 3GPP Release. This additional field is appended after the 3 first version fields using the hyphen "-" character and shall have the format "alpha.*n*", where "*n*" is an unsigned integer number without leading zeroes.

NOTE: An OpenAPI is referred to as "non-frozen", if for a given 3GPP Release the OpenAPI freeze milestone was not reached. Otherwise, an OpenAPI is referred to as "frozen".

After the freeze of a 3GPP Release, additional fields (called "build metadata" in Semantic Versioning Specification [17]), containing operator-specific version information, may be appended after the 3 first version fields using the plus sign "+" character and they shall consist of a list of dot-separated identifiers, where each identifier may contain only alphanumeric characters and hyphens ([0-9A-Za-z-]).

NOTE: Operator-specific version information are ignored when determining version precedence. Thus, two versions that differ only in the operator-specific version information, have the same precedence.

EXAMPLES:

"1.0.0-alpha.1"

"3.0.1+orange.2020-09"

4.3.1.2 Rules for incrementing field values

The first version of a new API under development shall obtain the version number "1.0.0-alpha.1". At the first publication of the 3GPP Technical Specification defining the API after the OpenAPI freeze of the first 3GPP Release that contains the API, the version number of the API shall be set to "1.0.0".

When a new version of the 3GPP TS containing OpenAPI file(s) is published, the fields of the corresponding API version number(s) shall be incremented according to the following rules:

- The 1st Field (MAJOR) shall be incremented only if the applied change is backward incompatible relative to the earlier, i.e. frozen version of the given OpenAPI (see the details below). For a non-frozen OpenAPI, the first backwards incompatible change relative to the latest frozen version triggers incrementing the 1st Field (MAJOR), while subsequent backwards incompatible changes do not increment the value, until the OpenAPI stays non-frozen. When the 1st Field (MAJOR) is incremented, this impacts the lower level fields as follows:
  - The 2nd Field (MINOR) shall be reset to "0";
  - The 3rd Field (PATCH) shall be reset to "0";