ETSI TS 104 231 V8.0.0 (2026-03)

TECHNICAL SPECIFICATION

Publicly Available Specification (PAS); O-RAN R1 interface Application Protocols for R1 Services (O-RAN.WG2.TS.R1AP-R004-v08.00)

CAUTION

The present document has been submitted to ETSI as a PAS produced by O-RAN Alliance and approved by the ETSI Technical Committee Mobile Standards Group (MSG).

Reference
DTS/MSG-001175

Keywords
interface, PAS

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

Contents

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 IPR online database.

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™**, **LTE™** and **5G™** logo 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.

Foreword

This Technical Specification (TS) has been produced by O-RAN Alliance and approved by ETSI Technical Committee Mobile Standards Group (MSG).

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.

1 Scope

The present document specifies the Application Protocols for R1 Services.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found in the ETSI docbox.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are necessary for the application of the present document.

[1] ETSI TS 129 501: "5G; 5G System; Principles and Guidelines for Services Definition; Stage 3".

[2] ETSI TS 129 500: "5G; 5G System; Technical Realization of Service Based Architecture; Stage 3".

[3] OpenAPI: "OpenAPI Specification v3.0.3".

[4] ETSI GS NFV-SOL 013: "Network Functions Virtualisation (NFV) Release 5; Protocols and Data Models; Specification of common aspects for RESTful NFV MANO APIs".

[5] O-RAN.WG2.TS.R1GAP-R004: "R1 interface: General Aspects and Principles" ("R1GAP").

[6] O-RAN.WG2.TS.R1UCR-R004 "R1 interface: Use Cases and Requirements" ("R1UCR").

[7] O-RAN.WG2.TS.R1TP-R004: "Transport Protocols for R1 Services" ("R1TP").

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

[9] ETSI TS 129 222 (V18.6.0): "LTE; 5G; Common API Framework for 3GPP Northbound APIs".

[10] IETF RFC 7807: "Problem Details for HTTP APIs".

[11] Semantic Versioning 2.0.0.

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

[13] IETF RFC 4229: "HTTP Header Field Registrations".

[14] json-schema 2020-12.

[15] Kafka Documentation.

[16] W3C® Recommendation-xmlschema-1 (2001/05/02): "XML Schema Part 1: Structures".

[17] W3C® Recommendation -xmlschema-2 (2001/05/02): "XML Schema Part 2: Datatypes".

[18] W3C® Recommendation -xml-names (1999/01/14): "Namespaces in XML".

[19] IETF RFC 1035: "Domain Names - implementation and specification".

[20] ETSI TS 128 532: "5G; Management and orchestration; Generic management services".

[21] ETSI TS 128 622: "Universal Mobile Telecommunications System (UMTS); LTE; 5G; Telecommunication management; Generic Network Resource Model (NRM) Integration Reference Point (IRP); Information Service (IS)".

[22] Void.

[23] O-RAN.WG2.TS.A1AP-R004: "A1 interface: Application Protocol" ("A1AP").

[24] O-RAN.WG2.TS.A1TD-R004: "A1 interface: Type Definitions" ("A1TD").

[25] IETF RFC 9110: "HTTP Semantics".

[26] ETSI TS 128 111: "5G; Management and orchestration; Fault management (FM)".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents may be useful in implementing an ETSI deliverable or add to the reader's understanding, but are not required for conformance to the present document.

[i.1] W3C®: Recommendation xmlschema -0 (2001/05/02): "XML Schema Part 0: Primer".

3 Definition of terms, symbols and abbreviations

3.1 Terms

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

API Consumer: Service Consumer consuming one or more services using APIs

NOTE: The Service Consumer role is introduced in R1GAP [5].

API Producer: Service Producer that offers its services for consumption via APIs

NOTE: The Service Producer role is introduced in R1GAP [5].

DME type: data type managed and exposed by the DME services and identified by a data type identifier

3.2 Symbols

Void.

3.3 Abbreviations

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

AEF	API Expsoing Function
AI	Artificial Intelligence
AI/ML	Artificial Intelligence/Machine Learning
AIML	Artificial Intelligence/Machine Learning
AP	Application Protocol
APF	API Producing Function
API	Application Programming Interface

API	Application Programming Interface
CAPIF	Common API Framework
CM	Configuration Management
DME	Data Management and Exposure
DNS	Domain Name System
FM	Fault Management
FQDN	Fully Qualified Domain Name
HTTP	Hypertext Transfer Protocol
HTTP/1	Hypertext Transfer Protocol
IOC	Information Object Class
JSON	JavaScript Object Notation
LDN	Local Distinguished Name
MANO	Management and Orchestration
MnS	Management Services
MOI	Managed Object Instance
Non-RT RIC	Non-Real Time Intelligent Controller
OAM	Operation And Maintenance
O-RU	O-RAN Radio Unit
PM	Performance Measurement
R1UCR	R1 Use-Case and Requirements
RAN	Radio Access Network
RT	Real Time
SME	Service Management and Exposure
SMO	Service Management and Orchestration
URI	Uniform Resource Identifier
XML	eXtensible Markup Language
YAML	Yet Another Markup Language
YANG	Yet Another Next Generation

4 Application protocol for the R1 services

4.1 Introduction

The present document contains a realization for the procedures identified in R1GAP [5]. It is based on transport protocols as defined in R1TP [7]. This definition of the R1 Application Protocols (R1AP) defined in the present document is based on the 3GPP service framework for network functions specified in ETSI TS 129 501 [1].

4.2 Version conventions for the present document

The version number of the present document follows the "xx.yy" versioning scheme. There could be implications for the interoperability between rApps and R1 service API implementations in SMO/Non-RT RIC framework that are based on different versions of the present document.

An incremented "xx" version field of the present document could indicate that a new major feature (e.g. a new R1 service) has been added or that an incompatible change has been made to one or more R1 service APIs. An incremented "yy" version field could indicate that an optional feature has been added, a technical issue has been fixed, or that clarifications or editorial corrections have been made.

The version conventions for RESTful R1 service APIs are defined in clause 5.2.

5 RESTful R1 service APIs

5.1 Overview

The design of the RESTful R1 service APIs is based on the procedures and requirements defined in R1UCR [6] and R1GAP [5], and on the protocol design framework as specified in ETSI TS 129 501 [1].

The API version of a service API includes a pre-release version (e.g. "-alpha.1") if the service API is under development.

The present document defines the protocols for the R1 service APIs listed in Table 5.1-1.

Table 5.1-1: RESTful R1 service APIs and their versions defined in the present document

R1 Services	Service API	API Version
Service management and exposure services	Service registration	1.2.0
	Service discovery	1.2.0
	Service events subscription	1.2.0
	Bootstrap	1.0.0
Data management and exposure services	Data registration	2.0.0-alpha.2
	Data discovery	2.0.0
	Data access	2.0.0-alpha.2
	HTTP based push data	1.0.0
	HTTP based pull data	1.0.0
	Data offer	1.0.0
RAN OAM-related services	Configuration management	1.0.0-alpha.1
	Fault management	1.0.0
	Configuration schema information	1.0.0-alpha.1
A1-related services	A1 policy management	1.0.0
AI/ML workflow services	AI/ML model registration	1.0.0
	AI/ML model discovery	1.0.0
	AI/ML model training	1.0.0-alpha.1
	AI/ML model deployment	1.0.0-alpha.1
	AI/ML model retrieve	1.0.0-alpha.1

5.2 Versioning of RESTful R1 service APIs

Each RESTful R1 service API is versioned independently. The API version number defined in the present document contains three numerical fields following a MAJOR.MINOR.PATCH pattern, and may contain a pre-release version field, according to SemVer [11].

The API version number held by an implementation may additionally include a build metadata field, according to SemVer [11], to indicate a specific deployment. The content of this field is implementation specific; it is provided by the deployment. The <apiVersion> path segment used in URI structures indicate the MAJOR field of the API version number. The full API version number is visible in the "version" field of the "info" object of each OpenAPI document in Annex A, as well as in the ServiceAPIDescription information communicated during service registration and service discovery (see clause B.3.4.1).

To indicate the full API version the API Consumer intends to use, the API Consumer may include the "Version" HTTP header (see IETF RFC 4229 [13]) in an HTTP request, in which case the header shall contain the version identifier as defined above. It is optional to include the build metadata field.

The API Producer shall include in the response the "Version" HTTP header signalling the used API version, including the build metadata if available. If the build metadata have been omitted in the request, the API Producer shall use the combination of MAJOR, MINOR, PATCH, and pre-release indicator as requested and the highest supported value for the build metadata field for that combination, if available. In case the API Consumer has not sent a "Version" header in the request, the API Producer shall use the latest available version, and signal it in the "Version" header.

NOTE: In case multiple versions are supported by an API Producer under the URI for a major version, this allows the API Consumer to request a particular version. This mechanism is referred to as "microversioning".

If the API version signalled by the API Consumer in the "Version" request header is not supported by the API Producer, the API Producer shall respond with a "406 Not Acceptable" error and may include in the response payload body a Problem Details structure providing more information on the cause of the error.

5.3 URI structure and supported content formats

This clause specifies the URI prefix and the supported content formats applicable to the RESTful R1 service APIs.

All resource URIs of the APIs shall have the following prefix:

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

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure defined in clause 4.4.1 of ETSI TS 129 501 [1], i.e.:

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

with the following components:

- The {apiRoot} shall be set as described in ETSI TS 129 501 [1], clause 4.4.1; however, the restrictions w.r.t. the operator specific FQDN of the host portion defined there do not apply.
- The <apiName> indicates the API name of the service interface in an abbreviated form. It is defined in the clause specifying the corresponding RESTful R1 service API.
- The <apiVersion> indicates the major version (see clause 5.2) of the API and is defined in the clause specifying the corresponding RESTful R1 serviceAPI.
- Each <apiSpecificResourceUriPart> represents a specific resource of the API. It is defined in the corresponding RESTful R1 service API for each one of the defined resources.

For HTTP requests and responses that have message content, the content format JSON (see IETF RFC 8259 [12]) shall be supported. The JSON format shall be signalled by the content type "application/json".

All resource URIs of the API shall comply with the URI syntax as defined in IETF RFC 3986 [8]. An implementation that dynamically generates resource URI parts (individual path segments, sequences of path segments that are separated by "/", query parameter values) shall ensure that these parts only use the character set that is allowed by IETF RFC 3986 [8] for these parts.

5.4 General considerations for RESTful R1 service APIs

5.4.1 Usage of HTTP header fields

HTTP headers are components of the headers section of the HTTP request and response messages. The usage of HTTP header fields shall follow the definitions in ETSI GS NFV-SOL 013 [4], clause 4.2.

5.4.2 Handling of large query results

The handling of large query results shall be supported by RESTful R1 service APIs as specified in ETSI GS NFV-SOL 013 [4], clause 5.4.2.

5.4.3 Error reporting

In RESTful interfaces, application errors are mapped to HTTP errors. Since HTTP error information is generally not enough to discover the root cause of the error, additional application specific error information is typically delivered in the message content based on the ProblemDetails data type.

HTTP error responses shall be supported as specified in ETSI TS 129 501 [1], clause 4.8. Protocol errors and application errors specified in ETSI TS 129 500 [2], Table 5.2.7.1-1, shall be supported for an HTTP method if the corresponding HTTP status codes are specified as mandatory for that HTTP method in ETSI TS 129 500 [2], Table 5.2.7.1-1.

If an HTTP method is not defined for a particular resource in the present document, that method is not supported. When that method is requested on the resource, the API Producer shall return a "405 Method Not Allowed" response. The message content may include a ProblemDetails structure.

6 Service management and exposure services

6.1 Service registration API

6.1.1 Introduction

This API allows the API Consumer to manage registrations of service APIs based on the procedures for "Registration of services" defined in R1GAP [5].

6.1.2 API version

For the service registration API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 2 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v1".

6.1.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined for the CAPIF_Publish_Service_API (see ETSI TS 129 222 [9], Figure 8.2.2.1-1 in clause 8.2.2).

Table 6.1.3-1 lists the individual resources defined for the API, the applicable HTTP methods as defined in ETSI TS 129 222 [9], clause 8.2) and the associated service operations.

Table 6.1.3-1: Resources and methods overview of the service registration API

Resource Name ETSI TS 129 222 [9]	Resource URI ETSI TS 129 222 [9]	HTTP method ETSI TS 129 222 [9]	Service Operation
APF published APIs	/{apfId}/service-apis	POST	Register service API
		GET	Query service APIs
Individual APF published API	/{apfId}/service-apis/{serviceApiId}	PUT	Update registered service API
		DELETE	Deregister service API
		PATCH	Partial update registered service API

6.1.4 Service Operations

6.1.4.1 Register service API

6.1.4.1.1 Operation definition

A Service Producer uses the Register service API operation as API Consumer to register with the API Producer service APIs for services it is capable of producing.

The operation is based on HTTP POST.

Figure 6.1.4.1.1-1: Register service API operation

The service operation is as follows:

1) The API Consumer shall send an HTTP POST request to the API Producer. The target URI shall identify the resource (.../{apfId}/service-apis/) under which the new registration is requested to be created. The message content shall carry a ServiceAPIDescription structure. The API Producer shall process the message content received in the HTTP POST message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall generate the service API identifier and construct the URI for the created resource. The API Producer shall return the HTTP POST response. On success "201 Created" shall be returned. The "Location" header shall be present and shall carry the URI of the new registration resource. The message content shall carry the registered service API description. On failure, the appropriate error code shall be returned, and the message response body may contain additional error information.

6.1.4.1.2 Referenced procedures

6.1.4.1.2.1 Register service procedure

The Register service API operation illustrated in Figure 6.1.4.1.1-1 is based on the Register service procedure defined in R1GAP [5].

6.1.4.2 Update registered service API

6.1.4.2.1 Operation definition

A Service Producer uses the Update registered service API operation as API Consumer to update a complete service API registration with the API Producer.

The operation is based on HTTP PUT.

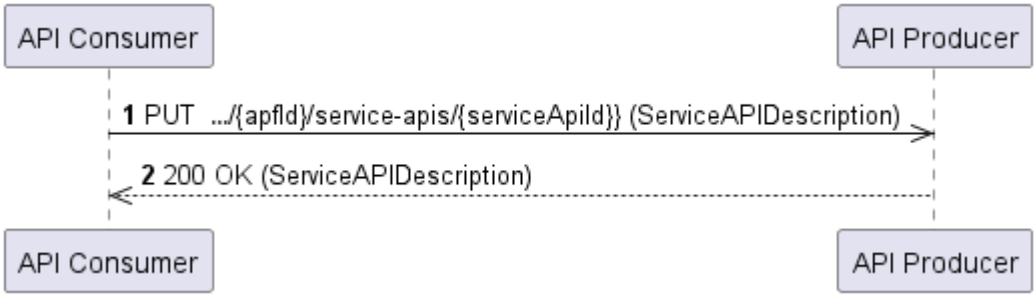

Figure 6.1.4.2.1-1: Update registered service API operation

The service operation is as follows:

1) The API Consumer shall send an HTTP PUT request to the API Producer. The target URI shall identify the resource (…/{apfId}/service-apis/{serviceApiId}). The message content shall carry an updated service API description. The API Producer shall determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP PUT response. On success "200 OK" shall be returned. The message content shall carry the updated service API description. On failure, the appropriate error code shall be returned, and the message response body may contain additional error information.

6.1.4.2.2 Referenced procedures

6.1.4.2.2.1 Update service registration procedure

The Update registered service API operation illustrated in Figure 6.1.4.2.1-1 is based on the Update service registration procedure defined in R1GAP [5].

6.1.4.3 Deregister service API

6.1.4.3.1 Operation definition

A Service Producer uses the Deregister service API operation as API Consumer to deregister a service with the API Producer.

The operation is based on HTTP DELETE.

Figure 6.1.4.3.1-1: Deregister service API operation

The service operation is as follows:

1) To deregister a service, The API Consumer shall send an HTTP DELETE. request to the API Producer. The target URI shall identify the resource to be deleted (…/{apfId}/service-apis/{serviceApiId}. The API Producer shall process the message content received in the HTTP DELETE message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP DELETE response. On success "204 No Content" shall be returned. On failure, the appropriate error code shall be returned, and the message response body may contain additional error information.

6.1.4.3.2 Referenced procedures

6.1.4.3.2.1 Deregister service procedure

The Deregister service API operation illustrated in Figure 6.1.4.3.1-1 is based on the Deregister service procedure defined in R1GAP [5].

6.1.4.4 Partially update registered service API

6.1.4.4.1 Operation definition

A Service Producer uses the Partially update registered service API operation as API Consumer to partially update a service API registration with the API Producer.

The operation is based on HTTP PATCH.

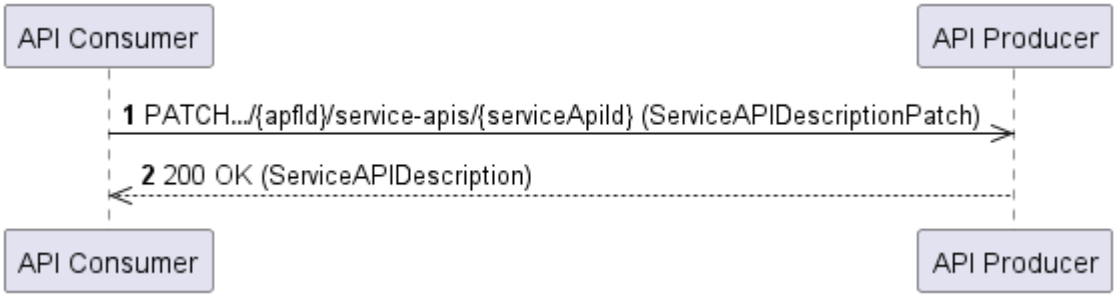

Figure 6.1.4.4.1-1: Partial update registered service API operation

The service operation is as follows:

1) The API Consumer shall send an HTTP PATCH request to the API Producer. The target URI shall identify the resource (…/{apfId}/service-apis/{serviceApiId}. The message content shall carry a service API description patch structure. The API Producer shall determine if the request sent by the API Consumer is authorized or not. If so, the API Producer shall process the message content received in the HTTP PATCH message.

2) The API Producer shall return the HTTP PATCH response. On success "200 OK" shall be returned. The message content shall carry the updated service API description. On failure, the appropriate error code shall be returned, and the message response body may contain additional error information.

6.1.4.4.2 Referenced procedures

6.1.4.4.2.1 Update service registration procedure

The Update registered service API operation illustrated in Figure 6.1.4.2.1-1 is based on the Update service registration procedure defined in R1GAP [5].

6.1.4.5 Query service APIs

6.1.4.5.1 Operation definition

The API Consumer uses the Query service APIs operation to retrieve the service API descriptions that are registered with the API Producer.

The operation is based on HTTP GET.

Figure 6.1.4.5.1-1: Query service API operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource (.../{apfId}/service-apis). The message content shall be empty. The API producer shall process the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the "200 OK" on success and the message content shall carry a list requested service API description. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

6.1.4.5.2 Referenced procedures

6.1.4.5.2.1 Query register service procedure

The Query service APIs operation illustrated in Figure 6.1.4.5.1-1 is based on the query registered service procedure defined in R1GAP [5].

6.1.5 Resources

6.1.5.1 Overview

This clause defines the resources for the service registration API based on ETSI TS 129 222 [9].

6.1.5.2 Resource: "APF published APIs"

6.1.5.2.0 Description

This resource is defined in ETSI TS 129 222 [9], clause 8.2 with the following URI:

{apiRoot}/published-apis/<apiVersion>/{apfId}/service-apis

By consuming this API, an rApp takes the role of the APF (API publishing function).

In addition to the provisions in ETSI TS 129 222 [9], clause 8.1, the following shall apply:

- The value of the {apfId} resource URI variable shall be set to the rAppId of the rApp that performs the service registration.

6.1.5.2.1 Resource Standard Methods

6.1.5.2.1.1 POST

This method shall support the URI query parameters, request data, response data structures and response codes specified in ETSI TS 129 222 [9], clause 8.2.2.2.3.1.

6.1.5.2.1.2 GET

This method shall support the URI query parameters, request data structures, response data structures, and response codes specified in ETSI TS 129 222 [9], clause 8.2.2.2.3.2.

6.1.5.2.2 Resource Custom Operations

None.

6.1.5.3 Resource: "Individual APF published API"

6.1.5.3.0 Description

This resource is defined in ETSI TS 129 222 [9], clause 8.2.2.3 with the following URI:

{apiRoot}/published-apis/<apiVersion>/{apfId}/service-apis/{serviceApiId}

By consuming this API, an rApp takes the role of the APF (API publishing function).

In addition to the provisions in ETSI TS 129 222 [9], clause 8.2, the following shall apply:

- The value of the {apfId} resource URI variable shall be set to the rAppId of the rApp that performed the service registration.

6.1.5.3.1 Resource Standard Methods

6.1.5.3.1.1 PUT

This method shall support the URI query parameters, request data structures, response data structures and response codes specified in ETSI TS 129 222 [9], clause 8.2.2.3.3.2.

6.1.5.3.1.2 DELETE

This method shall support the URI query parameters, request data, response data structures and response codes specified in ETSI TS 129 222 [9], clause 8.2.2.3.3.3.

6.1.5.3.1.3 PATCH

This method shall support the URI query parameters, request data structures, response data structures and response codes specified in ETSI TS 129 222 [9], clause 8.2.2.3.3.4.

6.1.5.3.2 Resource Custom Operations

None.

6.1.6 Custom operations without associated resources

None.

6.1.7 Notifications

None

6.1.8 Data model

6.1.8.1 General

The application data model is defined in ETSI TS 129 222 [9], clause 8.2.4. In additions, the adaptations specified in clause B.3 apply to this API.

6.1.8.2 Structured data types

None.

6.1.8.3 Simple data types and enumerations

None.

6.1.8.4 Re-used data types

The re-used data types are defined in clause B.3.

6.1.8.5 Service-specific registration information

None.

6.1.9 Error Handling

6.1.9.1 General

HTTP error handling is applicable for this API as specified in ETSI TS 129 222 [9], clause 7.7.

6.1.9.2 Protocol Errors

Protocol error handling shall be supported as specified in ETSI TS 129 222 [9], clause 8.1.5.2.

6.1.9.3 Application Errors

Application error handling shall be supported as specified in ETSI TS 129 222 [9], clause 8.2.5.3.

6.2 Service discovery API

6.2.1 Introduction

This API allows the API Consumer to perform service discovery based on the service discovery procedures defined in R1GAP [5]. The API is based on the CAPIF_Discover_Service_API as specified in ETSI TS 129 222 [9].

6.2.2 API version

For the service discovery API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 2 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v1".

6.2.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined for the CAPIF_Discovery_Service_API (see ETSI TS 129 222 [9], Figure 8.1.2.1-1 in clause 8.1.2).

Table 6.2.3-1 lists the individual resources defined for the API and the applicable HTTP methods as defined in ETSI TS 129 222 [9], clause 8.1 of) and the associated service operations.

Table 6.2.3-1: Resources and methods overview of the service discovery API

Resource Name ETSI TS 129 222 [9]	Resource URI ETSI TS 129 222 [9]	HTTP method ETSI TS 129 222 [9]	Service Operation
All published service APIs	.../allServiceAPIs	GET	Query service APIs

6.2.4 Service operations

6.2.4.1 Query service APIs

6.2.4.1.1 Operation definition

The API Consumer uses the Query service APIs operation to discover service APIs information.

The operation is based on HTTP GET as per Figure 6.2.4.1.1-1. The HTTP GET response contains information about all services that the API Consumer is authorized to access and that match the filtering criteria.

Figure 6.2.4.1.1-1: Query service APIs operation

This service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource (…/allServiceAPIs) and may also contain that includes the rApp identifier and optional filtering criteria. The API Producer shall process the service discovery details received in the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success "200 OK" shall be returned and the message content shall carry a list of service profiles that the API Consumer is authorized to access and that match the filtering criteria, if provided. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

6.2.4.1.2 Referenced procedures

6.2.4.1.2.1 Discovery services procedure

The Query service APIs operation illustrated in Figure 6.2.4.1.1-1 is based on the Discover services procedure as defined in R1GAP [5].

6.2.5 Resources

6.2.5.1 Overview

This clause defines the resource for the Service discovery API based on ETSI TS 129 222 [9].

6.2.5.2 Resource: "All published service APIs"

6.2.5.2.0 Description

This resource is defined in ETSI TS 129 222 [9], clause 8.1 with the following URI:

{apiRoot}/service-apis /<apiVersion>/allserviceapis

In addition to the provisions in ETSI TS 129 222 [9], clause 8.1, the following shall apply:

1) In the GET method, the URI query parameters "api-name" and "api-version" are relevant in the context of the SME services and shall be supported by the API Producer to allow filtering the query by name and/or version of the service API. The API Consumer may provide these parameters. The following parameters are not applicable in the context of the SME services and therefore need not be supported: comm-type, protocol, aef-id, data-format, api-cat, preferred-aef-loc, supported-features, api-supported-features.

2) In the GET method, the URI query parameter "api-invoker-id" shall carry the rAppId of the rApp that performs the service discovery.

6.2.5.2.1 Resource Standard Methods

6.2.5.2.1.1 GET

This method shall support the URI query parameters, response data structures and response codes specified in ETSI TS 129 222 [9], clause 8.1.2.2.3.1.

6.2.5.2.2 Resource Custom Operations

None.

6.2.6 Custom operations without associated resources

None.

6.2.7 Notifications

None.

6.2.8 Data model

6.2.8.1 General

The application data model is defined in ETSI TS 129 222 [9], clause 8.1.4, apply to this API. In addition, the adaptations specified in clause B.3 apply to this API.

6.2.8.2 Structured data types

None.

6.2.8.3 Simple data types and enumerations

None.

6.2.8.4 Re-used data types

The re-used data types are defined in clause B.3.

6.2.8.5 Service-specific registration information

None.

6.2.9 Error Handling

6.2.9.1 General

HTTP error handling is applicable for this API as specified in ETSI TS 129 222 [9], clause 7.7.

6.2.9.2 Protocol Errors

Protocol error handling shall be supported as specified in ETSI TS 129 222 [9], clause 8.1.5.2.

6.2.9.3 Application Errors

Application error handling shall be supported as specified in ETSI TS 129 222 [9], clause 8.2.5.3.

