|  |  |
| --- | --- |
| 3GPP TS 29.535 V0.0.0 (2020-11) | |
| Technical Specification | |
| 3rd Generation Partnership Project;  Technical Specification Group Core Network and Terminals;  5G System; AKMA Anchor Services  Stage 3  (Release 17) | |
|  | |
| *5G-logo_175px* | 3GPP-logo_web |
|  | |
| The present document has been developed within the 3rd Generation Partnership Project (3GPP TM) and may be further elaborated for the purposes of 3GPP. The present document has not been subject to any approval process by the 3GPPOrganizational Partners and shall not be implemented. This Specification is provided for future development work within 3GPPonly. The Organizational Partners accept no liability for any use of this Specification. Specifications and Reports for implementation of the 3GPP TM system should be obtained via the 3GPP Organizational Partners' Publications Offices. | |

|  |
| --- |
|  |
| ***3GPP***  Postal address  3GPP support office address  650 Route des Lucioles - Sophia Antipolis  Valbonne - FRANCE  Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16  Internet  http://www.3gpp.org |
| ***Copyright Notification***  No part may be reproduced except as authorized by written permission. The copyright and the foregoing restriction extend to reproduction in all media.  © 2020, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TSDSI, TTA, TTC).  All rights reserved.  UMTS™ is a Trade Mark of ETSI registered for the benefit of its members  3GPP™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners LTE™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners  GSM® and the GSM logo are registered and owned by the GSM Association |