6.3 Service events subscription API

6.3.1 Introduction

This API allows the API Consumer to subscribe to and unsubscribe from service event notifications as specified in R1GAP [5]. The API is based on the CAPIF_Events_API as specified in ETSI TS 129 222 [9].

6.3.2 API version

For the service events subscription API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 2 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1], clause 4.3.1.1, for a definition of the version fields). Consequently, the <apiVersion> URI part segment shall be set to "v1".

6.3.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined for the CAPIF_Events_API (see ETSI TS 129 222 [9], Figure 8.3.2.1-1 in clause 8.3.1).

Table 6.3.3-1 lists the individual resources defined for the API and the applicable HTTP methods as defined in ETSI TS 129 222 [9], clause 8.3.1, and the associated service operations.

Table 6.3.3-1: Resources and methods overview of the service subscription API

Resource Name ETSI TS 129 222 [9]	Resource URI ETSI TS 129 222 [9]	HTTP method ETSI TS 129 222 [9]	Service Operation
CAPIF Events Subscriptions	/{subscriberId}/subscriptions	POST	Subscribe service events
Individual CAPIF Events Subscription	/{subscriberId}/subscriptions/ {subscriptionId}	DELETE	Unsubscribe service events

6.3.4 Service operations

6.3.4.1 Subscribe service events

6.3.4.1.1 Operation definition

The API Consumer uses the subscribe service events API operation to subscribe to service event notifications.

The operation is based on HTTP POST.

Figure 6.3.4.1.1-1: Subscribe service events APIs operation

This service operation is as follows:

1) The API Consumer shall send an HTTP POST request to the API Producer. The target URI shall identify the resource (".../{subscriberId}/subscriptions/") under which the new subscription is requested to be created. The message content shall carry a EventSubscription structure. The API Producer shall process the request received in the HTTP POST message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return HTTP POST response. On success "201 Created" shall be returned. The "Location" HTTP header shall be present and shall carry the URI for the newly created resource i.e. subscription identifier. The message content shall carry a EventSubscription structure that represents the new resource. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

6.3.4.1.2 Referenced procedures

6.3.4.1.2.1 Subscribe service availability

The Subscribe service events operation illustrated in Figure 6.3.4.1.1-1 is based on the Subscribe service availability procedure defined in R1GAP [5].

6.3.4.2 Unsubscribe service events

6.3.4.2.1 Operation definition

The API Consumer uses the unsubscribe service events operation to unsubscribe from service event notifications.

The operation is based on HTTP DELETE.

Figure 6.3.4.2.1-1: Unsubscribe service events operation

This service operation is as follows:

1) The API Consumer shall send an HTTP DELETE request to the API Producer. The target URI shall identify the resource ("…/{subscriberId}/subscriptions/{subscriptionId}"). The message content shall be empty. The API Producer shall process the request received in the HTTP DELETE message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP DELETE response. On success "204 No Content" shall be returned. The message content shall be empty. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

6.3.4.2.2 Referenced procedures

6.3.4.2.2.1 Unsubscribe service availability

The Unsubscribe service events operation illustrated in Figure 6.3.4.3.1-1 is based on the Unsubscribe service availability procedure defined in R1GAP [5].

6.3.4.3 Notification service API

6.3.4.3.1 Operation definition

The API Producer sends notifications to all subscribed API Consumers.

The operation is based on HTTP POST.

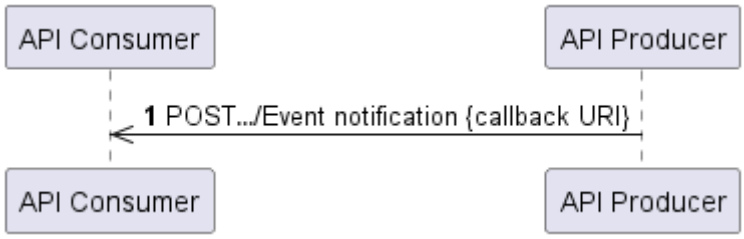

Figure 6.3.4.3.1-1: Notification service API operation

This service operation is as follows:

1) The API Producer shall send an HTTP POST request to the API Consumer. The target URI ("…/Eventnotification") shall contain the call back URI received in the subscription request to notify the generated events to all the subscribed API Consumers.

6.3.4.3.2 Referenced procedures

6.3.4.3.2.1 Notify service availability changes

The Notification service API operation illustrated in Figure 6.3.4.3.1-1 is based on the Notify service availability changes procedure defined in R1GAP [5].

6.3.5 Resources

6.3.5.1 Overview

This clause defines the resource for the Service events subscription API based on ETSI TS 129 222 [9].

6.3.5.2 Resource: "CAPIF Events Subscriptions"

6.3.5.2.0 Description

This resource is defined in ETSI TS 129 222 [9], clause 8.3, with the following URI:

{apiRoot}/capif-events/<apiVersion>/{subscriberId}/subscriptions

This resource shall support the resource URI variables defined in ETSI TS 129 222 [9], Table 8.3.2.2.2-1.

6.3.5.2.1 Resource Standard Methods

6.3.5.2.1.1 POST

This method shall support the URI query parameters, request data, response data structures and response codes specified in ETSI TS 129 222 [9], clause 8.3.2.2.3.1.

6.3.5.2.2 Resource Custom Operations

None.

6.3.5.3 Resource: "Individual CAPIF Events Subscription"

6.3.5.3.0 Description

This resource is defined in ETSI TS 129 222 [9], clause 8.3, with the following URI:

{apiRoot}/capif-events/<apiVersion>/{subscriberId}/subscriptions/{subscriptionId}

This resource shall support the resource URI variables defined in ETSI TS 129 222 [9], Table 8.3.2.3.2-1.

6.3.5.3.1 Resource Standard Methods

6.3.5.3.1.1 DELETE

This method shall support the URI query parameters, request data, response data structures and response codes specified in ETSI TS 129 222 [9], clause 8.3.2.3.3.1.

6.3.5.3.1.2 PUT

This method shall support the URI query parameters, request data, response data structures and response codes specified in ETSI TS 129 222 [9], clause 8.3.2.3.3.2.

6.3.5.3.1.2 PATCH

This method shall support the URI query parameters, request data, response data structures and response codes specified in ETSI TS 129 222 [9], clause 8.3.2.3.3.3.

6.3.5.3.2 Resource Custom Operations

None.

6.3.6 Custom operations without associated resources

None.

6.3.7 Notifications

6.3.7.1 General

The delivery of notification shall conform to ETSI TS 129 222 [9], clause 8.3.3.1.

6.3.7.2 Event Notification

6.3.7.2.1 Description

Event Notifications is used by the API Producer to notify all the subscribed API Consumers. The API Consumers shall subscribe to Event Notifications via the Individual CAPIF Events Subscription resource.

6.3.7.2.2 Notification definition

This method shall support the request data structures, URI query parameters and the response data structures as specified in ETSI TS 129 222 [9], clause 8.3.3.2.2.

6.3.8 Data Model

6.3.8.1 General

The application data model is defined in ETSI TS 129 222 [9], clause 8.3.4. In addition, the adaptations specified in clause B.3 of the present document and the data types listed in ETSI TS 129 222 [9], clause 7.2, also apply to this API.

6.3.8.2 Structured data types

None.

6.3.8.3 Simple data types and enumerations

None.

6.3.8.4 Re-used data types

The re-used data types are defined in clause B.3.

6.3.8.5 Service-specific registration information

None.

6.3.9 Error Handling

General error responses are defined in ETSI TS 129 222 [9], clause 7.7.

6.4 Bootstrap API

6.4.1 Introduction

This API allows the API Consumer to discover the entry points into the Service management and exposure services and the related token endpoints as defined in R1GAP [5].

6.4.2 API version

For the Bootstrap API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v1".

6.4.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined in clause 5.2. The <apiName> resource URI variable shall be "bootstrap". The <apiSpecificResourceUriPart> for each resource shall be set as described in clause 6.4.5.

Figure 6.4.3-1 shows the overall resource URI structure defined for the bootstrap API.

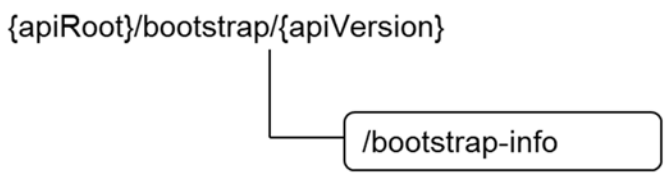

Figure 6.4.3-1: Resource URI structure of the Bootstrap API

Table 6.4.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 6.4.3-1: Resource and methods overview of the Bootstrap API

Resource name	Resource URI	HTTP method	Service Operation
Bootstrap info	…/bootstrap-info/	GET	Query bootstrap information

6.4.4 Service Operations

6.4.4.1 Query bootstrap information

6.4.4.1.1 Operation definition

The API Consumer uses this operation to retrieve bootstrap information.

The operation to query bootstrap information based on HTTP GET.

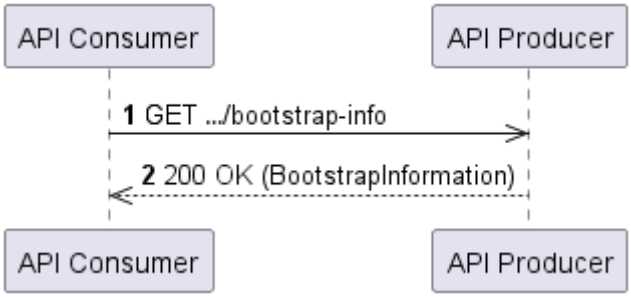

Figure 6.4.4.1.1-1: Query bootstrap information operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource (…/bootstrap-info). The message content shall be empty.

2) The API Producer shall return the HTTP response. On success, "200 OK" shall be returned. The message content shall carry the BootstrapInformation. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

6.4.4.1.2 Referenced procedures

6.4.4.1.2.1 Discover bootstrap procedure

The Query bootstrap operation illustrated in Figure 6.4.4.1.1-1 is based on the Discover bootstrap procedure defined for the Service management and exposure services in R1GAP [5].

6.4.5 Resources

6.4.5.1 Overview

The following clause defines the resources for the Bootstrap API.

6.4.5.2 Resource: "Bootstrap info"

6.4.5.2.1 Description

The resource represents the bootstrap information for service management and exposure services.

Only the methods defined in clause 6.4.5.2.3 shall be supported by this resource.

6.4.5.2.2 Resource Definition

Resource URI: **{apiRoot}/bootstrap/<apiVersion>/bootstrap-info**

The resource URI variables supported by the resource is defined in Table 6.4.5.2.2-1.

Table 6.4.5.2.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2
apiVersion	See clause 6.4.2

6.4.5.2.3 Resource Standard Methods

6.4.5.2.3.1 GET

This method shall support the URI query parameters specified in Table 6.4.5.2.3.1-1, the request data structure specified in Table 6.4.5.2.3.1-2 and the response data structure and response code specified in Table 6.4.5.2.3.1-3.

Table 6.4.5.2.3.1-1: URI query parameters supported by the GET method on this resource

Name	Data type	P	Cardinality	Description	Applicability
N/A					

Table 6.4.5.2.3.1-2: Data structures supported by the GET request body on this resource

Data Type	P	Cardinality	Description
N/A			A GET request has no message content

Table 6.4.5.2.3.1-3: Data structures supported by the GET response body on this resource

Data Type	P	Cardinality	Response codes	Description
BootstrapInformation	M	1	200 OK	The bootstrap information has been queried successfully by the rApp and the response contains a BootstrapInformation structure as a representation of the query resource.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

6.4.5.2.4 Resource Custom Operations

None.

6.4.6 Custom operation without associated resources

None.

6.4.7 Notifications

None.

6.4.8 Data Model

6.4.8.1 Structured data types

6.4.8.1.1 Overview

The following clauses define the structured data types and their attributes to be used by the bootstrap API.

6.4.8.1.2 Data type: BootstrapInformation

The BootstrapInformation data type represents entry point information for the Service management and exposure services. It contains the attributes defined in Table 6.4.8.1.2-1.

Table 6.4.8.1.2-1: Definition of type BootstrapInformation

Attribute Name	Data type	P	Cardinality	Description
apiEndpoints	array(ApiEndpointInformation)	M	1..N	Information about the API.

6.4.8.1.3 Data type: ApiEndpointInformation

The ApiEndpointInformation data type represents entry point information for a single API. It contains the attributes defined in Table 6.4.8.1.3-1.

Table 6.4.8.1.3-1: Definition of type ApiEndpointInformation

Attribute Name	Data type	P	Cardinality	Description
apiName	string	M	1	Name of API, as defined in the URI structure as <apiName>. The following values shall be supported: "service-apis" for discovery and "published-apis" for service registration
tokenEndPoint	InterfaceDescription	C	0..1	InterfaceDescription as defined in clause B.3.5, Token endpoint shall be provided if the API requires authorization over OAuth2.0
apiEndPoint	InterfaceDescription	M	1	InterfaceDescription as defined in clause B.3.5, End point of the API

6.4.8.2 Simple data types and enumerations

None.

6.4.8.3 Re-used data types

None.

6.4.8.4 Service-specific registration information

None.

6.4.9 Error Handling

6.4.9.1 General

In addition to the general provisions in clause 5.4.3, the requirements in the following clauses are applicable for Bootstrap API.

6.4.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

6.4.9.3 Application Errors

No additional application errors defined in the present document.

7 Data management and exposure services

7.1 Data registration API

7.1.1 Introduction

This API enables the API Consumer to register DME type production capabilities based on the data registration service procedure defined in R1GAP [5].

7.1.2 API version

For the data registration API as specified in the present document, the MAJOR version field shall be 2, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v2".

The API is under development and consequently the API version shall include the pre-release version "alpha.2".

7.1.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined in clause 5.2. The <apiName> resource URI variable shall be "dataregistration". The <apiSpecificResourceUriPart> for each resource shall be set as described in clause 7.1.5.

Figure 7.1.3-1 shows the overall resource URI structure defined for the data registration API.

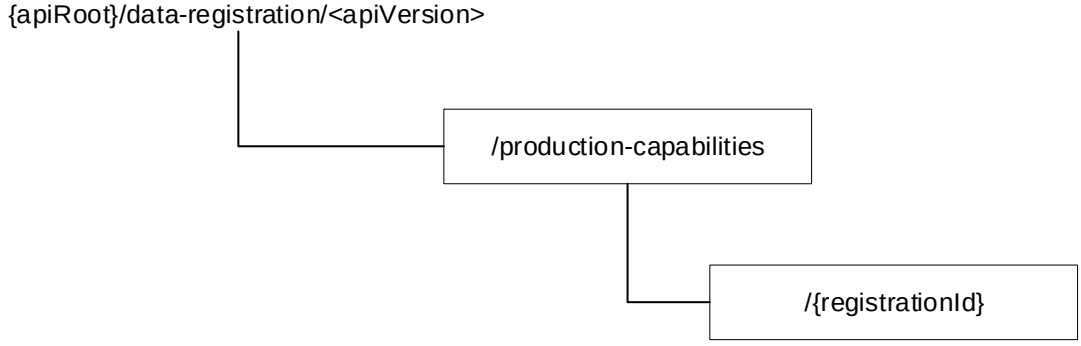

Figure 7.1.3-1: Resource URI structure of the Data registration API

Table 7.1.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 7.1.3-1: Resource and methods overview of the Data registration API

Resource name	Resource URI	HTTP method	Service Operation
Registered DME type production capabilities	…/production- capabilities	POST	Register DME type
Individual registered DME type production capability	…/production-capabilities/ {registrationId}	DELETE	Deregister DME type
		PUT	Update DME type
		GET	Query DME type

7.1.4 Service Operations

7.1.4.1 Register DME type

7.1.4.1.1 Operation definition

The API Consumer uses this operation to register DME type production capabilities.

The operation to register the capability to produce a DME type is based on HTTP POST.

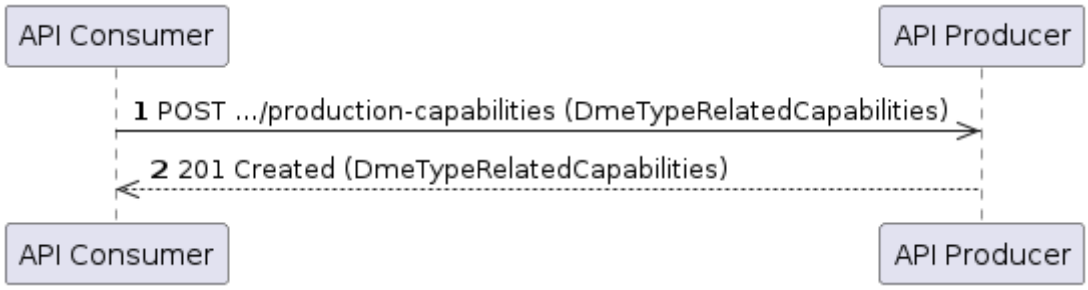

Figure 7.1.4.1.1-1: Register DME type operation

The service operation is as follows:

1) The API Consumer shall send an HTTP POST request to the API Producer. The target URI shall identify the resource (…/production-capabilities) under which the new registration is requested to be created. The message content shall carry a DmeTypeRelatedCapabilities structure.

2) The API Producer shall generate the registration identifier and construct the URI for the created resource. The API Producer shall return the HTTP POST response. On success, "201 Created" shall be returned. The "Location" header shall be present and shall carry the URI of the new registration resource. The message content shall carry a DmeTypeRelatedCapabilities structure that represents the new resource. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

7.1.4.1.2 Referenced procedures

7.1.4.1.2.1 Register DME type procedure

The Register DME type operation illustrated in Figure 7.1.4.1.1-1 is based on the Register DME type procedure defined for the Data registration service in R1GAP [5].

7.1.4.2 Deregister DME type

7.1.4.2.1 Operation definition

The API Consumer uses this operation to deregister DME type production capabilities.

The operation to deregister the capability to produce a DME type is based on HTTP DELETE.

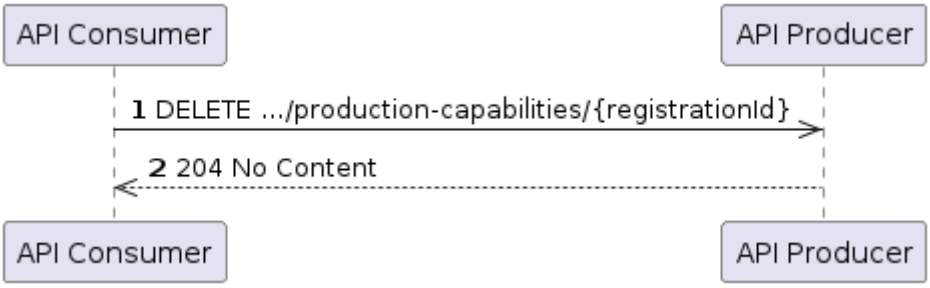

Figure 7.1.4.2.1-1: Deregister DME type operation

The service operation is as follows:

1) The API Consumer shall send an HTTP DELETE request to the API Producer. The target URI shall identify the resource to be deleted (…/production-capabilities/{registrationId}).

2) The API Producer shall return the HTTP DELETE response. On success, "204 No Content" shall be returned and the response message content shall be empty. On failure, the appropriate error code shall be returned, and the message response content may contain additional error information.

7.1.4.2.2 Referenced procedures

7.1.4.2.2.1 Deregister DME type procedure

The Deregister DME type operation illustrated in Figure 7.1.4.2.1-1 is based on the Deregister DME type procedure defined for the Data registration service in R1GAP [5].

7.1.4.3 Update DME type

7.1.4.3.1 Operation definition

The API Consumer uses this operation to update the registration of production capabilities related to a DME type.

The operation to update the registration of production capability related to a DME type is based on HTTP PUT.

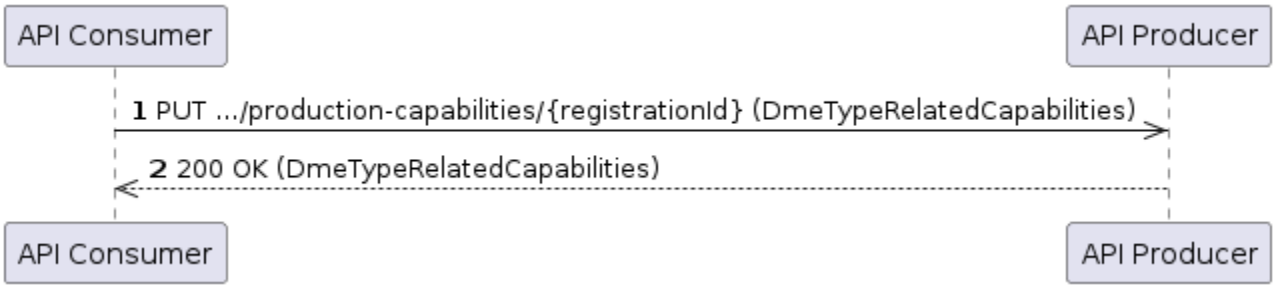

Figure 7.1.4.3.1-1: Update DME type operation

The service operation is as follows:

1) The API Consumer shall send an HTTP PUT request to the API Producer. The target URI shall identify the resource (…/production-capabilities/{registrationId}). The message content shall carry an updated DmeTypeRelatedCapabilities structure. The API producer shall process the HTTP PUT message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP response. On success, "200 OK" shall be returned. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

7.1.4.3.2 Referenced procedures

7.1.4.3.2.1 Update DME type procedure

The Update DME type operation illustrated in Figure 7.1.4.4.1-1 is based on the Update DME type procedure defined for the Data registration service in R1GAP [5].

7.1.4.4 Query DME type

7.1.4.4.1 Operation definition

The API Consumer uses this operation to query the registration of production capability information on a specific DME type that it has previously registered.

The operation to query the registration of production capability is based on HTTP GET.

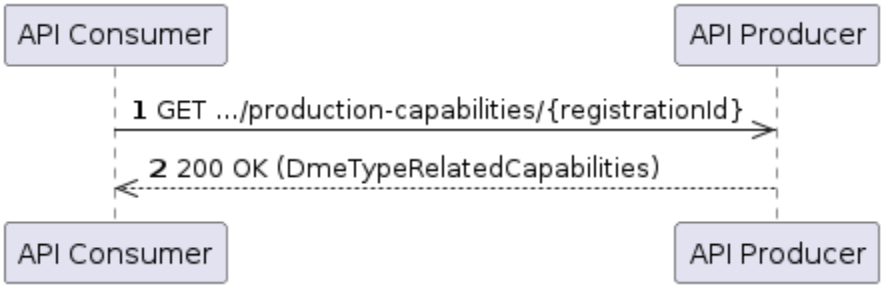

Figure 7.1.4.4.1-1: Query DME type operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource (…/production-capabilities/{registrationId}). The message content shall be empty. The API producer shall process the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP response. On success, "200 OK" shall be returned. The message content shall carry the queried production capability information. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

7.1.4.4.2 Referenced procedures

7.1.4.4.2.1 Query DME type procedure

The Query DME type operation illustrated in Figure 7.1.4.4.1-1 is based on the Query DME type procedure defined for the Data registration service in R1GAP [5].

7.1.5 Resources

7.1.5.1 Overview

The following clause defines the resources for the Data registration API.

7.1.5.2 Resource: "Registered DME type production capabilities"

7.1.5.2.1 Description

The resource represents the registered capabilities of an API Consumer to produce DME types.

Only the methods defined in clause 7.1.5.2.3 shall be supported by this resource.

7.1.5.2.2 Resource Definition

Resource URI: **{apiRoot}/data-registration/<apiVersion>/production-capabilities**

The resource URI variables supported by the resource are defined in Table 7.1.5.2.2-1.

Table 7.1.5.2.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2
apiVersion	See clause 7.1.2

7.1.5.2.3 Resource Standard Methods

7.1.5.2.3.1 POST

This method shall support the request data structure specified in Table 7.1.5.2.3.1-1 and the response data structure and response code specified in Table 7.1.5.2.3.1-2, and the HTTP headers specified in Table 7.1.5.2.3.1-3.

Table 7.1.5.2.3.1-1: Data structures supported by the POST request body on this resource

Data Type	P	Cardinality	Description
DmeTypeRelatedCapabilities	M	1	Registered capabilities of a Data Producer related to a DME type

Table 7.1.5.2.3.1-2: Data structures supported by the POST response body on this resource

Data Type	P	Cardinality	Response codes	Description
DmeTypeRelatedCapabilities	M	1	201 Created	The operation was successful, and the message content of the POST response contains a DmeTypeProdCapRegistration structure as a representation of the created resource.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

Table 7.1.5.2.3.1-3: Headers supported by the 201 Response Code on this resource

Name	Data type	P	Cardinality	Description
Location	string	M	1	Contains the URI of the newly created "Individual registered DME type production capability" resource, as defined in clause 7.1.5.3, with the registrationId in the URI.

7.1.5.2.4 Resource Custom Operations

None.

7.1.5.3 Resource: "Individual registered DME type production capability"

7.1.5.3.1 Description

The resource represents an individual registered DME type production capability.

Only the methods defined in clause 7.1.5.3.3 shall be supported by this resource.

7.1.5.3.2 Resource Definition

Resource URI: **{apiRoot}/data-registrations/<apiVersion>/production-capabilities/{registrationId}**

The resource URI variables supported by the resource are defined in Table 7.1.5.3.2-1.

Table 7.1.5.3.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2
apiVersion	See clause 7.1.2
registrationId	The related DME type production capabilities registration identifier

7.1.5.3.3 Resource Standard Methods

7.1.5.3.3.1 DELETE

This method shall support the request data structure specified in Table 7.1.5.3.3.1-1 and the response data structure and response code specified in Table 7.1.5.3.3.1-2.

Table 7.1.5.3.3.1-1: Data structures supported by the DELETE request body on this resource

Data type	P	Cardinality	Description
N/A			A DELETE request has no message content

Table 7.1.5.3.3.1-2: Data structures supported by the DELETE response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No Content	The DME type production capability registration associated with the registrationId has been deleted successfully. The message content shall be empty.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

7.1.5.3.3.2 PUT

This method shall support the request data structure specified in Table 7.1.5.3.3.2-1 and the response data structure and response code specified in Table 7.1.5.3.3.2-2.