Contents

Foreword 5

Introduction 6

1 Scope 7

2 References 7

3 Definitions, symbols and abbreviations 8

3.1 Definitions 8

3.2 Symbols 8

3.3 Abbreviations 8

4 Overview 8

5 Services offered by the <NF > 8

5.1 Introduction 8

5.2 <Service 1> Service 9

5.2.1 Service Description 9

5.2.2 Service Operations 9

5.2.2.1 Introduction 9

5.2.2.2 <Service operation 1> 9

5.2.2.2.1 General 9

5.2.2.2.2 <Procedure 1 using service operation 1 of service 1> 9

5.2.2.2.3 <Procedure 2 using service operation 1 of service 1> 9

5.2.2.3 <Service operation 2> 9

5.3 <Service 2 > Service 9

6 API Definitions 10

6.1 < Service 1> Service API 10

6.1.1 Introduction 10

6.1.2 Usage of HTTP 10

6.1.2.1 General 10

6.1.2.2 HTTP standard headers 10

6.1.2.2.1 General 10

6.1.2.2.2 Content type 10

6.1.2.3 HTTP custom headers 11

6.1.3 Resources 11

6.1.3.1 Overview 11

6.1.3.2 Resource: <resource 1> 12

6.1.3.2.1 Description 12

6.1.3.2.2 Resource Definition 12

6.1.3.2.3 Resource Standard Methods 12

6.1.3.2.3.1 < method 1 > 12

6.1.3.2.3.2 < method 2 > 13

6.1.3.2.4 Resource Custom Operations 13

6.1.3.2.4.1 Overview 14

6.1.3.2.4.2 Operation: < operation 1 > 14

6.1.3.2.4.2.1 Description 14

6.1.3.2.4.2.2 Operation Definition 14

6.1.3.2.4.3 Operation: < operation 2 > 14

6.1.3.3 Resource: <resource 2> 14

6.1.4 Custom Operations without associated resources 15

6.1.4.1 Overview 15

6.1.4.2 Operation: <operation 1> 15

6.1.4.2.1 Description 15

6.1.4.2.2 Operation Definition 15

6.1.4.3 Operation: < operation 2> 15

6.1.5 Notifications 16

6.1.5.1 General 16

6.1.5.2 <notification 1> 16

6.1.5.2.1 Description 16

6.1.5.2.2 Target URI 16

6.1.5.2.3 Standard Methods 16

6.1.5.2.3.1 POST 16

6.1.5.3 <notification 2> 17

6.1.6 Data Model 17

6.1.6.1 General 17

6.1.6.2 Structured data types 17

6.1.6.2.1 Introduction 17

6.1.6.2.2 Type: <TypeName 1> 17

6.1.6.2.3 Type: <TypeName 2> 18

6.1.6.3 Simple data types and enumerations 18

6.1.6.3.1 Introduction 18

6.1.6.3.2 Simple data types 18

6.1.6.3.3 Enumeration: <EnumType1> 19

6.1.6.3.4 Enumeration: <EnumType2> 19

6.1.6.4 Data types describing alternative data types or combinations of data types 19

6.1.6.4.1 Type: <TypeName 1> 19

6.1.6.4.2 Type: <TypeName 2> 20

6.1.6.5 Binary data 20

6.1.6.5.1 Binary Data Types 20

6.1.7 Error Handling 20

6.1.7.1 General 20

6.1.7.2 Protocol Errors 20

6.1.7.3 Application Errors 20

6.1.8 Feature negotiation 21

6.1.9 Security 21

6.2 < Service 2> Service API 21

Annex A (normative): OpenAPI specification 22

A.1 General 22

A.2 <Service 1> API 22

A.3 <Service 2> API 25

Annex B (informative): Change history 26

# 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.

# Introduction

This clause is optional. If it exists, it is always the second unnumbered clause.

# 1 Scope

This clause will describe the scope of the corresponding service specification.

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

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".

# 3 Definitions, symbols 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].

Definition format (Normal)

**<defined term>:** <definition>.

**example:** text used to clarify abstract rules by applying them literally.

## 3.2 Symbols

For the purposes of the present document, the following symbols apply:

Symbol format (EW)

<symbol> <Explanation>

## 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].

Abbreviation format (EW)

<ACRONYM> <Explanation>

# 4 Services offered by the AAnF

## 4.1 Introduction

This clause will list the different services produced by the NF.

Table 4.1-x summarizes the corresponding APIs defined for this specification.

Table 4.1-x: API Descriptions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Service Name** | **Clause** | **Description** | **OpenAPI Specification File** | **apiName** | **Annex** |
| <service name> | <ref clause> | <short description as included in the OpenAPI file> | <file name> | <apiName in the URI> | <ref Annex> |

## 4.2 Naanf\_AKMA Service

### 4.2.1 Service Description

This clause will provide a general description of the related service, include a description of the functional elements involved in the invocation of the service, i.e. NF Service Producer and NF Service Consumer(s), and list the service operations it supports.

### 4.2.2 Service Operations

One clause per service operation.

This clause will include a description of the different service operationssupported by the service. For RESTful service operations, the service operations depict the resources and the methods they support.

#### 4.2.2.1 Introduction

This clause will contain a generic introduction of the service operationsdescribed in the following clauses.

#### 4.2.2.2 Naanf\_AKMA\_AnchorKey\_Register service operation

##### 4.2.2.2.1 General

##### 4.2.2.2.n <Procedure n using Naanf\_AKMA\_AnchorKey\_Register service operation>

#### 4.2.2.3 Naanf\_AKMA\_ApplicationKey\_Get service operation

##### 4.2.2.2.1 General

##### 4.2.2.2.n <Procedure n using Naanf\_AKMA\_ApplicationKey\_Get service operation>

# 5 API Definitions

## 5.1 Naanf\_AKMA Service API

One clause per service, where <service 1> is to be replaced by the service name (e.g. Nsmf\_PDUSession).

### 5.1.1 Introduction

This clause specifies the API Name and Version.

The <Service 1> shall use the <Service 1> API.

The API URI of the <Service 1> API shall be:

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

The request URIs 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 "<service 1 API name>".

- The <apiVersion> shall be "v1".

- The <apiSpecificResourceUriPart> shall be set as described in clause 5.3.