Table 7.1.5.3.3.2-1: Data structures supported by the PUT request body on this resource

Data Type	P	Cardinality	Description
DmeTypeRelatedCapabilities	M	1	Updated DME type production capability information

Table 7.1.5.3.3.2-2: Data structures supported by the PUT response body on this resource

Data Type	P	Cardinality	Response codes	Description
DmeTypeRelatedCapabilities	M	1	200 OK	The DME type production capability registration associated with the registrationId has been updated successfully and the response contain the DmeTypeProdCapRegistration as a representation of the updated resource.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

7.1.5.3.3.3 GET

This method shall support the request data structure specified in Table 7.1.5.3.3.3-1 and the response data structure and response code specified in Table 7.1.5.3.3.3-2.

Table 7.1.5.3.3.3-1: Data structures supported by the GET request body on this resource

Data Type	P	Cardinality	Description
N/A			A GET request has no message content

Table 7.1.5.3.3.3-2: Data structures supported by the GET response body on this resource

Data Type	P	Cardinality	Response codes	Description
DmeTypeRelatedCapabilities	M	1	200 OK	The DME type production capability registration associated with the registrationId has been queried successfully and the response contain the DmeTypeProdCapRegistration as a representation of the query resource.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

7.1.5.3.4 Resource Custom Operations

None.

7.1.6 Custom operation without associated resources

None.

7.1.7 Notifications

None.

7.1.8 Data Model

7.1.8.1 Structured data types

7.1.8.1.1 Overview

The following clauses define the structured data types and their attributes to be used by the service API.

7.1.8.1.2 Data type: DmeTypeRelatedCapabilities

The DmeTypeRelatedCapabilities data type represents capabilities of a data provider entity (such as Data Producer, SME functions) related to providing data instances of a DME type for collection or consumption. It contains the attributes defined in Table 7.1.8.1.2-1.

Table 7.1.8.1.2-1: Definition of type DmeTypeRelatedCapabilities

Attribute Name	Data type	P	Cardinality	Description
dmeTypeDefinition	DmeTypeDefinition	M	1	Information of the DME type.
dataAccessEndpoint	Interface Description	O	0..1	Endpoint to which to send the data request or data subscription for data instances of this DME type. This attribute shall be provided in data registration requests and responses related to data registration. It may be provided in data discovery responses. If this information is not provided, the data access endpoint can be discovered by means outside the scope of the of the DME services.
dataDeliveryMode	array(DataDeliveryMode)	M	1..N	Supported modes for data delivery for this DME type, i.e. one time (data request) or continuous (data subscription) or both.
constraints	object	O	0..1	When this data structure is used for registration, this attribute represents producer constraints related to the DME type based on the dataProductionSchema. When this data structure is used for discovery, this attribute represents constraints applicable to the consumption of data instances of this DME type based on the dataProductionSchema.

7.1.8.1.3 Data type: DmeTypeDefinition

The DmeTypeDefinition data type represents information about a DME type. It contains the attributes defined in Table 7.1.8.1.3-1.

Table 7.1.8.1.3-1: Definition of type DmeTypeDefinition

Attribute Name	Data type	P	Cardinality	Description
dmeTypeId	DmeTypeIdStruct	M	1	The identifier of the DME type being registered.
metadata	Metadata	M	1	Metadata that can be used in discovering the DME type.
dataProductionSchema	object	O	0..1	Schema that defines the information necessary to formulate a data request or data subscription. If this attribute is not present, the schema is assumed to be known from the DME type definition that is referenced by dmeTypeId (see note).
dataDeliverySchemas	array (DeliverySchema)	M	1.. N	List of delivery schemas supported by the producer for the DME type being registered (see note).
dataDeliveryMechanisms	array (DataDeliveryMechanism)	M	1.. N	See clause B.4.4.2.
NOTE: The schemas for data production and data delivery are DME type specific and are not defined in the present document.				

7.1.8.1.4 Data type: Metadata

The Metadata data type contains the attributes defined in Table 7.1.8.1.4-1 and the set of metadata attributes may be extended by deployments.

Table 7.1.8.1.4-1: Definition of type Metadata

Attribute Name	Data type	P	Cardinality	Description
dataCategory	array(string)	M	1.. N	Defines the category of the DME type e.g. PM counters.
rat	array(string)	O	0.. N	Defines the radio access technology e.g. 5G.

7.1.8.1.5 Data type: DeliverySchema

The DeliverySchema data type contains the attributes defined in Table 7.1.8.1.5-1.

Table 7.1.8.1.5-1: Definition of type DeliverySchema

Attribute Name	Data type	P	Cardinality	Description
type	SchemaTypes	M	1	Type of the schema.
deliverySchemaId	string	M	1	A Data Producer may support one or more delivery schemas and for each supported schema type a delivery schema identifier is assigned. A Data Consumer uses this attribute while creating a data job and request to deliver the data using specific schema type which is identified by this attribute.
schema	string	O	0..1	The schema serialized to string. If this attribute is not present, the schema is assumed to be known from the DME type definition that is referenced by the DME type identifier.

7.1.8.2 Simple data types and enumerations

7.1.8.2.1 Introduction

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

7.1.8.2.2 Simple data types

No simple data types are defined in the present document.

7.1.8.2.3 Enumeration

7.1.8.2.3.1 Enumeration: SchemaTypes

Table 7.1.8.2.3.1-1: Enumeration SchemaTypes

Enumeration value	Description
JSON_SCHEMA	Following JSON Schema 2020-12 [14].
XML_SCHEMA	Following XML Schema [16], [17], [18] and [i.1].

7.1.8.3 Re-used data types

None.

7.1.8.4 Service-specific registration information

None.

7.1.9 Error Handling

7.1.9.1 General

In addition to the general provisions in clause 5.4.3, the requirements in the following clauses are applicable for the Data Registration API.

7.1.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

7.1.9.3 Application Errors

No additional application errors defined in the present document.

7.2 Data discovery API

7.2.1 Introduction

This API enables the API Consumer to discover the available DME types based on the data discovery service procedures defined in R1GAP [5].

7.2.2 API version

For the data discovery API as specified in the present document, the MAJOR version field shall be 2, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v2".

7.2.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined in clause 5.2. The <apiName> resource URI variable shall be "datadiscovery". The <apiSpecificResourceUriPart> for each resource shall be set as described in clause 7.2.5.

Figure 7.2.3-1 shows the overall resource URI structure defined for the data discovery API.

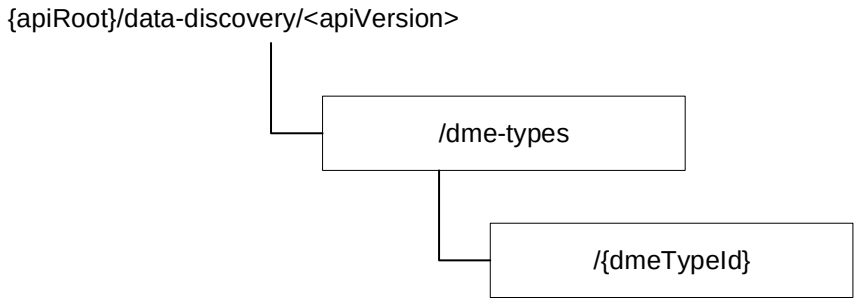

Figure 7.2.3-1: Resource URI structure of the Data discovery API

Table 7.2.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 7.2.3-1: Resources and methods overview of the Data discovery API

Resource name	Resource URI	HTTP method	Service Operation
All DME types	…/dme-types	GET	Discover DME types
Individual DME types	…/dme-types/{dmeTypeId}	GET	Query capabilities related to a DME type

7.2.4 Service operations

7.2.4.1 Discover DME types

7.2.4.1.1 Operation definition

The API Consumer uses this operation to discover the available DME types.

The operation to discover the DME types is based on HTTP GET.

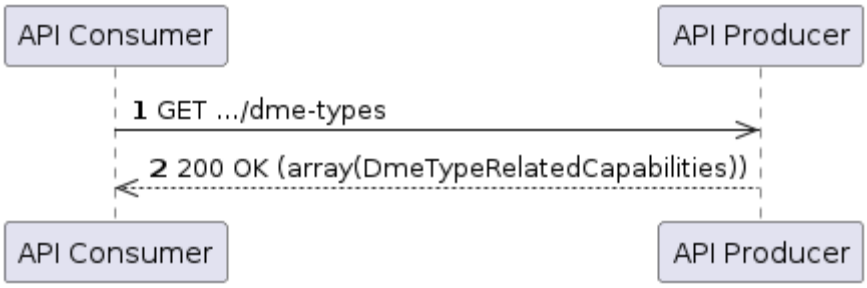

Figure 7.2.4.1.1-1: Discover DME types operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource (…/dme-types) and may also contain query parameters to discover the available DME types.

2) The API Producer shall return the HTTP GET response. On success, "200 OK" shall be returned and the message content shall carry an array of Dme type related capabilities. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

7.2.4.1.2 Referenced procedures

7.2.4.1.2.1 Discover DME types procedure

The Discover DME types operation illustrated in Figure 7.2.4.1.1-1 is based on the Discover DME types procedure defined for the Data discovery service in R1GAP [5].

7.2.4.2 Query DME type information

7.2.4.2.1 Operation definition

The API Consumer uses this operation to query for information about a specific DME type identified by a DME type identifier.

The operation to query for information about a specific DME type is based on HTTP GET.

Figure 7.2.4.2.1-1: Query DME type information operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource (…/dme-types/{dmeTypeId}) and the message content shall be empty.

2) The API Producer shall return the HTTP GET response. On success, "200 OK" shall be returned and the message content shall carry capability information related to the DME type identified by {dmeTypeId}. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

7.2.4.2.2 Referenced procedures

7.2.4.2.2.1 Query DME type information procedure

The Query DME type operation illustrated in Figure 7.2.4.2.1-1 is based on the Query DME type information procedure defined for the Data discovery service in R1GAP [5].

7.2.5 Resources

7.2.5.1 Overview

The following clause defines the resources for the Data discovery API.

7.2.5.2 Resource: "All DME types"

7.2.5.2.1 Description

The resource represents the available DME types. Only the methods defined in clause 7.2.5.2.3 shall be supported by this resource.

7.2.5.2.2 Resource Definition

Resource URI: **{apiRoot}/data-discovery/<apiVersion>/dme-types**

The resource URI variables supported by the resource are defined in Table 7.2.5.2.2-1.

Table 7.2.5.2.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 7.2.2.

7.2.5.2.3 Resource Standard Methods

7.2.5.2.3.1 GET

This method shall support the URI query parameters specified in Table 7.2.5.2.3.1-1, the request data structure specified in Table 7.2.5.2.3.1-2 and the response data structure and response code specified in Table 7.2.5.2.3.1-3.

Table 7.2.5.2.3.1-1: URI query parameters supported by the GET method on this resource

Name	Data type	P	Cardinality	Description	Applicability
identity-namespace	string	O	0..1	Identity namespace, which shall match the "namespace" part of the "dmeTypeId" attribute (see note 1).	
identity-name	string	O	0..1	Identity name, which shall match the "name" part of the "dmeTypeId" attribute (see note 1).	
data-category	array(string)	O	0..N	Set of data category entries, all of which shall match entries of the "dataCategory" attribute (see notes 1 and 2).	
NOTE 1: If multiple query parameters are provided these shall be combined with AND when evaluating the query. NOTE 2: The encoding of query parameter for array of string shall follow the guideline defined in ETSI TS 129 501 [1] clause 5.3.13.					

Table 7.2.5.2.3.1-2: Data structures supported by the GET request body on this resource

Data Type	P	Cardinality	Description
N/A			There is no object in the message content of the GET request.

Table 7.2.5.2.3.1-3: Data structures supported by the GET response body on this resource

Data Type	P	Cardinality	Response codes	Description
array (DmeTypeRelatedCapabilities)	M	0.. N	200 OK	The operation was successful. The message content of the GET response carries an array of DmeTypeRelatedCapabilities structures.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

7.2.5.2.4 Resource Custom Operations

None.

7.2.5.3 Resource: "Individual DME type"

7.2.5.3.1 Description

The resource represents a DME type.

7.2.5.3.2 Resource Definition

Resource URI: **{apiRoot}/data-discovery/<apiVersion>/dme-types/{dmeTypeId}**

The resource URI variables supported by the resource are defined in Table 7.2.5.3.2-1.

Table 7.2.5.3.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2
apiVersion	See clause 7.2.2
dmeTypeId	DME type identifier identifying a DME type

7.2.5.3.3 Resource Standard Methods

7.2.5.3.3.1 GET

This method shall support the URI query parameters specified in Table 7.2.5.3.3.1-1, the request data structure specified in Table 7.2.5.3.3.1-2 and the response data structure and response code specified in Table 7.2.5.3.3.1-3.

Table 7.2.5.3.3.1-1: URI query parameters supported by the GET method on this resource

Name	Data type	P	Cardinality	Description	Applicability
N/A					

Table 7.2.5.3.3.1-2: Data structures supported by the GET request body on this resource

Data type	P	Cardinality	Description
N/A			The message content of a GET request is empty.

Table 7.2.5.3.3.1-3: Data structures supported by the GET response body on this resource

Data type	P	Cardinality	Response codes	Description
DmeTypeRelatedCapabilities	M	1	200 OK	The operation was successful. The message content a DmeTypeRelatedCapabilities structure.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

7.2.5.3.4 Resource Custom Operations

None.

7.2.6 Custom operation without associated resources

None.

7.2.7 Notifications

None.

7.2.8 Data Model

7.2.8.1 Structured data types

7.2.8.1.1 Overview

The following clause defines the structured data types and their attributes to be used by the service API.

For this service API, no structured data types are defined in the present document.

7.2.8.2 Simple data types and enumerations

7.2.8.2.1 Introduction

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

7.2.8.2.2 Simple data types

For this service API, no simple data types are defined in the present document.

7.2.8.2.3 Enumerations

For this service API, no enumerations are defined in the present document.

7.2.8.3 Re-used data types

None.

7.2.8.4 Service-specific registration information

None.

7.2.9 Error Handling

7.2.9.1 General

In addition to the general provisions in clause 5.4.3, the requirements in the following clauses are applicable for the Data Discovery API.

7.2.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

7.2.9.3 Application Errors

No additional application errors defined in the present document.

7.3 Data access API

7.3.1 Introduction

This API enables the API Consumer to request or subscribe data instances based on the Data request service and Data subscription service procedures defined in R1GAP [5]. The API definition applies to both scenarios when rApp is the Service Consumer and when DME is the Service Consumer, respectively.

7.3.2 API version

For the Data access API as specified in the present document, the MAJOR version field shall be 2, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v2".

The API is under development and consequently the API version shall include the pre-release version "alpha.2".

7.3.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined in clause 5.2. The <apiName> resource URI variable shall be "data-access". The <apiSpecificResourceUriPart> for each resource shall be set as described in clause 7.3.5.

Figure 7.3.3-1 shows the overall resource URI structure defined for the Data access API.

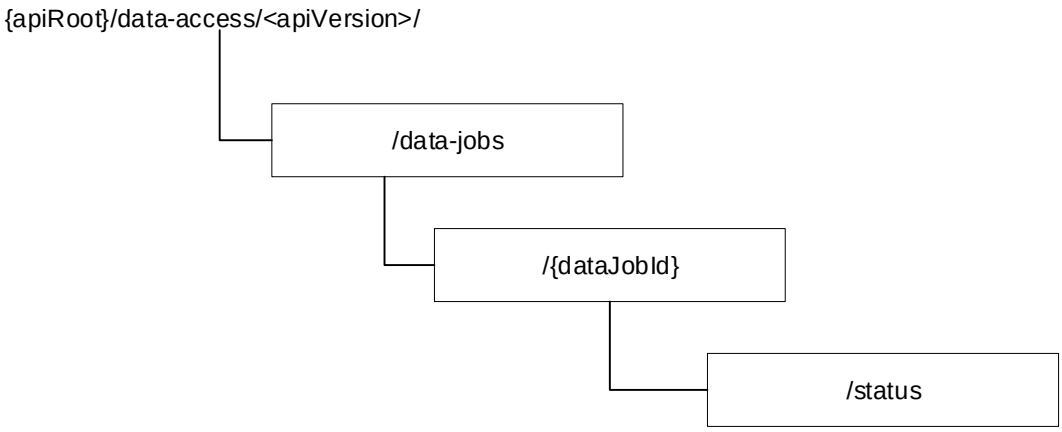

Figure 7.3.3-1: Resource URI structure of the Data access API

Table 7.3.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 7.3.3-1: Resources and methods overview of the data access API

Resource name	Resource URI	HTTP method	Service Operation
All data jobs	…/data-jobs	POST	Create an individual data job
		GET	Query data job identifiers
Individual data job	…/data-jobs/{dataJobId}	DELETE	Cancel data job
		PUT	Update data job
		GET	Query data job
Individual data job status	…/data-jobs/{dataJobId}/status	GET	Query data Job status

7.3.4 Service Operations

7.3.4.1 Create data job

7.3.4.1.1 Operation definition

The API Consumer uses Create data job operation to create the job for data request or subscription. A data job can be either for data request or data subscription.

An API Producer supporting the Data access API shall support at least one type of the services, data request (one time delivery mode) or data subscription (continuous delivery mode).

NOTE: The mean for communicating the service support capabilities with API Consumers is not specified in the present document.

The operations are based on HTTP POST.

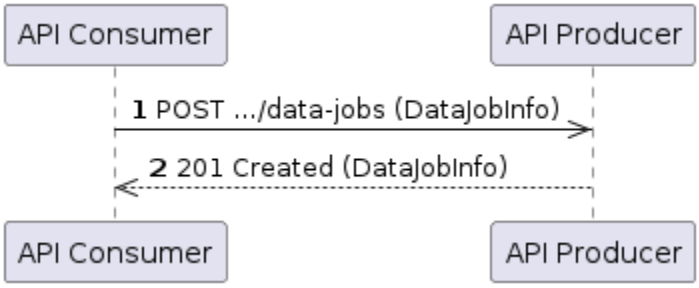

Figure 7.3.4.1.1-1: Create data job operation

The service operation is as follows:

1) The API Consumer shall send an HTTP POST request to the API Producer. The target URI shall identify the resource (…/data-jobs) under which the new data job is to be created. The message content shall carry a DataJobInfo.

2) The API Producer shall return the HTTP POST response. On success, "201 Created" shall be returned. The Location header shall be present and shall carry the URI of the new data job resource with dataJobId assigned by the service producer. The message content shall carry a DataJobInfo representing the created data job. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

7.3.4.1.2 Referenced procedures

7.3.4.1.2.1 Request data procedure

The Create data job operation illustrated in Figure 7.3.4.1.1-1 is based on the Request data procedure defined for the Data request service in R1GAP [5].

7.3.4.1.2.2 Subscribe data procedure

The Create data job operation illustrated in Figure 7.3.4.1.1-1 is based on the Subscribe data procedure defined for the Data subscription service in R1GAP [5].

7.3.4.2 Cancel data job

7.3.4.2.1 Operation definition

The API Consumer uses Cancel data job operation to cancel a data job.

The operations are based on HTTP DELETE.

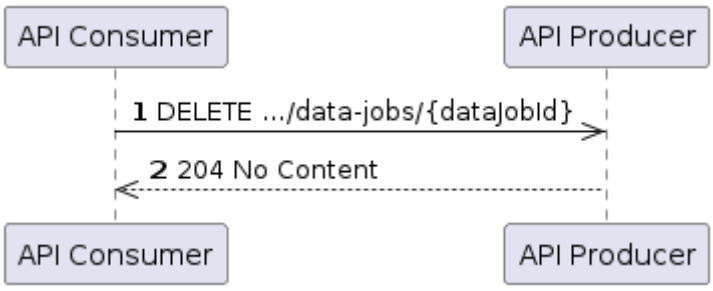

Figure 7.3.4.2.1-1: Cancel data job operation

The service operation is as follows:

1) The API Consumer shall send an HTTP DELETE request to API Producer. The target URI shall identify the resource (…/data-jobs/{dataJobId}).

2) The API Producer shall return the HTTP DELETE response. On success, "204 No Content" shall be returned. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

7.3.4.2.2 Referenced procedures

7.3.4.2.2.1 Cancel data request procedure

The Cancel data job operation illustrated in Figure 7.3.4.2.1-1 is based on the cancel data request procedure defined for the Data request service in R1GAP [5].

7.3.4.2.2.2 Unsubscribe data procedure

The Cancel data job operation illustrated in Figure 7.3.4.2.1-1 is based on the Unsubscribe data procedure defined for the Data subscription service in R1GAP [5].

7.3.4.3 Notify data availability

7.3.4.3.1 Operation definition

The API Producer uses this operation to notify data availability related to a data subscription.

The operation is based on HTTP POST.

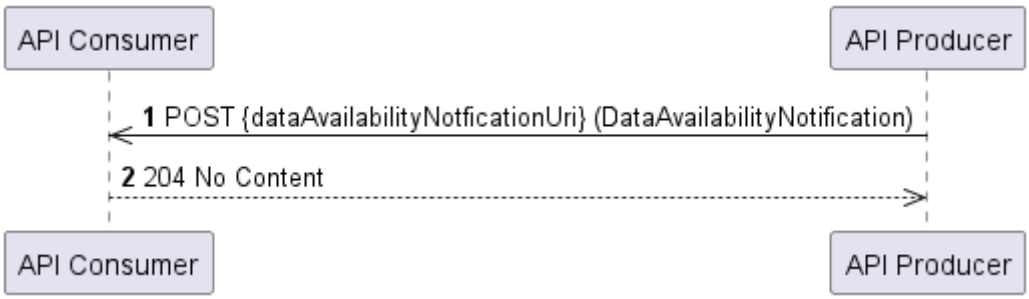

Figure 7.3.4.3.1-1: Notify data availability operation

The service operation is as follows:

1) The API Producer shall send an HTTP POST request to API Consumer. The target URI (dataAvailabilityNotificationUri) identifies the address where to send the notifications. The message content shall carry a DataAvailabilityNotification.

2) The API Consumer shall return the HTTP POST response. On success, "204 No Content" shall be returned. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

7.3.4.3.2 Referenced procedures

7.3.4.3.2.1 Notify data availability procedure

The Notify data job availability operation illustrated in Figure 7.3.4.3.1-1 is based on the Notify data availability procedure defined for the Data subscription service in R1GAP [5].

7.3.4.4 Update data job

7.3.4.4.1 Operation definition

The API Consumer uses this operation to update the created job for data subscription.

The operation to update a data job is based on HTTP PUT.

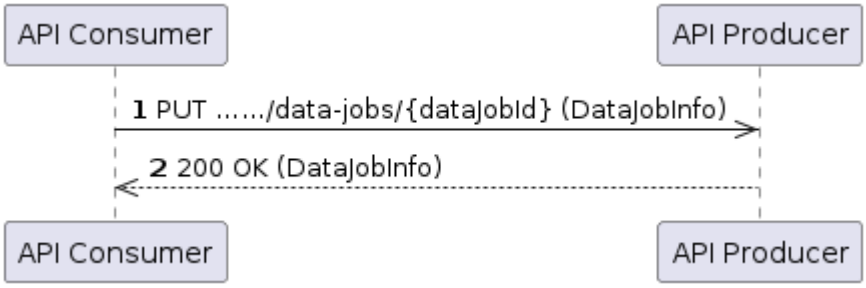

Figure 7.3.4.4.1-1: Update data job operation

The service operation is as follows:

1) The API Consumer shall send an HTTP PUT request to the API Producer. The target URI shall identify the resource (…/data-jobs/{dataJobId}). The message content shall carry an update DataJobInfo structure. The API Producer shall process the HTTP PUT message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP response. On success, "200 OK" shall be returned. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

3) The API Producer shall reject the update data job with "405 Method Not Allowed" if the dataJobId in the target URI is associated with delivery mode "ONE_TIME".

NOTE: Updating the Data Job can introduce discontinuities in the produced data that are specific to the DME type and/or Data Producer.

7.3.4.4.2 Referenced procedures

7.3.4.3.2.1 Update data job procedure

The Update data job operation illustrated in Figure 7.3.4.4.1-1 is based on the Update data subscription procedure defined for the Data subscription service in R1GAP [5].

7.3.4.5 Query data job

7.3.4.5.1 Operation definition

The API Consumer uses this operation to query the created job.

The operation to query the data job is based on HTTP GET.

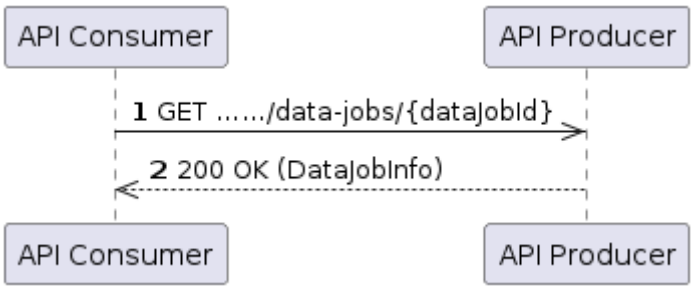

Figure 7.3.4.5.1-1: Query data job operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource (……/data-jobs/{dataJobId}). The message content shall be empty. The API producer shall process the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP response. On success, "200 OK" shall be returned. The message content shall carry the queried data job information. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

7.3.4.5.2 Referenced procedures

7.3.4.5.2.1 Query data job procedure

The Query data job operation illustrated in Figure 7.3.4.5.1-1 is based on the Query data subscription procedure defined for the Data subscription service in R1GAP [5].

7.3.4.6 Query data job status

7.3.4.6.1 Operation definition

The API Consumer uses this operation to query the data job status.

The operation to query the data job is based on HTTP GET.

Figure 7.3.4.6.1-1: Query data job status operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource (……/data-jobs/{dataJobId}/status). The message content shall be empty. The API producer shall process the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP response. On success, "200 OK" shall be returned. The message content shall carry the queried data job status information. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

7.3.4.6.2 Referenced procedures

7.3.4.6.2.1 Query data job procedure

The Query data job status operation illustrated in Figure 7.3.4.6.1-1 is based on the Query data subscription status procedure defined for the Data subscription service in R1GAP [5].

7.3.4.7 Query data job identifiers

7.3.4.7.1 Operation definition

The API Consumer uses Query data job identifiers operation to get the data job identifiers for all the jobs that are created for the specific API consumer. A data job can be either for data request or data subscription.

The operations are based on HTTP GET.

Figure 7.3.4.7.1-1: Query data job operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource (……/data-jobs/). The message content shall be empty. The API producer shall process the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP response. On success, "200 OK" shall be returned. The message content shall carry an array of data job identifiers (dataJobId) created by the API Consumer. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

7.3.4.7.2 Referenced procedures

7.3.4.7.2.1 Query data job identifiers procedure

The Query data job identifiers operation illustrated in Figure 7.3.4.7.1-1 is based on the Query data subscription procedure defined for the Data subscription service in R1GAP [5].

7.3.5 Resources

7.3.5.1 Overview

This clause defines the resources for the Data access API.

7.3.5.2 Resource: "All data jobs"

7.3.5.2.1 Description

The resource All data jobs represents all data jobs created by a particular consumer.

Only the methods defined in clause 7.3.5.2.3 shall be supported by these resources.

7.3.5.2.2 Resource Definition

Resource URI: **{apiRoot}/data-access/<apiVersion>/data-jobs**

The resource URI variables supported by the resource are defined in Table 7.3.5.2.2-1.

Table 7.3.5.2.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 7.3.2.

7.3.5.2.3 Resource Standard Methods

7.3.5.2.3.1 POST

This method shall support the request data structure specified in Table 7.3.5.2.3.1-1, and the response data structure and response code specified in Table 7.3.5.2.3.1-2.

Table 7.3.5.2.3.1-1: Data structures supported by the POST request body on this resource

Data Type	P	Cardinality	Description
DataJobInfo	M	1	Provides information for the data job to be created.

Table 7.3.5.2.3.1-2: Data structures supported by the POST response body on this resource

Data Type	P	Cardinality	Response codes	Description
DataJobInfo	M	1	201 Created	The operation was successful. The message content of the POST response contains a DataJobInfo representing the created resource.
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

Table 7.3.5.2.3.1-3: Headers supported by the 201-response code this resource

Name	Data type	P	Cardinality	Description
Location	String	M	1	Contains the URI of the newly created resource as defined in clause 7.3.5.2.2.

7.3.5.2.3.2 GET

This method shall support the request data structure specified in Table 7.3.5.2.3.2-1 and the response data structure and response code specified in Table 7.3.5.2.3.2-2.

Table 7.3.5.2.3.2-1: Data structure supported by the GET request body on this resource

Data type	P	Cardinality	Description
N/A			There is no object in the message content of a GET request.

Table 7.3.5.2.3.2-2: Data structures supported by the HTTP GET response body on this resource

Data type	P	Cardinality	Response codes	Description
array(dataJobId)	M	0..N	200 OK	All data job identifiers.
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.

7.3.5.2.4 Resource Custom Operations

None.

7.3.5.3 Resource: "Individual data job"

7.3.5.3.1 Description

The resource Individual data job represents an individual data job for a data request or data subscription.

Only the methods defined in clause 7.3.5.3.3 shall be supported by these resources.

7.3.5.3.2 Resource Definition

Resource URI: **{apiRoot}/data-access/<apiVersion>/data-jobs/{dataJobId}**

The resource URI variables supported by the resource are defined in Table 7.3.5.3.2-1.

Table 7.3.5.3.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 7.3.2.
dataJobId	The data job identifier assigned by the Service Producer.

7.3.5.3.3 Resource Standard Methods

7.3.5.3.3.1 DELETE

This method shall support the request data structure specified in Table 7.3.5.3.3.1-1 and the response data structure and response code specified in Table 7.3.5.3.3.1-2.

Table 7.3.5.3.3.1-1: Data structures supported by the DELETE request body on this resource

Data type	P	Cardinality	Description
N/A			There is no object in the message content of a DELETE request.

Table 7.3.5.3.3.1-2: Data structures supported by the DELETE response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No content	The operation was successful. The data request has been cancelled.
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

7.3.5.3.3.2 PUT

This method shall support the request data structure specified in Table 7.3.5.3.3.2-1 and the response data structure and response code specified in Table 7.3.5.3.3.2-2.

Table 7.3.5.3.3.2-1: Data structure supported by the PUT request body on this resource

Data type	P	Cardinality	Description
DataJobInfo	M	1	Provides update data job information.

Table 7.3.5.3.3.2-2: Data structure supported by the PUT response body on this resource

Data type	P	Cardinality	Response code	Description
DataJobInfo	M	1	200 OK	The operation was successful. Data job information updated successfully.
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

7.3.5.3.3.3 GET

This method shall support the request data structure specified in Table 7.3.5.3.3.3-1 and the response data structure and response code specified in Table 7.3.5.3.3.3-2.

Table 7.3.5.3.3.3-1: Data structure supported by the GET request body on this resource

Data type	P	Cardinality	Description
N/A			There is no object in the message content of a GET request.

Table 7.3.5.3.3.3-2: Data structure supported by the GET response body on this resource

Data type	P	Cardinality	Response code	Description
DataJobInfo	M	1	200 OK	The operation was successful.
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

7.3.5.3.4 Resource Custom Operations

None.

7.3.5.4 Resource: "Individual data job status"

7.3.5.4.1 Description

The resource Individual data job status represents an individual data job status for a data request or data subscription.

Only the methods defined in clause 7.3.5.4.3 shall be supported by these resources.

7.3.5.4.2 Resource Definition

Resource URI: **{apiRoot}/data-access/<apiVersion>/data-jobs/{dataJobId}/status**

The resource URI variables supported by the resource are defined in Table 7.3.5.4.2-1.

Table 7.3.5.4.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 7.3.2.
dataJobId	The data job identifier assigned by the Service Producer.

7.3.5.4.3 Resource Standard Methods

7.3.5.4.3.1 GET

This method shall support the request data structure specified in Table 7.3.5.4.3-1 and the response data structure and response code specified in Table 7.3.5.4.3-2.

Table 7.3.5.4.3-1: Data structure supported by the GET request body on this resource

Data type	P	Cardinality	Description
N/A			There is no object in the message content of a GET request.

Table 7.3.5.4.3-2: Data structure supported by the GET response body on this resource

Data type	P	Cardinality	Response code	Description
DataJobStatusInfo	M	1	200 OK	The operation was successful.
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

7.3.5.4.4 Resource Custom Operations

None.

7.3.6 Custom operation without associated resources

None.

7.3.7 Notifications

7.3.7.1 Notify data availability

7.3.7.1.1 Description

The notification informs the receiver about the availability of data for a data subscription and provides details about how to access them.

7.3.7.1.2 Resource Definition

The Resource URI is a callback URI provided when creating a data job for data subscription.

7.3.7.1.3 Resource Standard Methods

7.3.7.1.3.1 POST

This method shall support the request data structures specified in Table 7.3.7.1.3.1-1 and the response data structure and response codes specified in Table 7.3.7.1.3.1-2.

Table 7.3.7.1.3.1-1: Data structures supported by the HTTP POST request body on this resource

Data type	P	Cardinality	Description
DataAvailabilityNotification	M	1	Notify data availability

Table 7.3.7.1.3.1-2: Data structures supported by the HTTP POST response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No content	Confirmation of received notification
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

7.3.8 Data Model

7.3.8.1 Structured data types

7.3.8.1.1 Overview

The following clauses define the data type and attributes to be used in the resource representation.

7.3.8.1.2 Data type: DataJobInfo

The DataJobInfo contains the attributes defined in Table 7.3.8.1.2-1.

Table 7.3.8.1.2-1: Definition of type DataJobInfo

Attribute Name	Data type	P	Cardinality	Description
dataDeliveryMode	DataDeliveryMode	M	1	See clause 7.3.8.2.3.1.
dmeTypeId	DmeTypeId	M	1	See clause B.4.2.
productionJobDefinition	object	M	1	Job description based on the DME type specific dataProductionSchema.
dataDeliveryMethod	DataDeliveryMethod	M	1	See clause B.4.3.1.
dataDeliverySchemaId	string	M	1	A delivery schema identifier provided by a Data Producer during the data registration procedure.
pullDeliveryDetailsHttp	PullDeliveryDetailsHttp	C	0..1	See clause 7.3.8.1.3 (see note 1).
dataAvailabilityNotificationUri	Uri	C	0..1	Callback URI for data availability notifications (see note 2).
pushDeliveryDetailsHttp	PushDeliveryDetailsHttp	C	0..1	See clause 7.3.8.1.4 (see note 3).
streamingConfigurationKafka	StreamingConfigurationKafka	C	0..1	See clause 7.3.8.1.6 (see note 4).
dataJobInfoStatus	ProcessMonitor	C	0..1	ProcessMonitor datatype is specified in ETSI TS 128 622 [21] clause 4.3.43 (see note 5).

NOTE 1: If dataDeliveryMethod is PULL_HTTP and dataDeliveryMode is ONE_TIME, the pullDeliveryDetailsHttp attribute shall be present in the Create data job response.
NOTE 2: If dataDeliveryMethod is PULL_HTTP, and dataDeliveryMode is CONTINOUS, the dataAvailabilityNotificationUri attribute shall be present in the Create data job request and Create data job response.
NOTE 3: If dataDeliveryMethod is PUSH_HTTP, the pushDeliveryDetailsHttp attribute shall be present in the Create data job request and Create data job response.
NOTE 4: If dataDeliveryMethod is STREAMING_KAFKA, the streamingConfigurationKafka attribute shall be present in the Create data job request sent by the DME as API Consumer and in the Create data job response sent by the DME as API Producer. If the streamingConfigurationKafka attribute is present in the Create data job request, it shall be present in the corresponding Create data job response.
NOTE 5: When API Consumer requesting the datajob, the dataJobInfoStatus shall not present in the request. When API Producer creating a data job the response shall include the DataJobInfo with dataJobInfoStatus.

7.3.8.1.3 Data type: PullDeliveryDetailsHttp

The PullDeliveryDetailsHttp data type signals how to pull data using the HTTP protocol. It contains the attributes defined in Table 7.3.8.1.3-1.

Table 7.3.8.1.3-1: Definition of type PullDeliveryDetailsHttp

Attribute Name	Data type	P	Cardinality	Description
dataPullUri	Uri	M	1	URI which data can be pulled from.

7.3.8.1.4 Data type: PushDeliveryDetailsHttp

The PushDeliveryDetailsHttp data type signals how to push data using the HTTP protocol. It contains the attributes defined in Table 7.3.8.1.4-1.

Table 7.3.8.1.4-1: Definition of type PushDeliveryDetailsHttp

Attribute Name	Data type	P	Cardinality	Description
dataPushUri	Uri	M	1	URI to which data can be pushed.

7.3.8.1.5 Data type: DataAvailabilityNotification

The DataAvailabilityNotification contains the attributes defined in Table 7.3.8.1.5-1.

Table 7.3.8.1.5-1: Definition of type DataAvailabilityNotification

Attribute Name	Data type	P	Cardinality	Description
dataJobId	string	M	1	Data job identifier.
pullDeliveryDetailsHttp	PullDeliveryDetailsHttp	C	0..1	If the dataDeliveryMechanism attribute of the data job identified by the dataJobId attribute is PULL_HTTP, this attribute shall be included. Otherwise, it shall be absent.

7.3.8.1.6 Data type: StreamingConfigurationKafka

The StreamingConfigurationKafka data type signals a data streaming configuration for the Kafka protocol. It contains the attributes defined in Table 7.3.8.1.6-1.

Table 7.3.8.1.6-1: Definition of type StreamingConfigurationKafka

Attribute Name	Data type	P	Cardinality	Description
topicName	string	M	1	Name of the Kafka topic.
kafkaBootstrapServers	array (ServerAddressWithPort)	M	1.. N	See clause 7.3.8.1.7.

7.3.8.1.7 Data type: ServerAddressWithPort

The ServerAddressWithPort contains the attributes defined in Table 7.3.8.1.7-1.

Table 7.3.8.1.7-1: Definition of type ServerAddressWithPort

Attribute Name	Data type	P	Cardinality	Description
hostname	string	M	1	hostname shall follow DNS naming convention as defined in IETF RFC 1035 [19].
portAddress	integer	M	1	Port address, e.g. 9092.

7.3.8.2 Simple data types and enumerations

7.3.8.2.1 Introduction

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

7.3.8.2.2 Simple data types

None.

7.3.8.2.3 Enumerations

7.3.8.2.3.1 Enumeration: DataDeliveryMode

This indicates whether the data instance is created in a one-time data delivery (data request) or continuously (data subscription).

Table 7.3.8.2.3.1-1: Enumeration type of DataDeliveryMode

Enumerations Value	Description
ONE_TIME	indicate the data job to be created is for on-time data delivery, i.e. for data request.
CONTINUOUS	indicate the data job to be created is for continuous data delivery, i.e. for data subscription.

7.3.8.2.3.2 Void

7.3.8.3 Re-used data types

None.

7.3.8.4 Service-specific registration information

The following structure defines the content of the "serviceCapabilities" attribute in the "ServiceProperties" data type (see clause B.3.4.2) for registration and discovery of this service.

Table 7.3.8.4-1: Definition of the service-specific registration information

Attribute Name	Data type	P	Cardinality	Description
supportedDataDeliveryModes	array(DataDeliveryMode)	M	1..N	Indicates whether one-time or continuous data delivery, or both, are supported by the service.

7.3.9 Error Handling

7.3.9.1 General

For the Data access API, HTTP error responses shall be supported as specified in ETSI TS 129 501) [1]. Protocol errors and application errors specified in ETSI TS 129 500 [2], Table 5.2.7.2-1, shall be supported for an HTTP method if the corresponding HTTP status codes are specified as mandatory for that HTTP method in ETSI TS 129 500 [2], Table 5.2.7.1-1.

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

7.3.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

7.3.9.3 Application Errors

No additional application errors defined in the present document.

7.4 HTTP based Push data API

7.4.1 Introduction

This API enables the API Producer to push data to the API Consumer based on the Push data service procedures defined in R1GAP [5]. The API definition applies to both scenarios when rApp is the Service Consumer and when DME is the Service Consumer, respectively.

7.4.2 API version

For the Push data API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields).

7.4.3 Resource structure and methods

The resource URI is a callback URI provided when creating a data job or data offer.

7.4.4 Service Operations

7.4.4.1 Push data

7.4.4.1.1 Operation definition

The API Producer uses Push data operation to push data payload to the API Consumer.

The operation is based on HTTP POST.

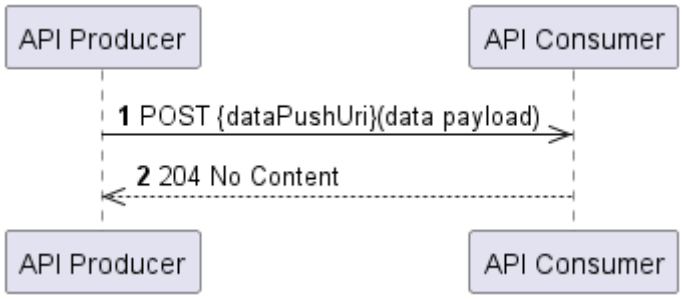

Figure 7.4.4.1.1-1: Push data operation

The service operation is as follows:

1) The API Producer shall send an HTTP POST request to the API Consumer. The target URI (dataPushUri) identifies the destination for pushing data to. The message content shall carry the data payload. The Content-Type header shall be present and set to the exact media type of the data payload.

2) The API Consumer shall return the HTTP POST response. On success, "204 No Content" shall be returned. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

7.4.4.1.2 Referenced procedures

7.4.4.1.2.1 Push data procedure

The Push data operation illustrated in Figure 7.4.4.1.1-1 is based on the Push data procedure defined for the Push data service in R1GAP [5].

7.4.5 Resources

7.4.5.1 Overview

This clause defines the resources for the Push data API.

7.4.5.2 Resource: "Push delivery URI"

7.4.5.2.1 Description

The resource represents the destination for pushing data to.

7.4.5.2.2 Resource Definition

The resource URI is a callback URI provided in the "PushDeliveryDetailsHttp" data structure when creating a data job for data subscription or a data offer.

7.4.5.2.3 Resource Standard Methods

7.4.5.2.3.1 POST

The method shall support carrying the data payload in the request body. The format of the data payload and Content-Type header are determined by the data message schema.

The method shall support the response data structures and response codes specified in Table 7.4.5.2.3.1-1.

Table 7.4.5.2.3.1-1: Data structures supported by the HTTP POST response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No content	Confirmation of received data delivery.
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

7.4.5.2.4 Resource Custom Operations

None.

7.4.6 Custom operation without associated resources

None.

7.4.7 Notifications

None.

7.4.8 Data Model

7.4.8.1 Structured data types

None.

7.4.8.2 Simple data types and enumerations

None.

7.4.8.3 Re-used data types

None.

7.4.8.4 Service-specific registration information

None.

7.4.9 Error Handling

7.4.9.1 General

For the Push data API, HTTP error responses shall be supported as specified in ETSI TS 129 501 [1]. Protocol errors and application errors specified in ETSI TS 129 500 [2], Table 5.2.7.2-1, shall be supported for an HTTP method if the corresponding HTTP status codes are specified as mandatory for that HTTP method in ETSI TS 129 500 [2], Table 5.2.7.1-1.

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

7.4.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

7.4.9.3 Application Errors

No additional application errors defined in the present document.

7.5 HTTP based Pull data API

7.5.1 Introduction

This API enables the API Consumer to pull data from the API Producer based on the Pull data service procedures defined in R1GAP [5]. The API definition applies to both scenarios when rApp is the Service Consumer and when DME is the Service Consumer, respectively.

7.5.2 API version

For the Pull data API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields).

7.5.3 Resource structure and methods

The resource URI is a target URI provided when creating a data job for data request or a data offer, or when notifying the data availability for data subscription.

7.5.4 Service Operations

7.5.4.1 Pull data

7.5.4.1.1 Operation definition

The API Consumer uses Pull data operation to pull data payload from the API Producer.

The operation is based on HTTP GET.

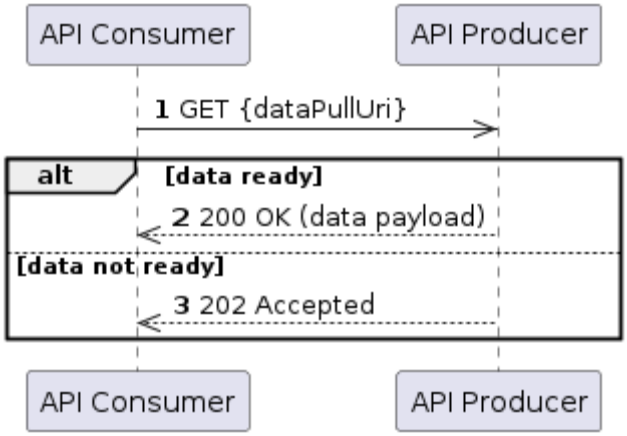

Figure 7.5.4.1.1-1: Pull data operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI (dataPullUri) identifies the destination for pulling data from.

2) The API Producer shall return the HTTP GET response. On success, "200 OK" shall be returned, the message content shall carry the data payload. The Content-Type header shall be present and set to the exact media type of the data payload.

3) If the data payload is not ready yet, "202 Accepted" shall be returned, with the Retry-After header optionally provided to indicate how long the API Consumer should wait before making a follow-up request. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

7.5.4.1.2 Referenced procedures

7.5.4.1.2.1 Pull data procedure

The Pull data operation illustrated in Figure 7.5.4.1.1-1 is based on the Retrieve data procedure defined for the Pull data service in R1GAP [5].

7.5.5 Resources

7.5.5.1 Overview

This clause defines the resources for the Pull data API.

7.5.5.2 Resource: "Pull delivery URI"

7.5.5.2.1 Description

The resource represents the destination for pulling data from.

7.5.5.2.2 Resource Definition

The resource URI is a target URI provided in the "PullDeliveryDetailsHttp" data structure when creating a data job for data request or a data offer, or when notifying the data availability for data subscription.

7.5.5.2.3 Resource Standard Methods

7.5.5.2.3.1 GET

This method shall support the request data structures specified in Table 7.5.5.2.3.1-1 the response data structures and response codes specified in Table 7.5.5.2.3.1-2 and the HTTP response headers as defined in Table 7.5.5.2.3.1-3.

Table 7.5.5.2.3.1-1: Data structures supported by the HTTP GET request body on this resource

Data type	P	Cardinality	Description
NA			There is no object in the message content of a GET request.

Table 7.5.5.2.3.1-2: Data structures supported by the HTTP GET response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			200 OK	Carry data payload in the response body.
N/A			202 Accepted	Data payload is not ready, retry later and the response body shall be empty.
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

The method shall support carrying the data payload in the response body of the "200OK" response. The format of the data payload and Content-Type header are determined by the data message schema.

Table 7.5.5.2.3.1-3: Headers supported by the 202 Response code on this resource

Name	Data type	P	Cardinality	Description
Retry-After	string	O	0..1	Indicates to the API Consumer the length of the time interval to wait until sending the next request, or the point in time when such request should earliest be sent. The format is defined in IETF RFC 9110 [25].

7.5.5.2.4 Resource Custom Operations

None.

7.5.6 Custom operation without associated resources

None.

7.5.7 Notifications

None.

7.5.8 Data Model

7.5.8.1 Structured data types

None.

7.5.8.2 Simple data types and enumerations

None.

7.5.8.3 Re-used data types

None.

7.5.8.4 Service-specific registration information

None.

7.5.9 Error Handling

7.5.9.1 General

For the Pull data API, HTTP error responses shall be supported as specified in clause 4.8 of ETSI TS 129 501 [1]. Protocol errors and application errors specified in ETSI TS 129 500 [2], Table 5.2.7.2-1, shall be supported for an HTTP method if the corresponding HTTP status codes are specified as mandatory for that HTTP method in ETSI TS 129 500 [2], Table 5.2.7.1-1.

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

7.5.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

7.5.9.3 Application Errors

No additional application errors defined in the present document.

7.6 Data offer API

7.6.1 Introduction

This API enables an API Consumer to trigger the API Producer to collect a data instance produced by the API Consumer and to store it for later consumption, based on the procedures for managing a data offer defined in R1GAP [5]. The API definition applies to the scenario when a Data Producer is the Service Consumer and the DME framework is the Service Producer.

7.6.2 API version

For the Data offer API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v1".

7.6.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined in clause 5.2. The <apiName> resource URI variable shall be "data-offer". The <apiSpecificResourceUriPart> for each resource shall be set as described in clause 7.6.5.

Figure 7.6.3-1 shows the overall resource URI structure defined for the Data offer API.

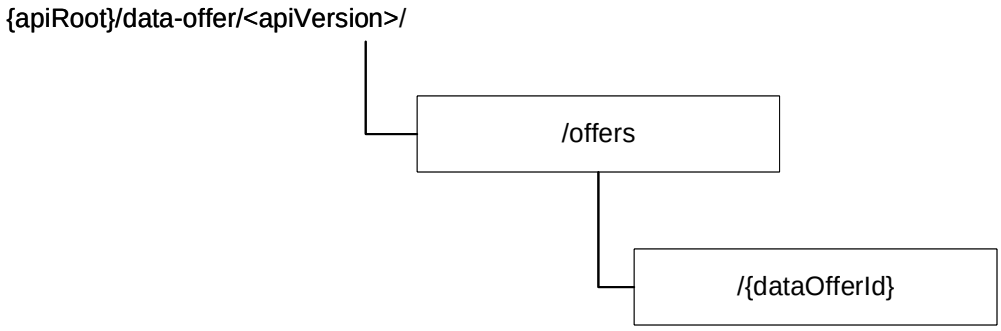

Figure 7.6.3-1: Resource URI structure of the Data offer API

Table 7.6.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 7.6.3-1: Resource and methods overview of the Data offer API

Resource name	Resource URI	HTTP method	Service Operation
All data offers	.../offers	POST	Create an individual data offer
Individual data offer	.../offers/{dataOfferId}	DELETE	Cancel data offer

7.6.4 Service Operations

7.6.4.1 Create data offer

7.6.4.1.1 Operation definition

The API Consumer uses the Create data offer operation to create a data offer.

The operation is based on HTTP POST.

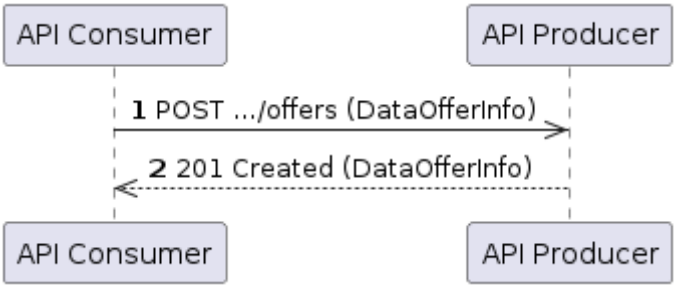

Figure 7.6.4.1.1-1: Create data offer operation

The service operation is as follows:

1) The API Consumer shall send an HTTP POST request to the API Producer. The target URI shall identify the resource (…/offers) under which the new data offer is to be created. The message content shall carry a DataOfferInfo structure.

2) The API Producer shall return the HTTP POST response. On success, "201 Created" shall be returned. The Location header shall be present and shall carry the URI of the new data offer resource with dataOfferId assigned by the API producer. The message content shall carry a DataOfferInfo representing the created data offer. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

7.6.4.1.2 Referenced procedures

7.6.4.1.2.1 Create data offer procedure

The Create data offer operation illustrated in Figure 7.6.4.1.1-1 is based on the Create data offer procedure defined for the Data offer service in R1GAP [5].

7.6.4.2 Cancel data offer

7.6.4.2.1 Operation definition

The API Consumer uses the Cancel data offer operation to cancel a data offer, i.e. to indicate to the API Producer that it has stopped the delivery of data for the data offer.

The operation is based on HTTP DELETE.

Figure 7.6.4.2.1-1: Cancel data offer operation

The service operation is as follows:

1) The API Consumer shall send an HTTP DELETE request to the API Producer. The target URI shall identify the resource (…/offers/{dataOfferId}).

2) The API Producer shall return the HTTP DELETE response. On success, "204 No Content" shall be returned. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

NOTE: For some data delivery methods, due to race conditions, residual produced data can arrive at the API Producer even after having received the HTTP DELETE request. The API Producer should be robust against this situation.

7.6.4.2.2 Referenced procedures

7.6.4.2.2.1 Cancel data offer procedure

The Cancel data offer operation illustrated in Figure 7.6.4.2.1-1 is based on the cancel data offer procedure defined for the Data offer service in R1GAP [5].

7.6.4.3 Notify data offer termination

7.6.4.3.1 Operation definition

The API Producer uses this operation to notify the API Consumer that it has stopped collecting the offered data. The API Consumer stops producing data.