### 5.1.2 Usage of HTTP

#### 5.1.2.1 General

This clause will include a reference to TS 29.500 for the description of the Transport and HTTP/2.0 protocol requirements and for the security requirements.

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 Naanf\_AKMA API is contained in Annex A.

#### 5.1.2.2 HTTP standard headers

##### 5.1.2.2.1 General

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

Add specific information for the API if applicable.

##### 5.1.2.2.2 Content type

This clause will indicate the encoding of HTTP requests/responses and the applicable MIME media type for the related Content-Type header. Adjust the text below if additional payload types are used e.g. for HATEOS.

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].

#### 5.1.2.3 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.

Add specific information for the API if applicable.

### 5.1.3 Resources

#### 5.1.3.1 Overview

This clause will describe the structure for the Resource URIs and the resources and methods used for the service.

Example:



Figure 5.1.3.1-1: Resource URI structure of the Naanf\_AKMA API

Table 5.1.3.1-1 provides an overview of the resources and applicable HTTP methods.

Table 5.1.3.1-1: Resources and methods overview

|  |  |  |  |
| --- | --- | --- | --- |
| Resource name | Resource URI | HTTP method or custom operation | Description |
| <Resource name> | <relative URI below root> | GET | <Operation executed by GET> |
| PUT | <Operation executed by PUT> |
| PATCH | <Operation executed by PATCH> |
| POST | <Operation executed by POST> |
| DELETE | <Operation executed by DELETE> |
| Custom operation | <Operation executed by custom operation> |

#### 5.1.3.2 Resource: <resource 1>

Where <resource 1> is to be replaced by the resource name, e.g. PduSession.

##### 5.1.3.2.1 Description

This clause will specify what the resource represents or what it is used for.

#### 5.1.3.2.2 Resource Definition

This clause will describe the Resource URI and the supported resource variables.

Resource URI: **{apiRoot}/<apiName>/<apiVersion>/xxx**

This resource shall support the resource URI variables defined in table 5.1.3.2.2-1.

Table 5.1.3.2.2-1: Resource URI variables for this resource

|  |  |  |
| --- | --- | --- |
| Name | Data type | Definition |
| apiRoot | string | See clause 5.1.1 |
| apiVersion | string | See clause 5.1.1 |
| <name> | <type> | <definition> |

##### 5.1.3.2.3 Resource Standard Methods

The following clauses will specify the standard methods supported by the resource.

It will describe, for each method, the use of the method, the URI query parameters supported by the method, request and response data structures and response codes, and if applicable, HTTP headers specific to the operation.

###### 5.1.3.2.3.1 < method 1 >

This clause will specify the meaning of the method applied on the resource.

This method shall support the URI query parameters specified in table 5.1.3.2.3.1-1.

Table 5.1.3.2.3.1-1: URI query parameters supported by the <method 1> method on this resource

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Data type | P | Cardinality | Description | Applicability |
| <name> or n/a | <type> or <leave empty> | <M, C or O> | 0..1 or 1 or 0..N or 1..N or <leave empty> | <only if applicable> |  |

This method shall support the request data structures specified in table 5.1.3.2.3.1-2 and the response data structures and response codes specified in table 5.1.3.2.3.1-3.

Table 5.1.3.2.3.1-2: Data structures supported by the <method 1> Request Body on this resource

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | P | Cardinality | Description |
| "<type>" or "array*(<type>*)" or "map*(<type>*)" or n/a | "M", "C" or "O" | "0..1", "1", or "M..N", or <leave empty> | <only if applicable> |

Table 5.1.3.2.3.1-3: Data structures supported by the <method 1> Response Body on this resource

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data type | P | Cardinality | Response  codes | Description |
| "*<type>*" or "array*(<type>*)" or "map*(<type>*)" or n/a | "M", "C" or "O" | "0..1", "1", or "M..N", or <leave empty> | <list applicable codes with name from the applicable RFCs> | <Meaning of the success case>  or  <Meaning of the error case with additional statement regarding error handling> |
| NOTE: The manadatory HTTP error status code for the <method 1> method listed in Table 5.2.7.1-1 of 3GPP TS 29.500 [4] also apply. | | | | |