The operation is based on HTTP POST.

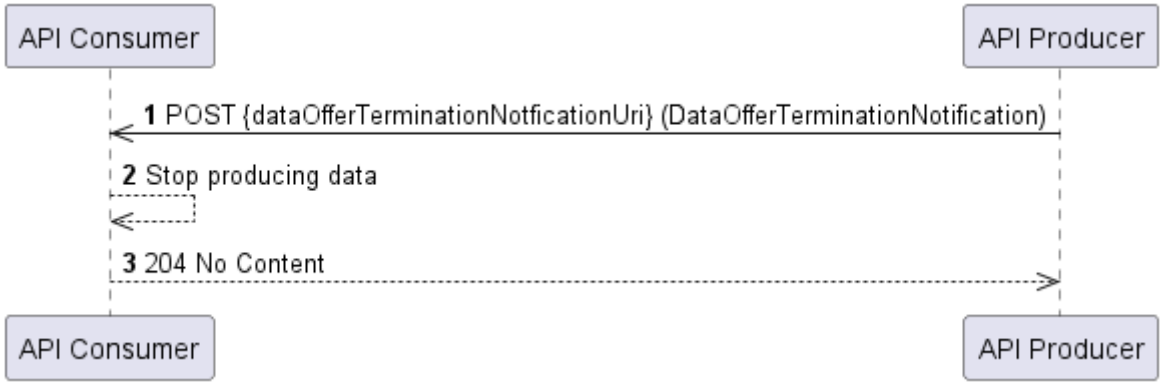

Figure 7.6.4.3.1-1: Notify data offer termination

The service operation is as follows:

1) The API Producer shall send an HTTP POST request to the API Consumer. The target URI (dataOfferTerminationNotificationUri which was provided during data offer creation) identifies the address where to send the notifications. The message content shall carry a DataOfferTerminationNotification.

2) The API Producer shall stop producing data.

3) The API Consumer shall return the HTTP POST response. On success, "204 No Content" shall be returned. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

NOTE: For some data delivery methods, due to race conditions, residual produced data can arrive at the API Producer even after having received the "204 No Content" response. The API Producer should be robust against this situation.

7.6.4.3.2 Referenced procedures

7.6.4.3.2.1 Notify data availability procedure

The Notify data offer termination operation illustrated in Figure 7.6.4.3.1-1 is based on the Notify data offer termination procedure defined for the Data offer service in R1GAP [5].

7.6.5 Resources

7.6.5.1 Overview

This clause defines the resources for the Data offer API.

7.6.5.2 Resource: "All data offers"

7.6.5.2.1 Description

The resource All data offers represents all data offers created by a particular consumer.

Only the methods defined in clause 7.6.5.2.3 shall be supported by these resources.

7.6.5.2.2 Resource Definition

Resource URI: **{apiRoot}/data-offer/<apiVersion>/offers**

The resource URI variables supported by the resource are defined in Table 7.6.5.2.2-1.

Table 7.6.5.2.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.3.
apiVersion	See clause 7.6.2.

7.6.5.2.3 Resource Standard Methods

7.6.5.2.3.1 POST

This method shall support the request data structure specified in Table 7.6.5.2.3.1-1, and the response data structure and response code specified in Table 7.6.5.2.3.1-2.

Table 7.6.5.2.3.1-1: Data structures supported by the POST request body on all data jobs

Data Type	P	Cardinality	Description
DataOfferInfo	M	1	Provides information for the data job to be created.

Table 7.6.5.2.3.1-2: Data structures supported by the POST response body on all data jobs

Data Type	P	Cardinality	Response codes	Description
DataOfferInfo	M	1	201 Created	The operation was successful. The message content of the POST response contains a DataOfferInfo representing the created resource.
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

Table 7.6.5.2.3.1-3: Headers supported by the 201-response code on the resource

Name	Data type	P	Cardinality	Description
Location	String	M	1	Contains the URI of the newly created resource as defined in clause 7.6.5.3.2.

7.6.5.2.4 Resource Custom Operations

None.

7.6.5.3 Resource: "Individual data offer"

7.6.5.3.1 Description

The resource Individual data offer represents an individual data offer.

The methods defined in clause 7.6.5.3.3 shall be supported by this resource.

7.6.5.3.2 Resource Definition

Resource URI: **{apiRoot}/data-offer /<apiVersion>/offers/{dataOfferId}**

The resource URI variables supported by the resource is defined in Table 7.6.5.3.2-1.

Table 7.6.5.3.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.3.
apiVersion	See clause 7.6.2.
dataOfferId	The data offer identifier assigned by the Service Producer.

7.6.5.3.3 Resource Standard Methods

7.6.5.3.3.1 DELETE

This method shall support the request data structure specified in Table 7.6.5.3.3.1-1 and the response data structure and response code specified in Table 7.6.5.3.3.1-2.

Table 7.6.5.3.3.1-1: Data structure supported by the DELETE request body on this resource

Data type	P	Cardinality	Description
N/A			There is no object in the message content of a DELETE request.

Table 7.6.5.3.3.1-2: Data structure supported by the DELETE response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No content	The operation was successful. The data offer has been cancelled.
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

7.6.5.3.4 Resource Custom Operations

None.

7.6.6 Custom operation without associated resources

None.

7.6.7 Notifications

7.6.7.1 Notify data offer termination

7.6.7.1.1 Description

The notification informs the Data Producer as API Consumer that the sender of the notification, i.e. the Api Producer, does not intend to collect the data instance related to the data offer from the Data Producer any longer. The Data Producer shall stop data production when receiving this notification.

7.6.7.1.2 Resource Definition

The resource URI is a callback URI that has been provided when creating the related data offer.

7.6.7.1.3 Resource Standard Methods

7.6.7.1.3.1 POST

This method shall support the request data structures specified in Table 7.6.7.1.3.1-1 and the response data structure and response codes specified in Table 7.6.7.1.3.1-2.

Table 7.6.7.1.3.1-1: Data structures supported by the HTTP POST request body on this resource

Data type	P	Cardinality	Description
DataOfferTerminationNotification	M	1	Notify data offer termination.

Table 7.6.7.1.3.1-2: Data structures supported by the HTTP POST response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No content	Confirmation of received notification.
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

7.6.8 Data Model

7.6.8.1 Structured data types

7.6.8.1.1 Overview

The following clauses define the data types and attributes to be used in the resource representation.

7.6.8.1.2 Data type: DataOfferInfo

The DataOfferInfo contains the attributes defined in Table 7.6.8.1.2-1.

Table 7.6.8.1.2-1: Definition of type DataOfferInfo

Attribute Name	Data type	P	Cardinality	Description
dataDeliveryMode	DataDeliveryMode	M	1	See clause 7.3.8.2.3.1
dmeTypeId	DmeTypeId	M	1	See clause B.4.2
productionJobDefinition	object	M	1	Job definition based on the DME type specific dataProductionSchema
dataDeliveryMethods	array (DataDeliveryMethod)	M	1..N	Data delivery method(s), see clause B.4.3.1 (see note).
dataDeliverySchemaIds	array (string)	M	1..N	Delivery schema identifiers (see note).
pullDeliveryDetailsHttp	PullDeliveryDetailsHttp	C	0..1	Access details for HTTP Pull delivery of the data, managed by the API Consumer, see clause 7.3.8.1.3. The API Consumer shall provide this attribute in the HTTP request to create a data offer if the value "PULL_HTTP" is included in "dataDeliveryMechanisms". The API Producer shall include this attribute in HTTP responses if it has chosen "PULL_HTTP" data delivery.

Attribute Name	Data type	P	Cardinality	Description
dataAvailabilityNotificationUri	Uri	C	0..1	Callback URI to receive data availability notifications, managed by the API Producer. The API Consumer shall not include this attribute in the HTTP request to create a data offer. The API Producer shall include this attribute in HTTP responses if it has chosen "PULL_HTTP" data delivery in "CONTINUOUS" delivery mode.
dataOfferTerminationNotificationUri	Uri	M	1	Callback URI to receive data offer termination notifications, managed by the API Consumer.
pushDeliveryDetailsHttp	PushDeliveryDetailsHttp	C	0..1	Access details for HTTP Push delivery of the data, managed by the API Producer, see clause 7.3.8.1.4. The API Consumer shall not provide this attribute in the HTTP request to create a data offer. The API Producer shall include this attribute in HTTP responses if it has chosen "PUSH_HTTP" data delivery.
streamingConfigurationKafka	StreamingConfigurationKafka	C	0..1	Configuration to stream data over a Kafka bus, see clause 7.3.8.1.6. This attribute shall not be present in the HTTP request to create a data offer. If the value "STREAMING_KAFKA" is included in "dataDeliveryMechanisms", this attribute shall be included in HTTP responses.
NOTE: In the HTTP request to create a data offer, the API Consumer shall include all items it supports. In HTTP responses, the API Producer shall include the single item that has been chosen from the alternatives proposed during data offer creation.				

7.6.8.1.3 Data type: DataOfferTerminationNotification

The DataOfferTerminationNotification contains the attributes defined in Table 7.6.8.1.3-1.

Table 7.6.8.1.3-1: Definition of type DataOfferTerminationNotification

Attribute Name	Data type	P	Cardinality	Description
dataOfferId	string	M	1	Identifier of the data offer to be terminated.

7.6.8.1.4 Data type: DataAvailabilityNotification

The DataAvailabilityNotification contains the attributes defined in Table 7.6.8.1.4-1.

Table 7.6.8.1.4-1: Definition of type DataAvailabilityNotification

Attribute Name	Data type	P	Cardinality	Description
dataOfferId	string	M	1	Identifier of the data offer to be terminated.
pullDeliveryDetailsHttp	PushDeliveryDetailsHttp	C	1	Access details for HTTP Push delivery of the data, managed by the API Producer, see clause 7.3.8.1.4. The API Consumer shall not provide this attribute in the HTTP request to create a data offer. The API Producer shall include this attribute in HTTP responses if it has chosen "PUSH_HTTP" data delivery.

7.6.8.2 Simple data types and enumerations

None.

7.6.8.3 Re-used data types

Table 7.6.8.3-1 specified data types re-used from the Data Access API.

Table 7.6.8.3-1: Re-used data types

Data type	Reference	Comments
DataDeliveryMethod	Clause B.4.3.1	
PullDeliveryDetailsHttp	Clause 7.3.8.1.3	
PushDeliveryDetailsHttp	Clause 7.3.8.1.4	
DataAvailabilityNotification	Clause 7.6.8.1.4	
StreamingConfigurationKafka	Clause 7.3.8.1.6	

7.6.8.4 Service-specific registration information

None.

7.6.9 Error Handling

7.6.9.1 General

For the Data offer API, HTTP error responses shall be supported as specified in clause 4.8 of ETSI TS 129 501 [1]. Protocol errors and application errors specified in ETSI TS 129 500 [2], Table 5.2.7.2-1, shall be supported for an HTTP method if the corresponding HTTP status codes are specified as mandatory for that HTTP method in ETSI TS 129 500 [2], Table 5.2.7.1-1.

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

7.6.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

7.6.9.3 Application Errors

No additional application errors defined in the present document.

8 RAN OAM related services

8.1 Configuration management API

8.1.1 Introduction

This API allows the API Consumer to request managing configuration data based on the procedures for "Configuration Management (CM) service" defined in R1GAP [5].

8.1.2 API version

For the Configuration management API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI variable shall be set to "v1".

The Configuration management API is under development and consequently the API version shall include the pre-release version "alpha.1".

8.1.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined for the Generic provisioning management service API (see ETSI TS 128 532 [20], Figure 12.1.1.3.1.1-1 in clause 12.1.1.3.1.1).

Table 8.1.3-1 lists the individual resources defined for the API, the applicable HTTP methods as defined in ETSI TS 128 532 [20], clause 12.1.1, and the associated service operations. Table 8.1.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 8.1.3-1: Resources and methods overview of the configuration management API

Resource Name ETSI TS 128 532 [20]	Resource URI ETSI TS 128 532 [20]	HTTP method ETSI TS 128 532 [20]	Service Operation
MOI	.../{URI-LDN-first part}/{class Name}={id}	GET	Read configuration data.
		PATCH	Write configuration changes.

NOTE: CM service procedures related to M-Plane nodes, such as O-RUs, are not currently supported in this API resource structure and can be defined in future releases.

8.1.4 Service operations

8.1.4.1 Read configuration data

8.1.4.1.1 Operation definition

A Service Consumer uses the Read configuration data API operation as API Consumer to read configuration with the API Producer.

The operation is based on HTTP GET as per Figure 8.1.4.1.1-1. The HTTP GET response contains configuration data that the API Consumer has requested.

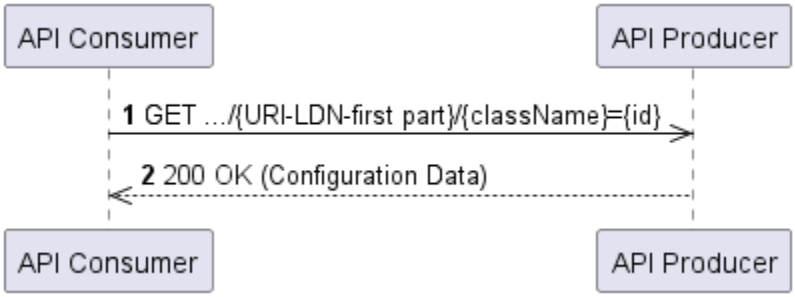

Figure 8.1.4.1.1-1: Read configuration data operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer that includes the Node Identifier and optional query criteria. The API Producer shall process the read information received in the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success "200 OK" shall be returned and the message content shall carry configuration data. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

8.1.4.1.2 Referenced procedures

8.1.4.1.2.1 Reading configuration data procedure

The procedure for reading configuration data API operation illustrated in Figure 8.1.4.1.1-1 is based on Read Configuration procedure defined in R1GAP [5].

8.1.4.2 Write configuration changes

8.1.4.2.1 Operation definition

A Service Consumer uses the Write configuration changes API operation as API Consumer to write configuration with the API Producer.

The operation is based on HTTP PATCH as per Figure 8.1.4.2.1-1. The HTTP PATCH response contains configuration data that the API Consumer has requested.

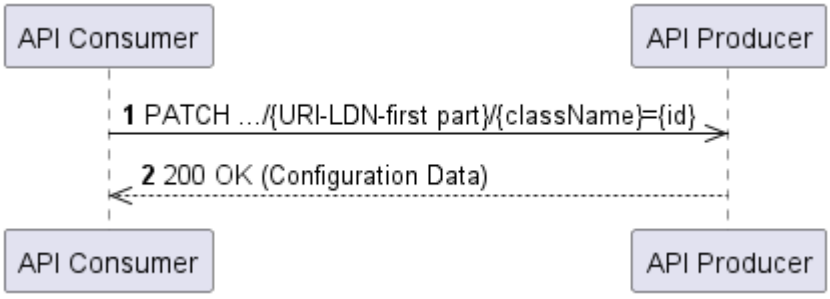

Figure 8.1.4.2.1-1: Write configuration changes operation

The service operation is as follows:

1) The API Consumer shall send an HTTP PATCH request to the API Producer that includes the Node Identifier and optional query criteria along with payload which includes attribute identifiers, attribute values and modify operator as part of modification list defined in ETSI TS 128 532 [20], clause 11.1.1.3.2. The API Producer shall process the write configuration changes information received in the HTTP PATCH message and determine if the request sent by the API consumer is authorized or not.

2) The API Producer shall return the HTTP PATCH response. On success "200 OK" shall be returned and the message content shall carry configuration data. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

NOTE: Response format to represent "partial success" is not specified in the present document.

8.1.4.2.2 Referenced procedures

8.1.4.2.2.1 Writing configuration change procedure

The procedure for writing configuration data API operation illustrated in Figure 8.1.4.2.1-1 is based on write Configuration procedure defined in R1GAP [5].

8.1.5 Resources

8.1.5.1 Overview

This clause defines the resources for the Configuration Management (CM) service API based on the RESTful HTTP-based solution set for the Generic provisioning management service defined in ETSI TS 128 532 [20], clause 12.1.1.

The Configuration Management (CM) service Producer take the role of the Generic provisioning MnS Producer defined in ETSI TS 128 532 [20]. By consuming this API, an rApp as Configuration Management (CM) service Consumer takes the role of the Generic provisioning MnS Consumer defined in ETSI TS 128 532 [20].

8.1.5.2 Resource: "MOI"

8.1.5.2.1 Description

The MOI resource represents a managed object instance, or a tree of these, containing the configuration data.

8.1.5.2.2 Resource definition

The resource URI structure of the provisioning MnS is defined in ETSI TS 128 532 [20], as per clause 12.1.1.3.1.1, with the following URI:

{apiRootProvMnS/<apiVersion>/{URI-LDN-first-part}/{className}={id}

The resource URI variables are specified in ETSI TS 128 532 [20], clause 12.1.1.3.2.1.2. HTTP methods GET and PATCH shall only be used.

8.1.5.2.3 Resource Standard Methods

8.1.5.2.3.1 GET

This method shall support the URI query parameters, request data structures, response data structures and response codes specified in ETSI TS 128 532 [20], clause 12.1.1.3.2.1.3.2.

8.1.5.2.3.2 PATCH

This method shall support the URI parameters, request data structures, response data structures and response codes specified in ETSI TS 128 532 [20], clause 12.1.1.3.2.1.3.3.

8.1.5.2.4 Resource custom operations

None.

8.1.6 Custom operations without associated resources

None.

8.1.7 Notifications

NOTE: The notifications are not specified in the present document.

8.1.8 Data model

8.1.8.1 General

The application data model is defined in ETSI TS 128 532 [20], clause 12.1.1.4, apply to this API.

8.1.8.2 Structured data types

None.

8.1.8.3 Simple data types and enumerations

None.

8.1.8.4 Re-used data types

None.

8.1.8.5 Service-specific registration information

None.

8.2 Fault management API

8.2.1 Introduction

This API allows the API Consumer to read information about alarms and to acknowledge alarms based on the procedures for the "Fault Management (FM) service" defined in R1GAP [5]. The API is based on the AlarmList IOC as specified in ETSI TS 128 111 [26], clause 7.3.2.

8.2.2 API version

For the fault management API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1]for a definition of the version fields). Consequently, the <apiVersion> URI variable shall be set to "v1".

8.2.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined for the Generic provisioning management service API (see ETSI TS 128 532 [20], Figure 12.1.1.3.1.1-1 in clause 12.1.1.3.1.1).

Table 8.2.3-1 lists the individual resources defined for the API, the applicable HTTP methods as defined in ETSI TS 128 532 [20], clause 12.1.1, and the associated service operations.

Table 8.2.3-1: Resources and methods overview of the fault management API

Resource Name ETSI TS 128 532 [20]	Resource URI ETSI TS 128 532 [20]	HTTP method ETSI TS 128 532 [20]	Service Operation
Alarm list	.../{URI-LDN-first part}/{className}={id}/AlarmList={alarmListId}	GET	Query alarms.
		PATCH	Change alarm acknowledgement status.

NOTE: FM service procedures related to M-Plane nodes, such as O-RUs, are not currently supported in this API resource structure and can be defined in future releases.

8.2.4 Service operations

8.2.4.1 Query alarms

8.2.4.1.1 Operation definition

A Service Consumer uses the Query alarms API operation as API Consumer to query alarm information from the API Producer.

The operation is based on HTTP GET as per Figure 8.2.4.1.1-1. The HTTP GET response contains a list of alarm records matching the query of the API Consumer.

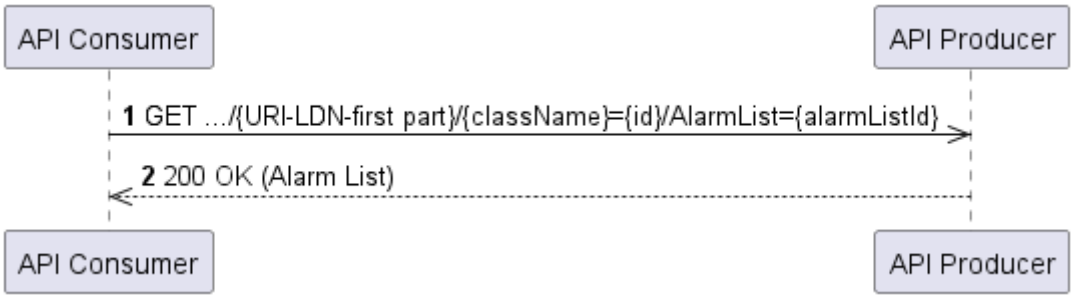

Figure 8.2.4.1.1-1: Query alarms operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer that includes either "SubNetwork" or "ManagedElement" as the class name, the related object instance identifier, the alarm list identifier, and optional query criteria. The API Producer shall process the information received in the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success "200 OK" shall be returned and the message content shall carry the representation of the alarm list. If query criteria were given, the content of the alarm entries in the returned alarm list shall match the query criteria. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

8.2.4.1.2 Referenced procedures

8.2.4.1.2.1 Query alarms procedure

The procedure for querying alarm information illustrated in Figure 8.2.4.1.1-1 is based on the Query Alarms procedure defined in R1GAP [5].

8.2.4.2 Change alarm acknowledgement state

8.2.4.2.1 Operation definition

A Service Consumer uses the Change alarm acknowledgement state API operation as API Consumer to acknowledge or unacknowledge one or more alarms with the API Producer.

The operation is based on HTTP PATCH as per Figure 8.2.4.2.1-1.

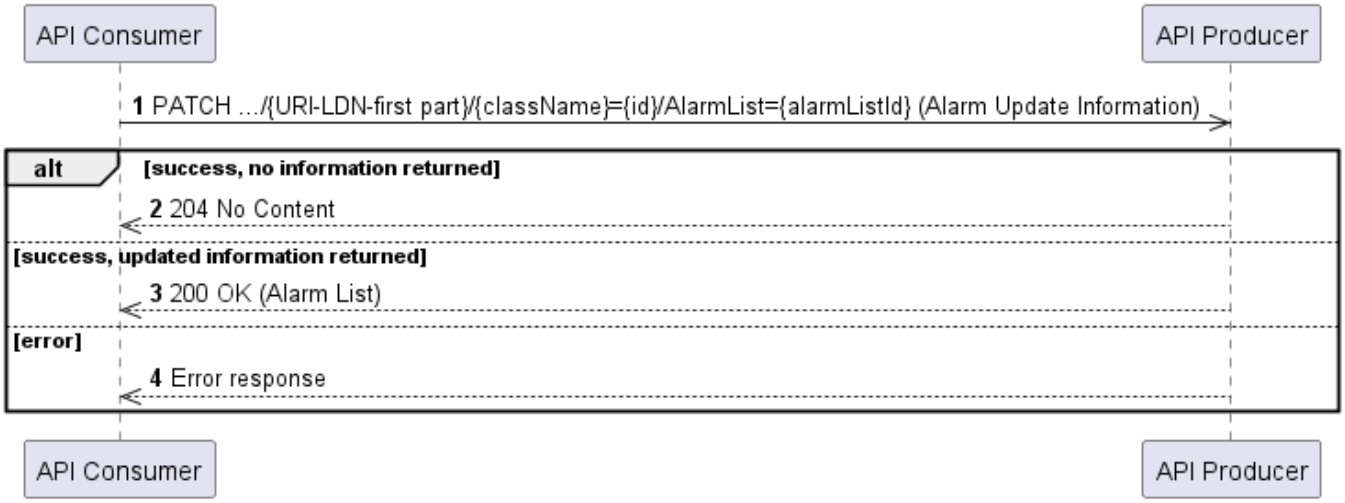

Figure 8.2.4.2.1-1: Change alarm acknowledgement state operation

The service operation is as follows:

1) The API Consumer shall send a PATCH request to the API Producer that includes in the resource URI either "SubNetwork" or "ManagedElement" as the class name, the related object instance identifier, and the alarm list identifier. Further, it shall include in the message content information about which alarms to be updated as well as information about the actual alarm acknowledgements / unacknowledgments. The API Producer shall process the information received in the HTTP PATCH message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP PATCH response.

3) On success, the API Producer should return a "204 No Content" response message with empty message content.

4) On success, the API Producer may alternatively return a "200 OK" response message with the representation of updated alarm list in the message content.

5) On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

8.2.4.2.2 Referenced procedures

8.2.4.2.2.1 Change alarm acknowledgement state procedure

The procedure for changing the acknowledgement state of one or more alarms illustrated in Figure 8.2.4.2.1-1 is based on the Change alarm acknowledgement state procedure defined in R1GAP [5].

8.2.5 Resources

8.2.5.1 Overview

This clause defines the resources for the Fault Management (FM) service API based on the RESTful HTTP-based solution set for the Generic provisioning management service defined in ETSI TS 128 532 [20], clause 12.1.1.

The Fault Management (FM) service Producer take the role of the Generic provisioning MnS Producer defined in ETSI TS 128 532 [20]. By consuming this API, an rApp as Fault Management (FM) service Consumer takes the role of the Generic provisioning MnS Consumer defined in ETSI TS 128 532 [20].

8.2.5.2 Resource: "Alarm list"

8.2.5.2.1 Description

The "Alarm list" resource represents a list of alarms related to either a subnetwork or a managed element. It allows to query alarms and to change the acknowledgement state of alarms.

8.2.5.2.2 Resource definition

The resource URI structure of the alarm list as managed by the Generic provisioning MnS is defined as per ETSI TS 128 532 [20], clause 12.1.1.3.1.1, for the AlarmList IOC as defined in ETSI TS 128 111 [26], clause 7.3.2, with the following URI:

{apiRoot}/ProvMnS/<apiVersion>/{URI-LDN-first-part}/{className}={id}/AlarmList={alarmListId}

The resource URI variables supported by the resource are defined in Table 8.2.5.2.2-1.

Table 8.2.5.2.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 8.2.2.
URI-LDN-first-part	See ETSI TS 128 532 [20], clause 12.1.1.3.2.1.2.
className	Name of the object class (IOC) of the managed object instance to which the alarm list is attached. Shall be either "SubNetwork" or "ManagedElement" as defined in ETSI TS 128 622 [21], clause 4.3.26.1.
id	Identifier of the managed object instance to which the alarm list is attached.
alarmListId	Identifier of the managed object instance which is the alarm list.

8.2.5.2.3 Resource Standard Methods

8.2.5.2.3.1 GET

This method shall support the URI query parameters, HTTP headers, response data structures and response codes specified in ETSI TS 128 532 [20], clause 12.1.1.3.2.1.3.2.

EXAMPLE 1: This request queries the whole alarm list: GET /SubNetwork=SN123/AlarmList=AL123 HTTP/1.1.

EXAMPLE 2: This request queries a single alarm: GET /SubNetwork=SN123/AlarmList=AL123?fields=/attributes/alarmRecords/ALARM456 HTTP/1.1.

NOTE: Selection of alarm records in the query based on then values of other alarm attributes than alarm identifier is not supported in the present version.

8.2.5.2.3.2 PATCH

This method shall support the URI query parameters, HTTP headers, response data structures and response codes specified in ETSI TS 128 532 [20], clause 12.1.1.3.2.1.3.3.

To change the alarm state, the patch payload shall contain values for the "ackState" attribute ("ACKNOWLEDGED" or "UNACKNOWLEDGED") and the "ackUserId" attribute and may contain a value for the "ackSystemId" attribute, as defined in ETSI TS 128 111 [26], clause 7.3.1.2.

8.2.5.2.4 Resource custom operations

None.

8.2.6 Custom operations without associated resources

None.

8.2.7 Notifications

None.

8.2.8 Data model

8.2.8.1 General

The application data model is defined in ETSI TS 128 111 [26], clause A.1.3. The key IOCs for this API are SubNetwork, ManagedElement and AlarmList.

8.2.8.2 Structured data types

None.

8.2.8.3 Simple data types and enumerations

None.

8.2.8.4 Re-used data types

None.

8.2.8.5 Service-specific registration information

None.

8.2.9 Error Handling

8.2.9.1 General

HTTP error handling is applicable for this API as specified in ETSI TS 128 111 [26], clause A.1.3.

8.2.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

8.2.9.3 Application Errors

No specific protocol errors are defined in the present document.

8.3 Configuration schema information API

8.3.1 Introduction

This API enables the API Consumer to retrieve configuration schema information based on the procedure defined in R1GAP [5].

8.3.2 API version

For the configuration schema information API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see clause 4.3.1.1 of ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v1".

The configuration schema information API is under development and consequently the API version shall include the pre-release version "alpha.1".

8.3.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined in clause 5.2. The <apiName> resource URI variable shall be "ran-oam-cm-schema-info". The <apiSpecificResourceUriPart> for each resource shall be set as described in clause 8.3.5.

Figure 8.3.3-1 shows the overall resource URI structure defined for the retrieving configuration information API.

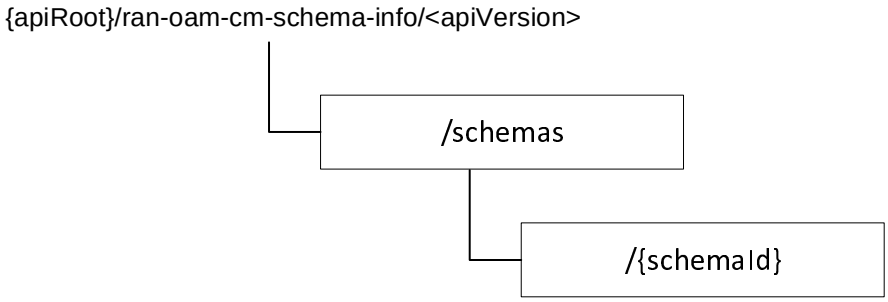

Figure 8.3.3-1: Resource URI structure of the configuration schemas information API

Table 8.3.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 8.3.3-1: Resource and methods overview of the configuration schemas information API

Resource Name	Resource URI	HTTP method	Service Operation
All schema information	…/schemas	GET	Get all schema information
Individual schema information	…/schemas/{schemaId}	GET	Get Individual schema information

8.3.4 Service operations

8.3.4.1 Get all schema information

8.3.4.1.1 Operation definition

An API consumer uses the Get all schema information operation to get Schemas.

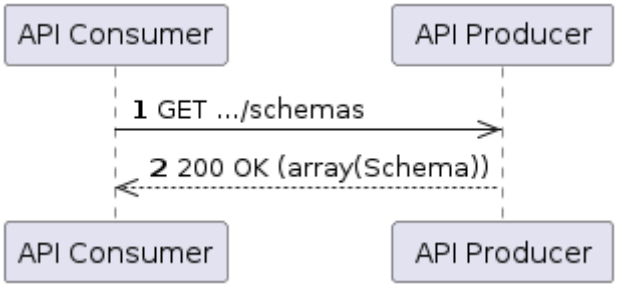

Figure 8.3.4.1.1-1: Get all schema information operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to API Producer. The target URI shall identify the resource (…/schemas). The API Producer shall process the request received in the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success "200 OK" shall be returned and the message content shall carry an array of schema. On failure, the appropriate error code shall be returned, and the message response body may contain additional error information.

8.3.4.1.2 Referenced procedures

8.3.4.1.2.1 Get schemas procedure

The Get all schema information operation illustrated in Figure 8.3.4.1.1-1 is based on Get schemas procedure defined in R1GAP [5].

8.3.4.2 Get Individual schema information

8.3.4.2.1 Operation definition

An API consumer uses the Get Individual schema information to get one or more schemas of interest for a schemaId.

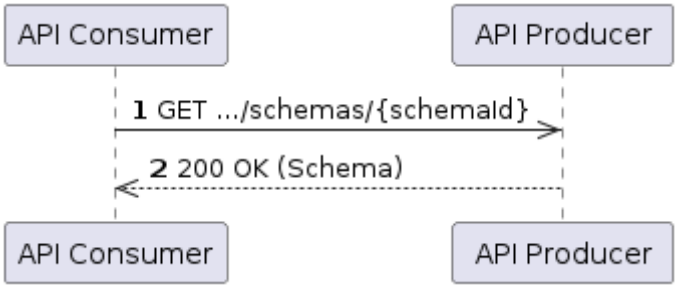

Figure 8.3.4.2.1-1: retrieving specific schema API operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to API Producer. The target URI shall identify the resource (…/schemas/{schemaId}), the message content shall be empty. The API Producer shall process the request received in the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success "200 OK" shall be returned and the message content shall carry the Schema matching the API Consumer HTTP GET request URI and schemaId. On failure, the appropriate error code shall be returned, and the message response body may contain additional error information.

8.3.4.2.2 Referenced procedures

8.3.4.2.2.1 Get Schemas procedure

The Get individual schema information operation illustrated in Figure 8.3.4.2.1-1 is based on Get schemas procedure defined in R1GAP [5].

8.3.5 Resources

8.3.5.1 Overview

This clause defines the resources for the retrieving configuration schemas API.

8.3.5.2 Resource: "All schema information"

8.3.5.2.1 Description

The resource represents a collection of configuration schema information.

Only the methods defined in clause 8.3.3 shall be supported by this resource.

8.3.5.2.2 Resource Definition

Resource URI: **{apiRoot}/ran-oam-cm-schema-info/<apiVersion>/schemas**

The resource URI variables supported by the resource is defined in Table 8.3.5.2.2-1.

Table 8.3.5.2.2-1: Resource URI variables

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 8.3.2.

8.3.5.2.3 Resource Standard Methods

8.3.5.2.3.1 HTTP GET

Table 8.3.5.2.3.1-1: Data structures supported by the GET Response Body on this resource

Data Type	P	Cardinality	Response codes	Description
array(Schema)	O	0..N	200 OK	The results with the schema information.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

8.3.5.2.4 Resource Custom Operations

None.

8.3.5.3 Resource: "Individual schema information"

8.3.5.3.1 Description

The resource represents a schema definition.

Only the methods defined in clause 8.3.3 shall be supported by this resource.

8.3.5.3.2 Resource Definition

Resource URI: **{apiRoot}/ran-oam-cm-schema-info/<apiVersion>/schemas/{schemaId}**

The resource URI variables supported by the resource is defined in Table 8.3.5.3.2-1.

Table 8.3.5.3.2-1: Resource URI variables

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 8.3.2.
schemaId	Where schemaId = "name" (for latest revision or if revision is not available) or "name@revision" for a specific revision.

8.3.5.3.3 Resource Standard Methods

8.3.5.3.3.1 HTTP GET

This method shall support the response data structure and response codes as specified in Table 8.3.5.3.3.1-1.

Table 8.3.5.3.3.1-1: Data structures supported by the GET Response Body on this resource

Data Type	P	Cardinality	Response codes	Description
Schema	M	1	200 OK	The results with the schema information matching the request.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

8.3.5.3.4 Resource Custom Operations

None.

8.3.6 Custom operations without associated resources

None

8.3.7 Notifications

None

8.3.8 Data model

8.3.8.0 Overview

The following clause specifies the application data model supported by the retrieving configuration schemas API.

8.3.8.1 Data Type: Schema

The Schema data type contains the attributes defined in Table 8.3.8.1-1.

Table 8.3.8.1-1: Definition of type Schema

Attribute Name	Data type	P	Cardinality	Description
name	string	M	1	The unique name of the schema.
revision	string	O	0..1	The unique revision of the schema.
location	Uri	M	1	Location of where to find the schema content.
type	SchemaType	M	1	See clause 8.3.8.2.3.1 (see note 2).
namespace	string	O	0..1	The namespace of the schema (see note 1).
features	Array(string)	O	0..N	List of supported features (see note 1).
deviations	Array(string)	O	0..N	List of schema deviation(s) (see note 1).
NOTE 1: Only applicable if type is YANG. NOTE 2: This data type is designed to provide extensibility. As specified in clause 8.3.8.2.3.1, Only YANG is supported in the present version of the present document.				

8.3.8.2 Simple data types and enumerations

8.3.8.2.1 Introduction

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

8.3.8.2.2 Simple data types

None.

8.3.8.2.3 Enumerations

8.3.8.2.3.1 Enumeration: SchemaTypes

Table 8.3.8.2.3.1-1: Enumeration SchemaTypes

Enumeration value	Description
YANG	YANG schema

8.3.8.3 Re-used data types

None.

8.3.9 Error Handling

8.3.9.1 General

In addition to the general provisions in clause 5.4.3, the requirements in the following clauses are applicable for the retrieving configuration schema API.

8.3.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

8.3.9.3 Application Errors

No specific protocol errors are defined in the present document.

9 A1 related services

9.1 A1 policy management API

9.1.1 Introduction

This API allows the API Consumer to Query policy types and create, query, update and delete A1 policies based on the procedures for "A1 policy management" defined in R1GAP [5].

9.1.2 API version

For the A1 policy management API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0, and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v1".

9.1.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined in clause 5.2. The <apiName> resource URI variable shall be "a1-policy-managment". The <apiSpecificResourceUriPart> for each resource shall be set as described in clause 9.1.5.

Figure 9.1.3-1 shows the overall resource URI structure defined for the A1 policy management API for querying policy types and policies, for life cycle management of policies, and for subscriptions to A1 policy status notifications.

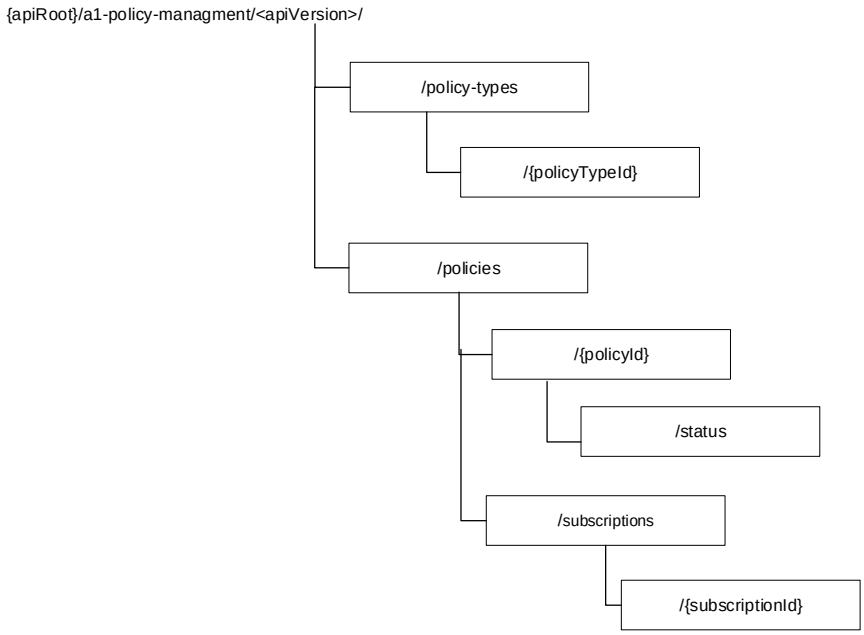

Figure 9.1.3-1: Resource URI structure of the A1 policy management API

Table 9.1.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 9.1.3-1: Resources and methods overview of the A1 policy management API

Resource name	Resource URI	HTTP method	Service Operation
All A1 policy types	…/policy-types	GET	Query A1 policy type identifiers.
Individual A1 policy type	…/policy-types/{policyTypeId}	GET	Query A1 policy type.
All A1 policies	…/policies	GET	Query A1 policy identifiers.
		POST	Create A1 policy.
Individual A1 policy	…/policies/{policyId}	GET	Query A1 policy.
		PUT	Update A1 policy.
		DELETE	Delete A1 policy.
Individual A1 policy status	…/policies/{policyId}/status	GET	Query A1 policy status.
All A1 policy status subscriptions	…/policies/subscriptions	POST	Subscribe A1 policy status.
Individual A1 policy status subscription	…/policies/subscriptions/{subscriptionId}	PUT	Update A1 policy status subscription.
		GET	Query A1 policy status subscription.
		DELETE	Unsubscribe A1 policy status.
Policy status change notifications	{notificationDestination}	POST	Notify A1 policy status changes.

9.1.4 Service operations

9.1.4.1 Query A1 policy type identifiers

9.1.4.1.1 Operation definition

The API Consumer uses this operation to query available A1 policy type identifiers.

The operation to query available A1 policy type identifiers is based on HTTP GET.

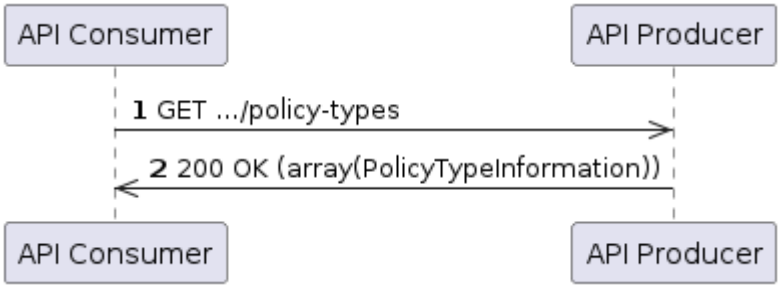

Figure 9.1.4.1.1-1: Query A1 policy type identifiers operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource "/policy-types" and optionally query parameters, the message content shall be empty. The API Producer shall process the request received in the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success, "200 OK" shall be returned. The message content shall carry an array of policy type information representing available policy types and for each policy type identifier the Near-RT RIC identifiers of those Near-RT RICs that support the related A1 policy type. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

NOTE: The behaviour of the query parameters is specified in Table 9.1.5.2.3.1-3.

9.1.4.1.2 Referenced procedures

9.1.4.1.2.1 Query A1 policy type identifiers procedure

The Query A1 policy type identifiers API operation illustrated in Figure 9.1.4.1.1-1 is based on the Query A1 policy type identifiers procedure defined in R1GAP [5].

9.1.4.2 Query A1 policy type

9.1.4.2.1 Operation definition

The API Consumer uses this operation to query an A1 policy type.

The Query A1 policy type operation is based on HTTP GET.

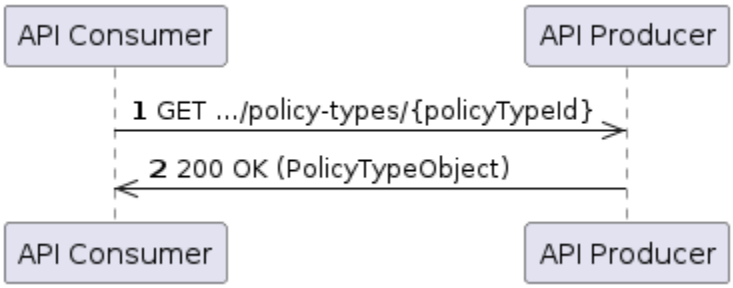

Figure 9.1.4.2.1-1: Query A1 policy type operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the policy type to be read based on the policyTypeId under the resource "/policy-types". The message content shall be empty. The API Producer shall process the request received in the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success, "200 OK" shall be returned. The message content shall carry a PolicyTypeObject representing the read policy type. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

9.1.4.2.2 Referenced procedures

9.1.4.2.2.1 Query A1 policy type procedure

The Query A1 policy type API operation illustrated in Figure 9.1.4.2.1-1 is based on the Query A1 policy type procedure defined in R1GAP [5].

9.1.4.3 Query A1 policy identifiers

9.1.4.3.1 Operation definition

The API Consumer uses this operation to query A1 policy identifiers.

The operation to query A1 policy identifiers is based on HTTP GET.

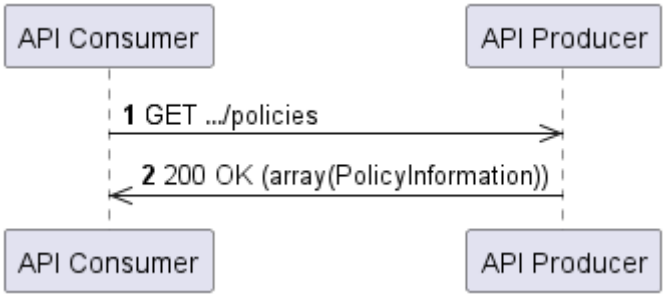

Figure 9.1.4.3.1-1: Query A1 policy identifiers operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource "/policies" and optionally query parameters. The message content shall be empty. The API Producer shall process the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success, "200 OK" shall be returned. The message content shall carry an array of policy information which includes Near-RT RIC identifiers where A1 policies exist and for each Near-RT RIC identifier the policy identifiers of those policies that exist in that Near-RT RIC. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

NOTE: The behaviour of the query parameters is specified in Table 9.1.5.2.3.1-3.

9.1.4.3.2 Referenced procedures

9.1.4.3.2.1 Query A1 policy identifiers procedure

The Query A1 policy identifiers API operation illustrated in Figure 9.1.4.3.1-1 is based on the Query A1 policy identifiers procedure defined in R1GAP [5].

9.1.4.4 Create A1 policy

9.1.4.4.1 Operation definition

The API Consumer uses this operation to create an A1 policy.

The Create A1 policy operation is based on HTTP POST.

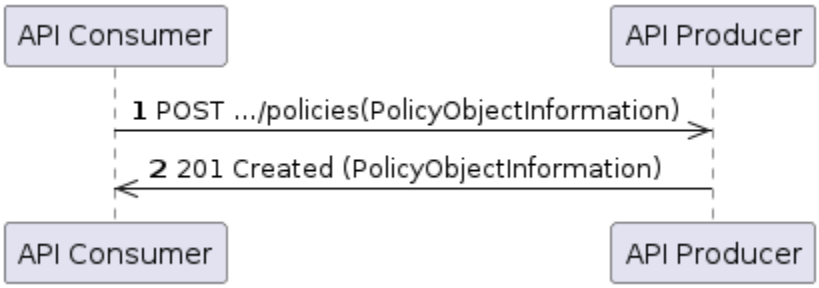

Figure 9.1.4.4.1-1: Create A1 policy operation

The service operation is as follows:

1) The API Consumer shall send an HTTP POST request to the API Producer. The target URI shall identify the resource "/policies" under which the A1 policy shall be created. The message content shall carry a PolicyObjectInformation which includes a nearRtRicId and a PolicyObject. The API Producer shall process the HTTP POST message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP POST response. On success, "201 Created" shall be returned. The message content shall carry the PolicyObjectInformation, and the "Location" HTTP header shall be present and shall carry the URI for the newly created service resource with policyId assigned by the API Producer. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

9.1.4.4.2 Referenced procedures

9.1.4.4.2.1 Create A1 policy procedure

The Create A1 policy API operation illustrated in Figure 9.1.4.4.1-1 is based on the Create A1 policy procedure defined in R1GAP [5].

9.1.4.5 Query A1 policy

9.1.4.5.1 Operation definition

The API Consumer uses this operation to query an A1 policy.

The Query A1 policy operation is based on HTTP GET.

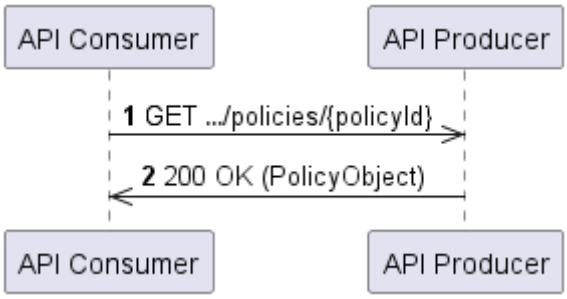

Figure 9.1.4.5.1-1: Query A1 policy operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the policy to be read based on the policyId under the resource "/policies". The message content shall be empty. The API Producer shall process the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success, "200 OK" shall be returned. The message content shall carry a PolicyObject representing the read policy. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

9.1.4.5.2 Referenced procedures

9.1.4.5.2.1 Query A1 policy procedure

The Query A1 policy API operation illustrated in Figure 9.1.4.5.1-1 is based on the Query A1 policy procedure defined in R1GAP [5].

9.1.4.6 Update A1 policy

9.1.4.6.1 Operation definition

The API Consumer uses this operation to update an A1 policy.

The Update A1 policy operation is based on HTTP PUT.

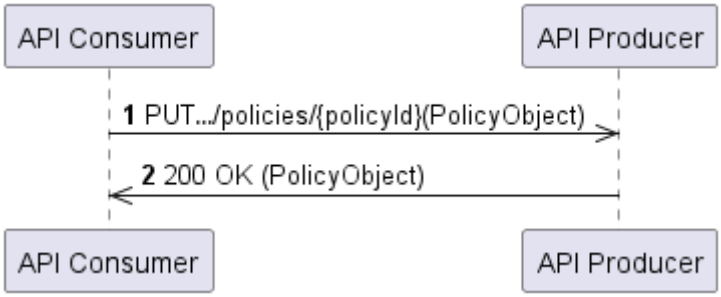

Figure 9.1.4.6.1-1: Update A1 policy operation

The service operation is as follows:

1) The API Consumer shall send an HTTP PUT request to the API Producer. The target URI shall identify the policy to be updated based on the policyId under the resource "/policies". The message content shall contain a PolicyObject. The API Producer shall process the HTTP PUT message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP PUT response. On success, "200 OK" shall be returned. The message content shall carry a PolicyObject representing the updated policy. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

9.1.4.6.2 Referenced procedures

9.1.4.6.2.1 Update A1 policy procedure

The Update A1 policy API operation illustrated in Figure 9.1.4.6.1-1 is based on the Update A1 policy procedure defined in R1GAP [5].

9.1.4.7 Delete A1 policy

9.1.4.7.1 Operation definition

The API Consumer uses this operation to delete an A1 policy.

The Delete A1 policy operation is based on HTTP GET.

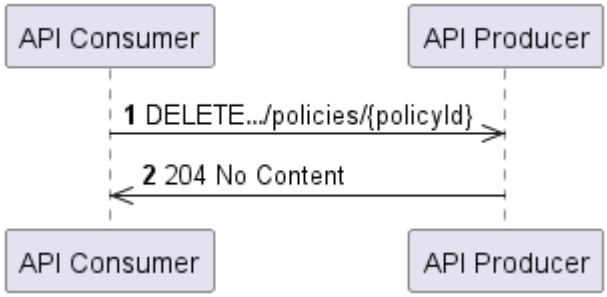

Figure 9.1.4.7.1-1: Delete A1 policy operation

The service operation is as follows:

1) The API Consumer shall send an HTTP DELETE request to the API Producer. The target URI shall identify the policy to be deleted based on the policyId under the resource "/policies". The message content shall be empty. The API Producer shall process the HTTP DELETE message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP DELETE response. On success, "204 No Content" shall be returned. The message content shall be empty. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

9.1.4.7.2 Referenced procedures

9.1.4.7.2.1 Delete A1 policy procedure

The Delete A1 policy API operation illustrated in Figure 9.1.4.7.1-1 is based on the Delete A1 policy procedure defined in R1GAP [5].

9.1.4.8 Query A1 policy status

9.1.4.8.1 Operation definition

The API Consumer uses this operation to query A1 policy status.

The operation to query A1 policy status is based on HTTP GET.

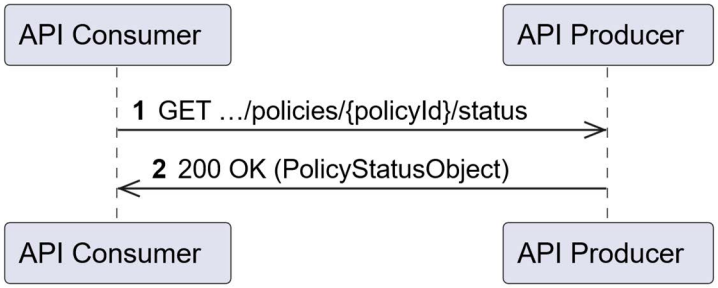

Figure 9.1.4.8.1-1: Query A1 policy status operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource "…/policies/{policyId}/status". The message content shall be empty. The API Producer shall process the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success, "200 OK" shall be returned. The message content shall carry a PolicyStatusObject representing the status of the A1 policy. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

9.1.4.8.2 Referenced procedures

9.1.4.8.1.1 Query A1 policy status procedure

The Query A1 policy status operation illustrated in Figure 9.1.4.8.1-1 is based on the Query A1 policy enforcement status procedure defined in R1GAP [5].

9.1.4.9 Subscribe A1 policy status

9.1.4.9.1 Operation definition

The API Consumer uses this operation to subscribe to notifications for status changes of A1 policies.

The operation to subscribe to A1 policy status is based on HTTP POST.