Table 5.1.3.2.3.1-4: Headers supported by the <e.g. GET> method on this resource

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Data type | P | Cardinality | Description |
| <header name> | <data type>  e.g. string | "M", "C" or "O" | "0..1", "1", "1..N", "1..N", or <leave empty> | <description> |

Table 5.1.3.2.3.1-5: Headers supported by the <e.g. 200> response code on this resource

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Data type | P | Cardinality | Description |
| <header name> | <data type>  e.g. string | "M", "C" or "O" | "0..1", "1", "1..N", "1..N", or <leave empty> | <description> |

Table 5.1.3.2.3.1-6: Links supported by the 200 Response Code on this endpoint

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Resource name | HTTP method or custom operation | Link parameter(s) | Description |
| <link name>  e.g. search | <resource 1>  e.g. Stored Search (Document) | <method 1>  e.g. GET | <parameter>  e.g. searchId | <description of the link> |

###### 5.1.3.2.3.2 < method 2 >

And so on if there are more than two methods supported by the resource. Same structure as in clause 5.1.3.2.3.1.

##### 5.1.3.2.4 Resource Custom Operations

The following clauses will specify the custom operations supported by the resource.

It will describe, for each custom operation, the use and the URI of the operation, the HTTP method on which it is mapped, request and response data structures and response codes, and if applicable, HTTP headers specific to the operation.

###### 5.1.3.2.4.1 Overview

Table 5.1.3.2.4.1-1: Custom operations

|  |  |  |  |
| --- | --- | --- | --- |
| Operation name | Custom operaration URI | Mapped HTTP method | Description |
| <custom operation name> | <custom operation URI> | e.g.POST | <Operation executed by Custom operation> |
|  |  |  |  |

###### 5.1.3.2.4.2 Operation: < operation 1 >

This clause will specify the meaning of the operation applied on the resource.

5.1.3.2.4.2.1 Description

This sublause will describe the custom operation and what it is used for, and the custom operation's URI.

5.1.3.2.4.2.2 Operation Definition

This clause will specify the custom operation and the HTTP method on which it is mapped.

This operation shall support the request data structures specified in table 5.1.3.2.4.2.2-1 and the response data structure and response codes specified in table 5.1.3.2.4.2.2-2.

Table 5.1.3.2.4.2.2-1: Data structures supported by the <e.g. POST> Request Body on this resource

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | P | Cardinality | Description |
| "*<type>*" or "array*(<type>*)" or "map*(<type>*)" | "M", "C" or "O" | "0..1", "1", or "M..N", or <leave empty> | <only if applicable> |

Table 5.1.3.2.4.2.2-2: Data structures supported by the <e.g. POST> Response Body on this resource

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data type | P | Cardinality | Response  codes | Description |
| "*<type>*" or "array*(<type>*)" or "map*(<type>*)" | "M", "C" or "O" | "0..1", "1" or "M..N", or <leave empty> | <list applicable codes with name from the applicable RFCs> | <Meaning of the success case>  or  <Meaning of the error case with additional statement regarding error handling> |
| NOTE: The manadatory HTTP error status code for the <e.g. POST> method listed in Table 5.2.7.1-1 of 3GPP TS 29.500 [4] also apply. | | | | |

###### 5.1.3.2.4.3 Operation: < operation 2 >

And so on if there are more than two operations supported by the resource. Same structure as in clause 5.1.3.2.4.1.

#### 5.1.3.3 Resource: <resource 2>

And so on if there are more than two resources supported by the service. Same structure as in clause 5.1.3.2.

### 5.1.4 Custom Operations without associated resources

#### 5.1.4.1 Overview

This clause will specify custom operations without any associated resource (i.e. RPC) supported by this API.

Table 5.1.4.1-1: Custom operations without associated resources

|  |  |  |
| --- | --- | --- |
| Custom operation URI | Mapped HTTP method | Description |
| <custom operation URI> | e.g.POST | <Operation executed by Custom operation> |
|  |  |  |

#### 5.1.4.2 Operation: <operation 1>

Where <operation 1> is to be replaced by the name of the custom operation, e.g. Authentication\_Information\_Request.

It will describe, for each custom operation, the use and the URI of the operation, the HTTP method on which it is mapped, request and response data structures and response codes, and if applicable, HTTP headers specific to the operation.

##### 5.1.4.2.1 Description

This sublause will describe the custom operation and what it is used for, and the custom operation's URI.

##### 5.1.4.2.2 Operation Definition

This clause will specify the custom operation and the HTTP method on which it is mapped.

This operation shall support the response data structures and response codes specified in tables 5.1.4.2.2-1 and 5.1.4.2.2-2.

Table 5.1.4.2.2-1: Data structures supported by the <e.g. POST> Request Body on this resource

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | P | Cardinality | Description |
| "*<type>*" or "array*(<type>*)" or "map*(<type>*)" | "M", "C" or "O" | "0..1", "1", or "M..N", or <leave empty> | <only if applicable> |

Table 5.1.4.2.2-2: Data structures supported by the <e.g. POST> Response Body on this resource

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data type | P | Cardinality | Response  codes | Description |
| "*<type>*" or "array*(<type>*)" or "map*(<type>*)" | "M", "C" or "O" | "0..1", "1" or "M..N", or <leave empty> | <list applicable codes with name from the applicable RFCs> | <Meaning of the success case>  or  <Meaning of the error case with additional statement regarding error handling> |
| NOTE: The manadatory HTTP error status code for the <e.g. POST> method listed in Table 5.2.7.1-1 of 3GPP TS 29.500 [4] also apply. | | | | |

#### 5.1.4.3 Operation: < operation 2>

And so on if there are more than one custom operations supported by the service. Same structure as in clause 5.1.4.2.

### 5.1.5 Notifications

#### 5.1.5.1 General

This clause will specify the use of notifications and corresponding protocol details if required for the specific service. When notifications are supported by the API, it will include a reference to the general description of notifications support over the 5G SBIs specified in TS 29.500 / TS 29.501.

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].

Table 5.1.5.1-1: Notifications overview

|  |  |  |  |
| --- | --- | --- | --- |
| Notification | Callback URI | HTTP method or custom operation | Description  (service operation) |
| <notification 1>  e.g. Status Change Notification | < Callback URI >  e.g. {StatusCallbackUri} | e.g POST | e.g. Notify Event |
|  |  |  |  |

#### 5.1.5.2 <notification 1>

##### 5.1.5.2.1 Description

The Event Notification is used by the NF service producer to report one or several observed Events to a NF service consumer that has subscribed to such Notifications.

##### 5.1.5.2.2 Target URI

The Callback URI **"{notifUri}"** shall be used with the callback URI variables defined in table 5.1.5.2.2-1.

Table 5.1.5.2.2-1: Callback URI variables

|  |  |
| --- | --- |
| Name | Definition |
| notifUri | String formatted as URI with the Callback Uri |

##### 5.1.5.2.3 Standard Methods

###### 5.1.5.2.3.1 POST

This method shall support the request data structures specified in table 5.1.5.2.3.1-1 and the response data structures and response codes specified in table 5.1.5.2.3.1-1.