Figure 9.1.4.9.1-1: Subscribe to A1 policy status operation

The service operation is as follows:

1) The API Consumer shall send an HTTP POST request to the API Producer. The target URI shall identify the resource "/policies/subscriptions" under which the new subscription is requested to be created. The message content shall carry a PolicyStatusSubscription structure. The API Producer shall process the request received in the HTTP POST message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP POST response. On success, "201 Created" shall be returned. The location header shall be present and shall carry the URI of new subscription resource with subscriptionId assigned by the API producer. The message content shall carry a PolicyStatusSubscription structure that represents the new resource. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

9.1.4.9.2 Referenced procedures

9.1.4.9.2.1 Subscribe A1 policy status procedure

The Subscribe A1 policy status operation illustrated in Figure 9.1.4.9.1-1 is based on the Subscribe A1 policy status procedure defined in R1GAP [5].

9.1.4.10 Update A1 policy status subscription

9.1.4.10.1 Operation definition

The API Consumer uses this operation to update a subscription for A1 policy status notifications.

The operation to update an individual A1 policy status subscription is based on HTTP PUT.

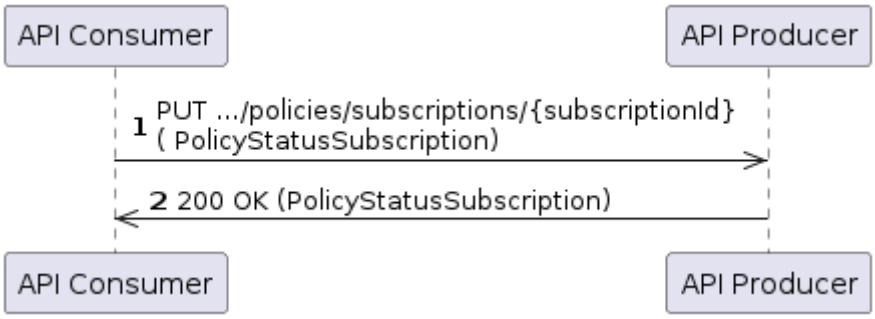

Figure 9.1.4.10.1-1: Update A1 policy status subscription operation

The service operation is as follows:

1) The API Consumer shall send an HTTP PUT request to the API Producer. The target URI shall identify the resource (…/policies/subscriptions /{subscriptionId}). The message content shall carry the updated PolicyStatusSubscription structure. The API Producer shall process the request received in the HTTP PUT message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP PUT response. On success, "200 OK" shall be returned. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

9.1.4.10.2 Referenced procedures

9.1.4.10.2.1 Update A1 policy status subscription procedure

The Update A1 policy status subscription operation illustrated in Figure 9.1.4.10.1-1 is based on the Update A1 policy status procedure defined in R1GAP [5].

9.1.4.11 Query A1 policy status subscription

9.1.4.11.1 Operation definition

The API Consumer uses this operation to query a subscription for A1 policy status.

The operation to query an individual A1 policy status subscription is based on HTTP GET.

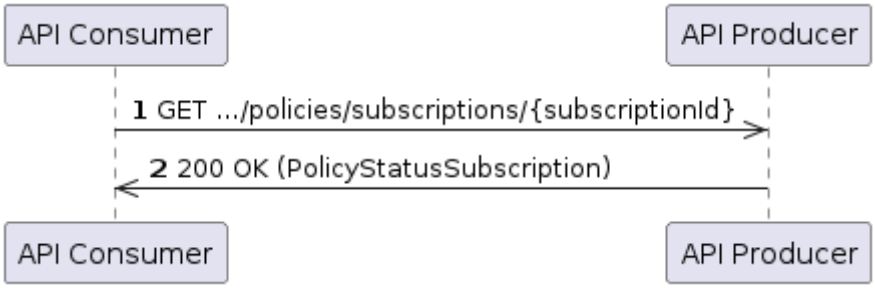

Figure 9.1.4.11.1-1: Query A1 policy status subscription operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource "/policies/subscriptions/{subscriptionId}". The message content shall be empty. The API Producer shall process the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success, "200 OK" shall be returned. The message content shall carry the PolicyStatusSubscription representing the queried policy status subscription. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

9.1.4.11.2 Referenced procedures

9.1.4.11.2.1 Query A1 policy status subscription procedure

The Query A1 policy status subscription operation illustrated in Figure 9.1.4.11.1-1 is based on the Query A1 policy status subscription procedure defined in R1GAP [5].

9.1.4.12 Unsubscribe A1 policy status

9.1.4.12.1 Operation definition

The API Consumer uses this operation to unsubscribe from notifications on status changes of A1 policies.

The operation to unsubscribe from A1 policy status notifications is based on HTTP DELETE.

Figure 9.1.4.12.1-1: Unsubscribe from A1 policy status operation

The service operation is as follows:

1) The API Consumer shall send an HTTP DELETE request to the API Producer. The target URI shall identify the resource "/policies/subscriptions/{subscriptionId}". The message content shall be empty. The API Producer shall process the request received in the HTTP DELETE message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP DELETE response. On success, "204 No Content" shall be returned. The message content shall be empty. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

9.1.4.12.2 Referenced procedures

9.1.4.12.2.1 Unsubscribe A1 policy status procedure

The Unsubscribe A1 policy status operation illustrated in Figure 9.1.4.12.1-1 is based on the Unsubscribe A1 policy status procedure defined in R1GAP [5].

9.1.4.13 Notify A1 policy status changes

9.1.4.13.1 Operation definition

The API Producer uses this operation to notify the API Consumer about status changes of an A1 policy.

The operation to notify A1 policy status is based on HTTP POST.

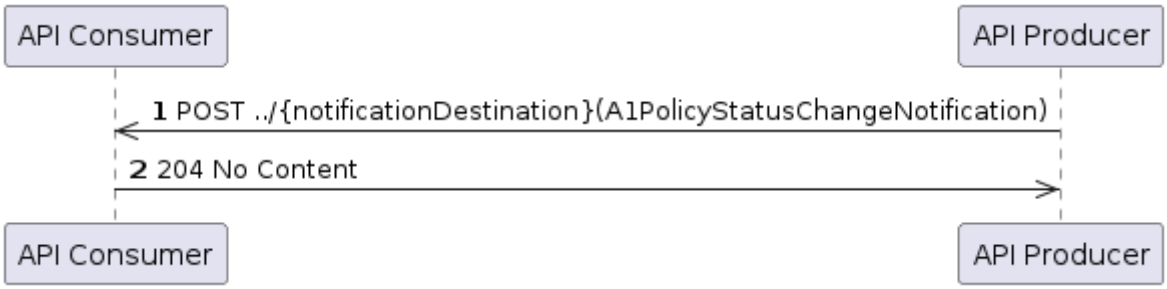

Figure 9.1.4.13.1-1: Notify A1 policy status changes API operation

The service operation is as follows:

1) The API Producer shall send an HTTP POST request to the API Consumer. The target URI (notificationDestination) identifies the sink for policy status change notifications. The message body shall contain a A1PolicyStatusChangeNotifications.

2) The API Consumer shall return the HTTP POST response with "204 No Content". The message body shall be empty. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

9.1.4.13.2 Referenced procedures

9.1.4.13.2.1 Notify A1 policy status procedure

The Notify A1 policy status changes operation illustrated in Figure 9.1.4.13.1-1 is based on the Notify A1 policy status changes procedure defined in R1GAP [5].

9.1.5 Resources

9.1.5.1 Overview

This clause defines the resources for the A1 policy management API.

9.1.5.2 Resource: "All A1 policy types"

9.1.5.2.1 Description

The resource All A1 policy types represents all A1 policy types that are available in all Near-RT RIC's over the A1 Interface.

The methods defined in clause 9.1.5.2.3 shall be supported by these resources.

9.1.5.2.2 Resource Definition

Resource URI: **{apiRoot}/a1-policy-management /<apiVersion>/policy-types**

The resource URI variables supported by the resource are defined in Table 9.1.5.2.2-1.

Table 9.1.5.2.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 9.1.2.

9.1.5.2.3 Resource Standard Methods

9.1.5.2.3.1 GET

This method shall support the URI query parameters specified in Table 9.1.5.2.3.1-1, the request data structure specified in Table 9.1.5.2.3.1-2 and the response data structures, and response code specified in Table 9.1.5.2.3.1-3.

Table 9.1.5.2.3.1-1: URI query parameters supported by the GET method on this resource

Name	Data type	P	Cardinality	Description	Applicability
nearRtRicId	string	O	0..1	The identifier of Near-RT RIC (see note).	
typeName	string	O	0..1	The unique label of the policy type (see note).	
NOTE: If multiple query parameters are provided these shall be combined with AND when evaluating the query.					

Table 9.1.5.2.3.1-2: Data structures supported by the GET request body on this resource

Data Type	P	Cardinality	Description
N/A			There is no object in the message content of the GET request.

Table 9.1.5.2.3.1-3: Data structures supported by the GET response body on this resource

Data Type	P	Cardinality	Response codes	Description
array(PolicyTypeInformation)	M	0.. N	200 OK	The operation is successful, the policy type information (see notes 1, 2 and 3).
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.
NOTE 1: If a Near-RT RIC identifier has been provided as query parameter, the response body shall contain only entries for policy types supported by the related Near-RT RIC. NOTE 2: If a policy type identifier has been provided as query parameter, the response body shall contain only entries for the related policy type. NOTE 3: If both a Near-RT RIC identifier and a policy type identifier have been provided as query parameters, the response body shall contain only entries for the related policy type supported by the related Near-RT RIC.				

9.1.5.2.4 Resource Custom Operations

None.

9.1.5.3 Resource: "Individual A1 policy type"

9.1.5.3.1 Description

The resource individual A1 policy type represents the A1 policy type that are available in the A1 policy management service.

Only the methods defined in clause 9.1.5.3.3 shall be supported by these resources.

9.1.5.3.2 Resource Definition

Resource URI: **{apiRoot}/a1-policy-management /<apiVersion>/policy-types/{policyTypeId}**

The resource URI variables supported by the resource are defined in Table 9.1.5.3.2-1.

Table 9.1.5.3.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 9.1.2.
policyTypeId	The policy type identifier as defined in A1TD [24].

9.1.5.3.3 Resource Standard Methods

9.1.5.3.3.1 GET

This method shall support the URI query parameters specified in Table 9.1.5.3.3.1-1, and the response data structure and response code specified in Table 9.1.5.3.3.1-2.

Table 9.1.5.3.3.1-1: Data structures supported by the HTTP GET request body on this resource

Data type	P	Cardinality	Description
N/A			There is no object in the message content of a GET request.

Table 9.1.5.3.3.1-2: Data structures supported by the HTTP GET response body on this resource

Data type	P	Cardinality	Response codes	Description
PolicyTypeObject	M	1	200 OK	Requested policy type object as defined in A1TD [24].
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.

9.1.5.3.4 Resource Custom Operations

None.

9.1.5.4 Resource: "All A1 policies"

9.1.5.4.1 Description

The resource All A1 policies represents the A1 policy that are available in the A1 policy management service.

Only the methods defined in clause 9.1.5.4.3 shall be supported by these resources.

9.1.5.4.2 Resource Definition

Resource URI: **{apiRoot}/a1-policy-management /<apiVersion>/policies**

The resource URI variables supported by the resource are defined in Table 9.1.5.4.2-1.

Table 9.1.5.4.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 9.1.2.

9.1.5.4.3 Resource Standard Methods

9.1.5.4.3.1 GET

This method shall support the URI query parameters specified in Table 9.1.5.4.3.1-1, the request data structure specified in Table 9.1.5.4.3.1-2 and the response data structure and response code specified in Table 9.1.5.4.3.1-3.

Table 9.1.5.4.3.1-1: URI query parameters supported by the GET method on this resource

Name	Data type	P	Cardinality	Description	Applicability
nearRtRicId	string	O	0..1	The identifier of Near-RT RIC (see note).	
policyTypeId	string	O	0..1	The identifier of the policy (see note).	
NOTE: If multiple query parameters are provided these shall be combined with AND when evaluating the query.					

Table 9.1.5.4.3.1-2: Data structures supported by the HTTP GET request body on this resource

Data type	P	Cardinality	Description
N/A			There is no object in the message content of a GET request.

Table 9.1.5.4.3.1-3: Data structures supported by the HTTP GET response body on this resource

Data type	P	Cardinality	Response codes	Description
array(PolicyInformation)	M	0..N	200 OK	The operation is successful, and the response body carries a list of policy information entries.
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.
NOTE 1: If a Near-RT RIC identifier has been provided as query parameter, the response body shall contain only entries for policies existing in the related Near-RT RIC. NOTE 2: If a policy type identifier has been provided as query parameter, the response body shall contain only entries for policies of the related policy type. NOTE 3: If both a Near-RT RIC identifier and a policy type identifier have been provided as query parameters, the response body shall contain only entries for policies of the related policy type existing in the related Near-RT RIC.				

9.1.5.4.3.2 POST

This method shall support the request data structures specified in Table 9.1.5.4.3.2-1 and the response data structures and response codes specified in Table 9.1.5.4.3.2-2 and the HTTP headers specified in Table 9.1.5.4.3.2-3.

Table 9.1.5.4.3.2-1: Data structures supported by the HTTP POST request body on this resource

Data type	P	Cardinality	Description
PolicyObjectInformation	M	1	Information related to the creation of the policy.

Table 9.1.5.4.3.2-2: Data structures supported by the HTTP POST response body on this resource

Data type	P	Cardinality	Response codes	Description
PolicyObjectInformation	M	1	201 Created	Confirmation of creation of the policy.
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.

Table 9.1.5.4.3.2-3: Headers supported by the 201 Response Code on this resource

Name	Data type	P	Cardinality	Description
Location	string	M	1	Contains the URI of the newly created "Individual registered" A1 policy resource, as defined in clause 9.1.5.3, with the policyId in the URI.

9.1.5.4.4 Resource Custom Operations

None.

9.1.5.5 Resource: "Individual A1 policy"

9.1.5.5.1 Description

The resource Individual A1 policy represents an A1 policy created by the A1 policy management service.

Only the methods defined in clause 9.1.5.5.3 shall be supported by these resources.

9.1.5.5.2 Resource Definition

Resource URI: **{apiRoot}/a1-policy-management /<apiVersion>/policies/{policyId}**

The resource URI variables supported by the resource are defined in Table 9.1.5.5.2-1.

Table 9.1.5.5.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 9.1.2.
policyId	Policy Identifier of the policy as defined in A1AP [23].

9.1.5.5.3 Resource Standard Methods

9.1.5.5.3.1 PUT

This method shall support the request data structures specified in Table 9.1.5.5.3.1-1 and the response data structure and response codes specified in Table 9.1.5.5.3.1-2.

Table 9.1.5.5.3.1-1: Data structures supported by the HTTP PUT request body on this resource

Data type	P	Cardinality	Description
PolicyObject	M	1	Update the Policy.

Table 9.1.5.5.3.1-2: Data structures supported by the HTTP PUT response body on this resource

Data type	P	Cardinality	Response codes	Description
PolicyObject	M	1	200 OK	Confirmation of updated policy.
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.

9.1.5.5.3.2 GET

This method shall support the request data structures specified in Table 9.1.5.5.3.2-1 and the response data structures and response codes specified in Table 9.1.5.5.3.2-2.

Table 9.1.5.5.3.2-1: Data structures supported by the HTTP GET request body on this resource

Data type	P	Cardinality	Description
N/A		0	There is no object in the message content of a GET request.

Table 9.1.5.5.3.2-2: Data structures supported by the HTTP GET response body on this resource

Data type	P	Cardinality	Response codes	Description
PolicyObject	M	1	200 OK	Requested policy object.
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.

9.1.5.5.3.3 DELETE

This method shall support the request data structures specified in Table 9.1.5.5.3.3-1 and the response data structures and response codes specified in Table 9.1.5.5.3.3-2.

Table 9.1.5.5.3.3-1: Data structures supported by the HTTP DELETE request body on this resource

Data type	P	Cardinality	Description
N/A			There is no object in the message content of a DELETE request.

Table 9.1.5.5.3.3-2: Data structures supported by the HTTP DELETE response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No content	Confirmation of successful deletion.
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.

9.1.5.5.4 Resource Custom Operations

None.

9.1.5.6 Resource: "Individual A1 policy status"

9.1.5.6.1 Description

The resource Individual A1 policy status represents the status of an A1 policy that is available in the A1 policy management service.

Only the methods defined in clause 9.1.5.6.3 shall be supported by these resources.

9.1.5.6.2 Resource Definition

Resource URI: **{apiRoot}/a1-policy-management /<apiVersion>/policies/{policyId}/status**

The resource URI variables supported by the resource are defined in Table 9.1.5.6.2-1.

Table 9.1.5.6.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 9.1.2.

9.1.5.6.3 Resource Standard Methods

9.1.5.6.3.1 GET

This method shall support the request data structures specified in Table 9.1.5.6.3.1-1 and the response data structures and response codes specified in Table 9.1.5.6.3.1-2.

Table 9.1.5.6.3.1-1: Data structures supported by the HTTP GET request body on this resource

Data type	P	Cardinality	Description
N/A		0	There is no object in the message content of a GET request.

Table 9.1.5.6.3.1-2: Data structures supported by the HTTP GET response body on this resource

Data type	P	Cardinality	Response codes	Description
PolicyStatusObject	M	1	200 OK	Requested policy status object as defined in A1TD [24], clause 6.4.2.
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.

9.1.5.6.4 Resource Custom Operations

None.

9.1.5.7 Resource: "All A1 policy status subscriptions"

9.1.5.7.1 Description

The resource represents the subscriptions for A1 policy status notifications.

The methods defined in clause 9.1.5.7.3 shall be supported by this resource.

9.1.5.7.2 Resource Definition

Resource URI: **{apiRoot}/a1-policy-management /<apiVersion>/policies/subscriptions**

The resource URI variables supported by the resource are defined in Table 9.1.5.7.2-1.

Table 9.1.5.7.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 9.1.2.

9.1.5.7.3 Resource Standard Methods

9.1.5.7.3.1 POST

This method shall support the request data structures specified in Table 9.1.5.7.3.1-1 and the response data structures and response codes specified in Table 9.1.5.7.3.1-2 and the HTTP headers specified in Table 9.1.5.7.3.1-3.

Table 9.1.5.7.3.1-1: Data structures supported by the HTTP POST request body on this resource

Data type	P	Cardinality	Description
PolicyStatusSubscription	M	1	Information of A1 Policy status subscription information.

Table 9.1.5.7.3.1-2: Data structures supported by the POST response body on this resource

Data Type	P	Cardinality	Response codes	Description
PolicyStatusSubscription	M	1	201 Created	The operation was successful. The message content of the POST response contains a PolicyStatusSubscriptionInfo representing the created resource.
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

Table 9.1.5.7.3.1-3: Headers supported by the 201-response code on the resource

Name	Data type	P	Cardinality	Description
Location	String	M	1	Contains the URI of the newly created "Individual A1 policy status subscription" resource as defined in clause 9.1.5.5.2 with subscriptionId in the URI.

9.1.5.7.4 Resource Custom Operations

None.

9.1.5.8 Resource: "Individual A1 policy status subscription"

9.1.5.8.1 Description

The resource represents a subscription for A1 policy status notifications.

The methods defined in clause 9.1.5.8.3 shall be supported by this resource.

9.1.5.8.2 Resource Definition

Resource URI: **{apiRoot}/a1-policy-management /<apiVersion>/policies/subscriptions/{subscriptionId}**

The resource URI variables supported by the resource are defined in Table 9.1.5.8.2-1.

Table 9.1.5.8.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 9.1.2.
subscriptionId	Identifier of subscription.

9.1.5.8.3 Resource Standard Methods

9.1.5.8.3.1 PUT

This method shall support the request data structures specified in Table 9.1.5.8.3.1-1 and the response data structure and response codes specified in Table 9.1.5.8.3.1-2.

Table 9.1.5.8.3.1-1: Data structures supported by the HTTP PUT request body on this resource

Data type	P	Cardinality	Description
PolicyStatusSubscription	M	1	Updated Policy status subscription information.

Table 9.1.5.8.3.1-2: Data structures supported by the HTTP PUT response body on this resource

Data type	P	Cardinality	Response codes	Description
PolicyStatusSubscription	M	1	200 OK	Confirmation of updated policy status subscription.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

9.1.5.8.3.2 GET

This method shall support the request data structures specified in Table 9.1.5.8.3.2-1 and the response data structures and response codes specified in Table 9.1.5.8.3.2-2.

Table 9.1.5.8.3.2-1: Data structures supported by the HTTP GET request body on this resource

Data type	P	Cardinality	Description
N/A		0	There is no object in the message content of a GET request.

Table 9.1.5.8.3.2-2: Data structures supported by the HTTP GET response body on this resource

Data type	P	Cardinality	Response codes	Description
PolicyStatusSubscription	M	1	200 OK	Requested policy status subscription information associated with the subscriptionId has been queried successfully and the response contains the PolicyStatusSubscriptionInfo as representation of the queried resource.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

9.1.5.8.3.3 DELETE

This method shall support the request data structures specified in Table 9.1.5.8.3.3-1 and the response data structures and response codes specified in Table 9.1.5.8.3.3-2.

Table 9.1.5.8.3.3-1: Data structures supported by the HTTP DELETE request body on this resource

Data type	P	Cardinality	Description
N/A			There is no object in the message content of a DELETE request.

Table 9.1.5.8.3.3-2: Data structures supported by the HTTP DELETE response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No content	Confirmation of successful deletion.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

9.1.5.8.4 Resource Custom Operations

None.

9.1.6 Custom operation without associated resources

None.

9.1.7 Notifications

9.1.7.1 Resource: Policy status change notifications

9.1.7.1.1 Description

The resource represents the destination for A1 policy status change notifications.

9.1.7.1.2 Resource Definition

The resource URI (notificationDestination) is provided when subscribing to A1 policy status notifications.

9.1.7.1.3 Resource Standard Methods

9.1.7.1.3.1 POST

This method shall support the request data structures specified in Table 9.1.7.1.3.1-1 and the response data structure and response codes specified in Table 9.1.7.1.3.1-2.

Table 9.1.7.1.3.1-1: Data structures supported by the HTTP POST request body on this resource

Data type	P	Cardinality	Description
A1PolicyStatusChange Notification	M	1	Notify policy status change as specified in clause 9.1.8.1.6.

Table 9.1.7.1.3.1-2: Data structures supported by the HTTP POST response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No content	Confirmation of received notification.

9.1.8 Data Model

9.1.8.1 Structured data types

9.1.8.1.1 Overview

The following clause defines the structured data types and their attributes to be used by the A1 policy management API.

9.1.8.1.2 Data type: PolicyTypeInformation

The PolicyTypeInformation data type represents a pair of policy type identifier and related Near-RT RIC identifier. It contains the attributes defined in Table 9.1.8.1.2-1.

Table 9.1.8.1.2-1: Definition of type PolicyTypeInformation

Attribute Name	Data type	P	Cardinality	Description
policyTypeId	string	M	1	Policy Type identifier as defined in A1AP [23], clause 6.2.3.1.3.
nearRtRicId	NearRtRicId	M	1	Near-RT RIC identifier.

The data model for the data types transported is defined in A1TD [24].

9.1.8.1.3 Data type: PolicyInformation

The PolicyInformation data type represents a pair of policy identifier and related Near-RT RIC identifier. It contains the attributes defined in Table 9.1.8.1.3-1.

Table 9.1.8.1.3-1: Definition of type PolicyInformation

Attribute Name	Data type	P	Cardinality	Description
policyId	string	M	1	Policy Identifier of a policy as defined in A1AP [23].
nearRtRicId	NearRtRicId	M	1	Near-RT RIC identifier.

9.1.8.1.4 Data type: PolicyObjectInformation

The PolicyObjectInformation data type represents a policy object, related Near-RT RIC identifier and optional policy type identifier. It contains the attributes defined in Table 9.1.8.1.4-1.

Table 9.1.8.1.4-1: Definition of type PolicyObjectInformation

Attribute Name	Data type	P	Cardinality	Description
policyObject	object	M	1	Policy Object is a JSON representation of an A1 policy; the A1 policies are specified in A1TD [24].
nearRtRicId	NearRtRicId	M	1	Near-RT RIC identifier.
policyTypeId	PolicyTypeId	O	0..1	policy type identifier as defined in A1AP [23].

9.1.8.1.5 Data type: PolicyStatusSubscription

The PolicyStatusSubscription data type represents the subscription information of A1 policy status. It contains the attributes defined in Table 9.1.8.1.5-1.

Table 9.1.8.1.5-1: Definition of type PolicyStatusSubscription

Attribute Name	Data type	P	Cardinality	Description
subscriptionScope	QueryFilter	C	1	See clause 9.1.8.3.3.1. See notes 2, 4, 6, 8 and 10.
notificationDestination	URI	M	1	Call back URI for A1 policy status notifications
policyIdList	array(policyId)	C	1..N	List of identifiers of A1 policies as defined in clause 6.2.6 of A1AP [23]. See notes 1, 3 and 4.
policyTypeIdList	array(policyTypeId)	C	1..N	List of A1 policy type identifiers as defined in clause 9.1.5.4.3.1 See notes 1, 5, 6, 9 and 10.
nearRtRicIdList	array(NearRtRicId)	C	1..N	List of Near-RT RIC identifiers as defined in clause 9.1.5.4.3.1. See notes 1, 7, 8, 9 and 10.

NOTE 1: It is conditionally optional to include either a policyIdList; or a policytypeIdList; or a nearRtRicIdList; or a policytypeIdList and a nearRtRicIdList; or none of the lists.
NOTE 2: If neither policyidList nor policytypeIdList nor nearRtRicIdList is provided, then a subscriptionScope shall be provided.
NOTE 3: If a policyIdList is provided, the subscription is for the status of the indicated A1 policies.
NOTE 4: If a policyIdList is provided then subscriptionScope shall not be provided.
NOTE 5: If a policytypeIdList is provided, the subscription is for the status of A1 policies of the indicated A1 policy types.
NOTE 6: If both policytypeIdList and subscriptionScope are provided, then the subscription is for the status of A1 policies of the indicated A1 policy types that fulfill the subscriptionScope.
NOTE 7: If a nearRtRicIdList is provided, the subscription is for the status of A1 policies created in the indicated Near-RT RICs.
NOTE 8: If both nearRtRicIdList and subscriptionScope are provided, then the subscription is for the status of A1 policies created in the indicated Near-RT RICs that fulfill the subscriptionScope.
NOTE 9: If both policytypeIdList and nearRtRicIdList are provided, then the subscription is for the status of A1 policies of the indicated A1 policy types created in the indicated Near-RT RICs.
NOTE 10: If policytypeIdList, nearRtRicIdList and subscriptionScope are provided, then the subscription is for the status of A1 policies of the indicated A1 policy types created in the indicated Near-RT RICs that fulfill the subscriptionScope.