Table 5.1.5.2.3.1-2: Data structures supported by the POST Request Body

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | P | Cardinality | Description |
| "*<type>*" or "array*(<type>*)" or "map*(<type>*)" | "M", "C" or "O" | "0..1", "1", or "M..N", or <leave empty> | <only if applicable> |

Table 5.1.5.2.3.1-3: Data structures supported by the POST Response Body

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data type | P | Cardinality | Response codes | Description |
| "*<type>*" or "array*(<type>*)" or "map*(<type>*)" | "M", "C" or "O" | "0..1", "1" or "M..N", or <leave empty> | <list applicable codes with name from the applicable RFCs> | <Meaning of the success case>  or  <Meaning of the error case with additional statement regarding error handling> |
| 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. | | | | |

#### 5.1.5.3 <notification 2>

And so on if there are more than one notifications supported by the service. Same structure as in clause 5.1.5.2.

### 5.1.6 Data Model

#### 5.1.6.1 General

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

Data types that may be common to multiple APIs (offered by the same or different NFs) should be specified in a new separate TS (similar approach as for TS 29.230 for Diameter AVPs).

Table 5.1.6.1-1 specifies the data types defined for the N<NF> service based interface protocol.

Table 5.1.6.1-1: N<NF> specific Data Types

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | Clause defined | Description | Applicability |
|  |  |  |  |

Table 5.1.6.1-2 specifies data types re-used by the N<NF> 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<NF> service based interface.

Table 5.1.6.1-2: N<NF> re-used Data Types

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | Reference | Comments | Applicability |
|  |  |  |  |

#### 5.1.6.2 Structured data types

This clause will specify the structured data types.

##### 5.1.6.2.1 Introduction

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

##### 5.1.6.2.2 Type: <TypeName 1>

"Attribute name": Name of attributes that belong to the specified data type. The attribute names within a structured data type shall be unique, and their relative order inside the structured data type shall not imply any specific ordering of the corresponding JSON member names in a JSON object.

"Data type": Data type of the attribute values. If the data type is indicated as "<type>", the attribute value shall be of data type <type>. If the data type is indicated as "array(<type>)", the attribute value shall be an array (see IETF RFC 825]) that contains elements of data type <type>. If the data type is indicated as "map(<type>)", the attribute value shall be an object (see IETF RFC 8259) encoded in the corresponding OpenAPI specification as a map which values are data type <type>. <type> can either be "integer", "number", "string" or "boolean" (as defined in the OpenAPI specification [4]), or a data type defined in a 3GPP specification.

"P": Presence condition of a data structure in request body. It shall be one of "M" (for Mandatory), "C" (for Conditional) and "O" (for Optional).

"Cardinality": Defines the allowed number of occurrence of data type <type>. A cardinality of "M..N", is only allowed for data types "array(<type>)" and "map(<type>)" and indicates the number of elements within the array or map; the values M and N can either be the characters "M" and "N", respectively, or integer numbers; with M being greater than or equal 0, and N being greater than 0 and M. For data type "<type>", the cardinality shall be set to "0..1" if the Presence condition is "C" or "O", and to "1" if the Presence condition is "M".

"Description": Describes the meaning and use of the attribute and may contain normative statements..

Applicability: If the attribute is only applicable for optional feature(s) negotiated using the mechanism defined in clause 6.6 of 3GPP TS 29.500 [4], the name of the corresponding feature(s) shall be indicated in this column. If no feature is indicated. the attribute can be used with any feature.If no optional features are defined for an API, the applicability column can be omitted for that API

Table 5.1.6.2.2-1: Definition of type <TypeName 1>

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | Data type | P | Cardinality | Description | Applicability |
| <*attribute name*> | "*<type>*" or "array*(<type>*)" or "map*(<type>*)" | "M", "C" or "O" | "0..1", "1" or "M..N" | <only if applicable> |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

##### 5.1.6.2.3 Type: <TypeName 2>

And so on if there are more types to specify.

#### 5.1.6.3 Simple data types and enumerations

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

##### 5.1.6.3.1 Introduction

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

##### 5.1.6.3.2 Simple data types

The simple data types defined in table 5.1.6.3.2-1 shall be supported.

Table 5.1.6.3.2-1: Simple data types

|  |  |  |  |
| --- | --- | --- | --- |
| Type Name | Type Definition | Description | Applicability |
|  | <one simple data type, i.e. boolean, integer, number, or string> |  |  |

##### 5.1.6.3.3 Enumeration: <EnumType1>

The enumeration <EnumType1> represents <something>. It shall comply with the provisions defined in table 5.1.6.3.3-1.

Table 5.1.6.3.3-1: Enumeration < EnumType1>

|  |  |  |
| --- | --- | --- |
| Enumeration value | Description | Applicability |
|  |  |  |

##### 5.1.6.3.4 Enumeration: <EnumType2>

And so on if there are more enumerations to define.

#### 5.1.6.4 Data types describing alternative data types or combinations of data types

##### 5.1.6.4.1 Type: <TypeName 1>

The data types describing alternative data types or combinations of data types shall represent an OpenAPI schema object using the "oneOf", "anyOf" or "allOf" keyword to list alternative or to be combined data types (see the OpenAPI specification [4] and https://swagger.io/docs/specification/data-models/oneof-anyof-allof-not/).

An instance (i.e. a corresponding part of a JSON file to be evaluated against the schema) matches , i.e. a list of mutually exclusive alternative data types., as described using the OpenAPI "oneOf" keyword, if the instance matches against one and only one of the alternative data types.

An instance (i.e. a corresponding part of a JSON file to be evaluated against the schema) matches a list of non-exclusive alternative data types, as described using the OpenAPI "anyOf" keyword, if the instance matches against one or more of the alternative data types.

An instance (i.e. a corresponding part of a JSON file to be evaluated against the schema) matches a list of to be combined data types, as described using the OpenAPI "allOf" keyword, if the instance matches against all of the to be combined data types."Attribute name": Name of attributes that belong to the specified data type. The attribute names within a structured data type shall be unique, and their relative order inside the structured data type shall not imply any specific ordering of the corresponding JSON elements in a JSON object.

"Data type": Data type of the alternative or to be combined data type. If the data type is indicated as "<type>", the alternative or to be combined data type shall be of data type <type>. If the data type is indicated as "array(<type>)", the alternative or to be combined data type shall be an array (see IETF RFC 8259 [3]) that contains elements of data type <type>. If the data type is indicated as "map(<type>)", the alternative or to be combined data type shall be an object (see IETF RFC 8259 [3]) encoded in the corresponding OpenAPI specification as a map which values are of data type <type>. <type> can either be "integer", "number", "string" or "boolean" (as defined in the OpenAPI specification [4]), or a data type defined in a 3GPP specification.

"Cardinality": Defines the allowed number of occurrence of data type <type>. A cardinality of "M..N", is only allowed for data types "array(<type>)" and "map(<type>)" and indicates the number of elements within the array or map; the values M and N can either be the characters "M" and "N", respectively, or integer numbers; with M being greater than or equal 0, and N being greater than 0. For data type "<type>", the cardinality shall be set to "1".

"Description": Describes the meaning and use of the attribute and may contain normative statements.Applicability: If the attribute is only applicable for optional feature(s) negotiated using the mechanism defined in clause 6.6 of 3GPP TS 29.500 [4], the name of the corresponding feature(s) shall be indicated in this column. If no feature is indicated. the attribute can be used with any feature.

Applicability: If the type is only applicable for optional feature(s) negotiated using the mechanism defined in clause 6.6 of 3GPP TS 29.500 [2], the name of the corresponding feature(s) shall be indicated in this column. If no feature is indicated. the type can be used with any feature. If no optional features are defined for an API, the applicability column can be omitted for that API.

Table 5.1.6.4.1-1: Definition of type <Type name 1> as a list of <"mutually exclusive alternatives" / "non-exclusive alternatives" / "to be combined data types">

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | Cardinality | Description | Applicability |
| "*<type>*" or "array*(<type>*)" or "map*(<type>*)" | "1" or "*M*..*N*" | <only if applicable> |  |

##### 5.1.6.4.2 Type: <TypeName 2>

And so on if there are more types to specify.

#### 5.1.6.5 Binary data

This clause will specify what is encoded in binary part, if multipart media type is agreed to be supported by CT4 and is supported by the API. It shall be omitted if not applicable.

##### 5.1.6.5.1 Binary Data Types

Table 5.1.6.5.1-1: Binary Data Types

|  |  |  |
| --- | --- | --- |
| Name | Clause defined | Content type |
| < Binary Data 1 >  e.g. N1 SM Message | < clause >  e.g. 5.1.6.5.2 | <content type>  e.g. vnd.3gpp.5gnas |
|  |  |  |

5.1.6.5.2 < Binary Data 1 >

*And so on if there are more binary data to specify*

### 5.1.7 Error Handling

This clause will include a reference to the general error handling principles specified in TS 29.501, and further specify any general error handling aspect specific to the API, if any Error handling specific to each method (and resource) is specified in clauses 5.1.3. and 6.1.4.

#### 5.1.7.1 General

For the Naanf\_AKMA 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 Naanf\_AKMA API.

#### 5.1.7.2 Protocol Errors

No specific procedures for the Naanf\_AKMA service are specified.

Or add specific information for the API if applicable.

#### 5.1.7.3 Application Errors

The application errors defined for the Naanf\_AKMA service are listed in Table 5.1.7.3-1.

Table 5.1.7.3-1: Application errors

|  |  |  |
| --- | --- | --- |
| Application Error | HTTP status code | Description |
|  |  |  |

## 5.1.8 Feature negotiation

The optional features in table 5.1.8-1 are defined for the Naanf\_AKMA API. They shall be negotiated using the extensibility mechanism defined in clause 6.6 of 3GPP TS 29.500 [4].

Table 5.1.8-1: Supported Features

|  |  |  |
| --- | --- | --- |
| Feature number | Feature Name | Description |
|  |  |  |

## 5.1.9 Security

As indicated in 3GPP TS 33.501 [8] and 3GPP TS 29.500 [4], the access to the Naanf\_AKMA 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 Naanf\_AKMA 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 Naanf\_AKMA service.

The Naanf\_AKMA API defines a single scope " naanf-akma" 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 1: 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 Naanf\_AKMA API

Annex <X> (informative):  
Change history

This is the last annex for TS/TSs which details the change history using the following table.  
This table is to be used for recording progress during the WG drafting process till TSG approval of this TS/TR.  
For TRs under change control, use one line per approved Change Request  
Date: use format YYYY-MM  
CR: four digits, leading zeros as necessary  
Rev: blank, or number (max two digits)  
Cat: use one of the letters A, B, C, D, F  
Subject/Comment: for TSs under change control, include full text of the subject field of the Change Request cover  
New vers: use format [n]n.[n]n.[n]n

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Change history** | | | | | | | |
| **Date** | **Meeting** | **TDoc** | **CR** | **Rev** | **Cat** | **Subject/Comment** | **New version** |
|  |  |  |  |  |  |  |  |

Change history of this template:

(Shall be removed when a draft specification is produced)

|  |  |  |
| --- | --- | --- |
| 2017-10-23 | Initial Draft. | 0.1.0 |
| 2019-06 | Template with OpenAPI priority | 0.2.0 |
| 2019-06 | TS template update | 0.3.0 |
| 2019-12 | Add API description table in Clause 5.1 | 0.4.0 |
| 2020-03 | New tables added in template | 0.5.0 |
| 2020-03 | An update of TS clause A.1 | 0.5.0 |
| 2020-06 | API URI correction and template udpate | 0.6.0 |
| 2020-09 | Storage of YAML files in 3GPP Forge and updates in 6.1.5. | 0.7.0 |