9.1.8.1.6 Data type: A1PolicyStatusChangeNotification

Table 9.1.8.1.6-1: Definition of type A1PolicyStatusChangeNotification

Attribute Name	Data type	P	Cardinality	Description
subscriptionId	SubscriptionId	M	1	Identifier of Subscription as specified in clause 9.1.5.5.2.
policyStates	array(SubscriptionStatusObject)	M	1..N	List of policy states to be notified about.

9.1.8.1.7 Data type: SubscriptionStatusObject

Table 9.1.8.1.7-1: Definition of type SubscriptionStatusObject

Attribute Name	Data type	P	Cardinality	Description
policyId	PolicyId	M	1	Policy identifier as specified in clause 6.2.6 of A1AP [23].
policyStatusObject	PolicyStatusObject	M	1..N	Policy status object as specified in clause 6.4.2 of A1TD [24].

9.1.8.2 Simple data types and enumerations

9.1.8.2.1 Introduction

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

9.1.8.2.2 Simple data types

The resource identifiers defined in clause include policy type identifier and policy identifier based on the simple data types specified in Table 9.1.8.2.2-1.

Table 9.1.8.2.2-1: General definition of simple data types

Type Name	Type Definition	Description	Applicability
PolicyTypeId	string	policy type identifier as defined in A1TD [24].	
PolicyId	string	policy identifier of an A1 policy as defined in A1 AP [23].	
NearRtRicId	string	Near RT RIC identifier.	

9.1.8.2.3 Enumerations

9.1.8.2.3.1 Enumeration: QueryFilter

This indicates whether the request is for A1 policies created by the requesting API Consumer, for A1 policies created by other API Consumers, or for all A1 policies created by any API Consumer.

Table 9.1.8.2.3.1-1: Enumeration type of QueryFilter

Enumerations Value	Description
OWN	indicate the A1 policies created by API Consumer.
OTHERS	indicate the A1 policies created other API Consumers.
ALL	indicate the A1 policies created by any API Consumers.

9.1.8.3 Re-used data types

None.

9.1.8.4 Service-specific registration information

None.

9.1.9 Error Handling

9.1.9.1 General

In addition to the general provisions in clause 5.4.3, the requirements in the following clauses are applicable for the A1 policy management API.

9.1.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

9.1.9.3 Application Errors

The application errors defined for the A1 policy management service are listed in Table 9.1.9.3-1.

Table 9.1.9.3-1: Application errors

Application Error	HTTP status code	Description
Unauthorized	401	Used when the API consumer lacks proper authentication credentials or has provided invalid credentials.
Forbidden	403	Used when the API Consumer has successfully authenticated the user, but the user is still denied access to the requested resource.
Bad Request	400	Used when the A1 policy management service cannot or will not process a request, e.g. when the validation of PolicyObject towards a policy type schema, or the validation of PolicyStatusObject towards a policy status schema, fails.
Not Found	404	Used when the Near-RT RIC did not find a current representation for the resource representing a policy type or a policy, e.g. for a policy type that is not available or a policy that does not exist.
Method Not Allowed	405	Used when the HTTP method is not supported by the resource defined for the A1 policy management API.
Conflict	409	Used if detecting that a policy requested to be created or updated may be overlapping or conflicting with a policy that exists in the Near-RT RIC.

Application errors should be mapped to the most applicable 4xx/5xx HTTP error status code. If no such status code is applicable, one of the status codes 400 (Bad Request) or 500 (Internal Server Error) should be used.

The HTTP status codes listed in Table 9.1.9.3-1 shall be used as defined in clause 5.4.3 for the A1 policy management procedures and clause 9.1.5 for the resources.

Implementations may use additional HTTP error status codes in addition to those listed in Table 9.1.9.3-1, as long as they are valid HTTP status codes.

A list of all valid HTTP status codes and their specification documents can be obtained from the HTTP status code registry [21].

In addition, the response body may contain a JSON representation of a "ProblemDetails" data structure in the payload body as defined in clause 9.1.8.2.2. In that case, as defined by IETF RFC 7807 [10], the "Content-Type" HTTP header shall be set to "application/problem+json".

10 AI/ML workflow services

10.1 AI/ML model registration API

10.1.1 Introduction

This API enables the API Consumer to register, query, update and deregister an AI/ML model based on the AI/Ml model registration service defined in R1GAP [5].

10.1.2 API version

For the AI/ML model registration API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1]for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v1".

10.1.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined in clause 5.2. The <apiName> resource URI variable shall be "ai-ml-model-registration". The <apiSpecificResourceUriPart> for each resource shall be set as described in clause 10.1.5.

Figure 10.1.3-1 shows the overall resource URI structure defined for the model registration API.

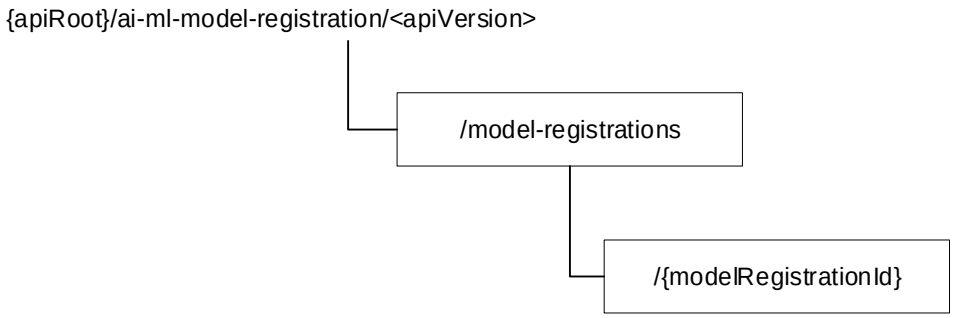

Figure 10.1.3-1: Resource URI structure of the AI/ML model registration API

Table 10.1.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 10.1.3-1: Resource and methods overview of the AI/ML model registration API

Resource name	Resource URI	HTTP method	Service Operation
Registered model registrations	…/ model-registrations	POST	Register model information.
Individual registered model registration	…/ model-registrations/{modelRegistrationId}	GET	Query model registration information.
		PUT	Update model registration information.
		DELETE	Deregister model registration information.

10.1.4 Service operations

10.1.4.1 Register model information

10.1.4.1.1 Operation definition

The API Consumer uses this operation to register AI/ML model information.

The operation to register the model information is based on HTTP POST.

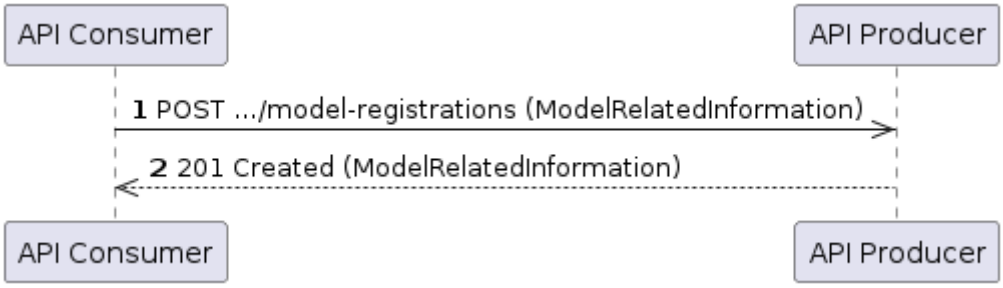

Figure 10.1.4.1.1-1: Register model information operation

The service operation is as follows:

1) The API Consumer shall send an HTTP POST request to the API Producer. The target URI shall identify the resource (…/model-registrations) under which the new registration is requested to be created. The message content shall carry a ModelRelatedInformation structure.

2) The API Producer shall generate the model registration identifier and construct the URI for the created resource. The API Producer shall return the HTTP POST response. On success, "201 Created" shall be returned. The "Location" header shall be present and shall carry the URI of the new registration resource. The message content shall carry a ModelRelatedInformation structure that represents the new resource. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

10.1.4.1.2 Referenced procedures

10.1.4.1.2.1 Register AI/ML model procedure

The Register model information operation illustrated in Figure 10.1.4.1.1-1 is based on the Register AI/ML model procedure defined for the AI/ML workflow services in R1GAP [5].

10.1.4.2 Deregister model information

10.1.4.2.1 Operation definition

The API Consumer uses this operation to delete the registered model information.

The operation to deregister an AI/ML model information that was previously registered is based on HTTP DELETE.

Figure 10.1.4.2.1-1: Deregister model information operation

The service operation is as follows:

1) The API Consumer shall send an HTTP DELETE request to the API Producer. The target URI shall identify the resource to be deleted (…/model-registrations/{modelRegistrationId}).

2) The API Producer shall return the HTTP DELETE response. On success, "204 No Content" shall be returned and the response message content shall be empty. On failure, the appropriate error code shall be returned, and the message response content may contain additional error information.

10.1.4.2.2 Referenced procedures

10.1.4.2.2.1 Deregister AI/ML Model procedure

The Deregister model information operation illustrated in Figure 10.1.4.2.1-1 is based on the Deregister AI/ML model procedure defined for the AI/ML workflow services in R1GAP [5].

10.1.4.3 Update model information

10.1.4.3.1 Operation definition

The API Consumer uses this operation to Update model information.

The operation to update the model information is based on HTTP PUT.

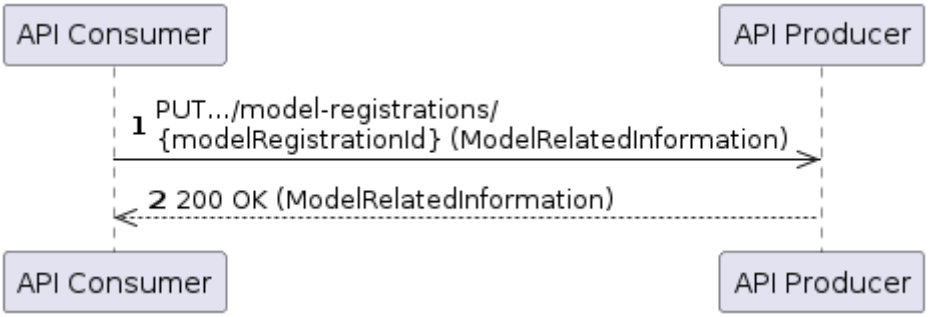

Figure 10.1.4.3.1-1: Update model information operation

The service operation is as follows:

1) The API Consumer shall send an HTTP PUT request to the API Producer. The target URI shall identify the resource (…/model-registrations/{modelRegistrationId}). The message content shall carry an updated ModelRelatedInformation structure. The API producer shall process the HTTP PUT message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP PUT response. On success, "200 OK" shall be returned. The message body shall contain updated ModelRelatedInformation structure. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

10.1.4.3.2 Referenced procedures

10.1.4.3.2.1 Update AI/ML model registration procedure

The Update model information operation illustrated in Figure 10.1.4.1.1-1 is based on the Update AI/ML model registration procedure defined for the AI/ML workflow services in R1GAP [5].

10.1.4.4 Query model information

10.1.4.4.1 Operation definition

The API Consumer uses this operation to query model information that it has previously registered.

The operation to query model information is based on HTTP GET.

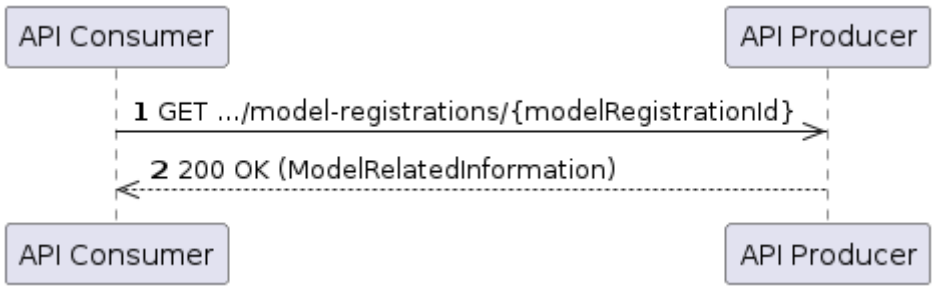

Figure 10.1.4.4.1-1: Query model information operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource (…/model-registrations/{modelRegistrationId}). The message content shall be empty. The API producer shall process the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success, "200 OK" shall be returned. The message content shall carry the queried model related information. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

10.1.4.4.2 Referenced procedures

10.1.4.4.2.1 Query AI/ML model registration procedure

The Query model information operation illustrated in Figure 10.1.4.4.1-1 is based on the Query AI/ML model registration procedure defined for the AI/ML workflow services in R1GAP [5].

10.1.5 Resources

10.1.5.1 Overview

The following clause defines the resources for the AI/ML model registration API.

10.1.5.2 Resource: "Registered model registrations"

10.1.5.2.1 Description

The resource represents the model information of an rApp that it wants to register.

Only the methods defined in clause 10.1.5.2.3 shall be supported by this resource.

10.1.5.2.2 Resource Definition

Resource URI: **{apiRoot}/ai-ml-model-registration/<apiVersion>/model-registrations**

The resource URI variables supported by the resource is defined in Table 10.1.5.2.2-1.

Table 10.1.5.2.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 10.1.2.

10.1.5.2.3 Resource Standard Methods

10.1.5.2.3.1 POST

This method shall support the request data structure specified in Table 10.1.5.2.3.1-1 and the response data structure and response code specified in Table 10.1.5.2.3.1-2, and the HTTP headers specified in Table 10.1.5.2.3.1-3.

Table 10.1.5.2.3.1-1: Data structures supported by the POST request body on this resource

Data Type	P	Cardinality	Description
ModelRelatedInformation	M	1	Information related to the model

Table 10.1.5.2.3.1-2: Data structures supported by the POST response body on this resource

Data Type	P	Cardinality	Response codes	Description
ModelRelatedInformation	M	1	201 Created	The operation was successful, and the message content of the POST response contains a ModelRelatedInformation structure as a representation of the created resource.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

Table 10.1.5.2.3.1-3: Headers supported by the 201 Response Code on this resource

Name	Data type	P	Cardinality	Description
Location	string	M	1	Contains the URI of the newly created "Individual registered ModelInformation" resource, as defined in clause 10.1.5.3, with the registrationId in the URI.

10.1.5.3 Resource: "Individual registered model registration"

10.1.5.3.1 Description

The resource represents the model information of an rApp that it wants to update, deregister, and query.

Only the methods defined in clause 10.1.5.3.3 shall be supported by this resource.

10.1.5.3.2 Resource Definition

Resource URI: **{apiRoot}/ai-ml-model-registration/<apiVersion>/model-registrations/{modelRegistrationId}**

The resource URI variables supported by the resource are defined in Table 10.1.5.3.2-1.

Table 10.1.5.3.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 10.1.2.
modelRegistrationId	The registration identifier of the model.

10.1.5.3.3 Resource Standard Methods

10.1.5.3.3.1 DELETE

This method shall support the request data structure specified in Table 10.1.5.3.3.1-1 and the response data structure and response code specified in Table 10.1.5.3.3.1-2.

Table 10.1.5.3.3.1-1: Data structure supported by the DELETE request body on this resource

Data type	P	Cardinality	Description
N/A			A DELETE request has no message content.

Table 10.1.5.3.3.1-2: Data structure supported by the DELETE response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No Content	The AI/ML model registration associated with the modelRegistrationId has been deleted successfully. The message content shall be empty.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

10.1.5.3.3.2 PUT

This method shall support the request data structure specified in Table 10.1.5.3.3.2-1 and the response data structure and response code specified in Table 10.1.5.3.3.2-2.

Table 10.1.5.3.3.2-1: Data structures supported by the PUT request body on this resource

Data Type	P	Cardinality	Description
ModelRelatedInformation	M	1	Updated model related information.

Table 10.1.5.3.3.2-2: Data structures supported by the PUT response body on this resource

Data Type	P	Cardinality	Response codes	Description
ModelRelatedInformation	M	1	200 OK	The model related information associated with the modelRegistrationId has been updated successfully and the response contain the ModelRelatedInformation as a representation of the updated resource.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

10.1.5.3.3.3 GET

This method shall support the request data structure specified in Table 10.1.5.3.3.3-1 and the response data structure and response code specified in Table 10.1.5.3.3.3-2.

Table 10.1.5.3.3.3-1: Data structures supported by the GET request body on this resource

Data Type	P	Cardinality	Description
N/A			A GET request has no message content.

Table 10.1.5.3.3.3-2: Data structures supported by the GET response body on this resource

Data Type	P	Cardinality	Response codes	Description
ModelRelatedInformation	M	1	200 OK	The model related information registration associated with the modelRegistrationId has been queried successfully and the response contain the ModelRelatedInformation as a representation of the query resource.
ProblemDetails	O	0..1	4xx/5xx	The operation has failed, and the message content may contain Problem description details.

10.1.5.3.4 Resource Custom Operations

None.

10.1.6 Custom operation without associated resources

None.

10.1.7 Notifications

None.

10.1.8 Data Model

10.1.8.1 Structured data types

10.1.8.1.1 Overview

The following clauses define the structured data types and their attributes to be used by the AI/ML model registration API.

10.1.8.1.2 Data type: ModelRelatedInformation

The ModelRelatedInformation data type represents registration information for an AI/ML model. It contains the attributes defined in Table 10.1.8.1.2-1.

Table 10.1.8.1.2-1: Definition of type ModelRelatedInformation

Attribute Name	Data type	P	Cardinality	Description
modelId	ModelId	M	1	Identifier of a model.
description	string	M	1	Description of the AIML model.
modelInformation	ModelInformation	M	1	Information of the AIML model.
modelLocation	URI	O	0..1	Location of the model stored in the runtime catalogue that can be discovered and referred to when using AI/ML workflow services.

10.1.8.1.3 Data type: ModelId

The ModelId data type represents information of AI/ML model. It contains the attributes defined in Table 10.1.8.1.3-1.

Table 10.1.8.1.3-1: Definition of type ModelId

Attribute Name	Data type	P	Cardinality	Description
modelName	String	M	1	Name of the model as specified in R1GAP[5] .
modelVersion	String	M	1	Version of the model as specified in R1GAP [5].
artifactVersion	String	O	0..1	Artifact version of AIML model as specified in R1GAP [5].

10.1.8.1.4 Data type: ModelInformation

The Model information data type contains the attributes defined in Table 10.1.8.1.4-1.

Table 10.1.8.1.4-1: Definition of type ModelInformation

Attribute Name	Data type	P	Cardinality	Description
metadata	MetaData	M	1	Metadata of the model.
inputDataType	array(dataTypeId)	M	1..N	Input data type for the model, the structure of dataTypeId is specified in clause 7.1.8.
outputDataType	array(dataTypeId)	M	1..N	Output data type for the model, the structure of dataTypeId is specified in clause 7.1.8.
targetEnvironment	array(TargetEnvironment)	O	0..N	Information on the target environment is required for deployment of an AI/ML model.

10.1.8.1.5 Data type: MetaData

The Metadata data type contains the attributes defined in Table 10.1.8.1.5-1.

Table 10.1.8.1.5-1: Definition of type MetaData

Attribute Name	Data type	P	Cardinality	Description
author	String	M	1	Author of the AIML model.
owner	String	O	0..1	Ownership of the AIML model to regulate how the model can be used in the Run-Time environment.

10.1.8.1.6 TargetEnvironment

The TargetEnvironment data type contains the attributes defined in Table 10.1.8.1.6-1.

Table 10.1.8.1.6-1: Definition of type TargetEnvironment

Attribute Name	Data type	P	Cardinality	Description
platformName	String	M	1	Name of the platform.
environmentType	String	M	1	Name of the platform Execution service type, and this is dependent on the platformName.
dependencyList	URI	M	1	Location to the template that has all the list of dependencies platform should provide needs to be installed for the model. (for example, scikit-learn 0.21.3).

10.1.8.2 Simple data types and enumerations

10.1.8.2.1 Enumerations

For this AI/ML model registration API, no enumerations are defined in the present document.

10.1.9 Error Handling

10.1.9.1 General

In addition to the general provisions in clause 5.4.3, the requirements in the following clauses are applicable for the AI/ML model Registration API.

10.1.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

10.1.9.3 Application Errors

No additional application errors defined in the present document.

10.2 AI/ML model discovery API

10.2.1 Introduction

This API enables the API Consumer to discover an AI/ML model based on the AI/ML model discovery service defined in R1GAP [5].

10.2.2 API version

For the AI/ML model discovery API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v1".

10.2.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined in clause 5.2. The <apiName> resource URI variable shall be "ai-ml-model-discovery".

Figure 10.2.3-1 shows the overall resource URI structure defined for the AI/ML model discovery API.

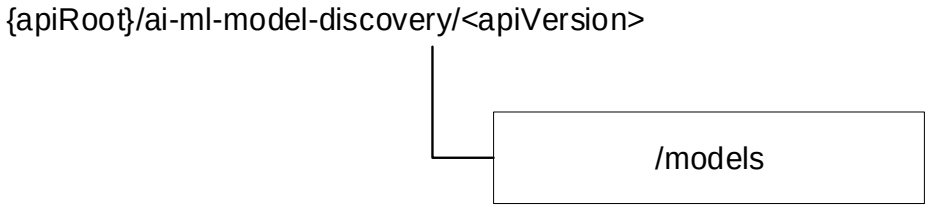

Figure 10.2.3-1: Resource URI structure of the AI/ML model discovery API

Table 10.2.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 10.2.3-1: Resource and methods overview of the AI/ML model discovery API

Resource name	Resource URI	HTTP method	Service Operation
ALL registered models	…/models	GET	Discover registered AI/ML models.

10.2.4 Service operations

10.2.4.1 Discover AI/ML models

10.2.4.1.1 Operation definition

The API Consumer uses this operation to discover the registered AI/ML models.

The operation to discover the AI/ML models is based on HTTP GET.

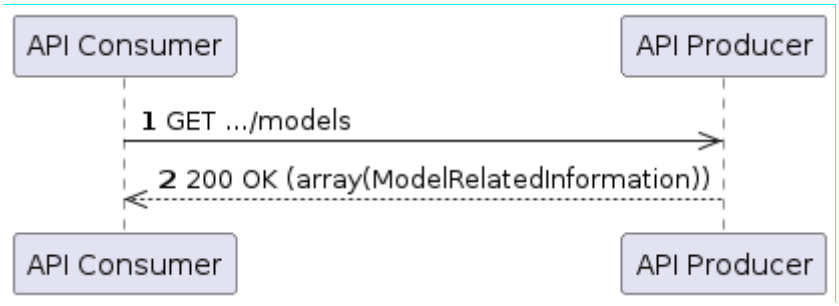

Figure 10.2.4.1-1: Discover models operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource (…/models) and may also contain query parameters to discover the registered model identifiers. The API Producer shall process the request received in the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success, "200 OK" shall be returned and the message content shall carry an array of ModelRelatedInformation that shall include modelId and corresponding metadata. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

10.2.4.1.2 Referenced procedures

10.2.4.1.2.1 Discover AI/ML model procedure

The Discover AI/ML models operation illustrated in Figure 10.2.4.1-1 is based on the Discover AI/ML model procedure defined for the AI/ML workflow service in R1GAP [5].

10.2.5 Resources

10.2.5.1 Overview

This clause defines the resource for the AI/ML model discovery API.

10.2.5.2 Resource: "All registered models"

10.2.5.2.1 Description

The resource represents all registered AI/ML models in the Non-RT RIC.

The methods defined in clause 10.2.5.2.3 shall be supported by this resource.

10.2.5.2.2 Resource Definition

Resource URI: **{apiRoot}/ai-ml-model-discovery/<apiVersion>/models**

The resource URI variables supported by the resource is defined in Table 10.2.5.2.2-1.

Table 10.2.5.2.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 10.2.2.

10.2.5.2.3 Resource Standard Methods

10.2.5.2.3.1 GET

This method shall support the URI query parameters specified in Table 10.2.5.2.3.1-1, the request data structure specified in Table 10.2.5.2.3.1-2 and the response data structures, and response code specified in Table 10.2.5.2.3.1-3.

Table 10.2.5.2.3.1-1: URI query parameters supported by the GET method on this resource

Name	Data type	P	Cardinality	Description	Applicability
model-name	string	O	0..1	Name of the model as specified in R1GAP [5] (see note).	
model-version	string	O	0..1	Version of the model as specified in R1GAP [5] (see note).	
NOTE: If multiple query parameters are provided these shall be combined with AND when evaluating the query.					

Table 10.2.5.2.3.1-2: Data structures supported by the GET request body on this resource

Data Type	P	Cardinality	Description
N/A			There is no object in the message content of the GET request.

Table 10.2.5.2.3.1-3: Data structures supported by the HTTP GET response body on this resource

Data type	P	Cardinality	Response codes	Description
array (ModelRelatedInformation)	O	0..N	200 OK	The message content of the GET response carries an array of model identifiers and metadata of the models.
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.

10.2.5.2.4 Resource Custom Methods

None.

10.2.6 Custom operation without associated resources

None.

10.2.7 Notifications

NOTE: No notifications are specified in the current version of the present document.

10.2.8 Data Model

10.2.8.1 Structured data types

10.2.8.1.1 Overview

The following clause defines the structured data types and their attributes to be used by the service API.

10.2.8.1.2 Data type: ModelRelatedInformation

The ModelRelatedInformation data type represents registration information for an AI/ML model. It contains the attributes defined in Table 10.2.8.1.2-1.

Table 10.2.8.1.2-1: Definition of type ModelRelatedInformation

Attribute Name	Data type	P	Cardinality	Description
modelId	ModelId	M	1	Identifier of a model as specified in clause 10.1.8.1.3.
metadata	string	M	1	Description of the AIML model that includes the AI/ML model related information and training related information.

10.2.8.2 Simple data types and enumerations

10.2.8.2.1 Introduction

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

10.2.8.2.2 Simple data types

For this service API, no simple data types are defined in the present document.

10.2.8.2.3 Enumerations

For this service API, no enumerations are defined in the present document.

10.2.9 Error Handling

10.2.9.1 General

In addition to the general provisions in clause 5.4.3, the requirements in the following clauses are applicable for the AI/ML model discovery API.

10.2.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

10.2.9.3 Application Errors

No additional application errors defined in the present document.

10.3 AI/ML model training API

10.3.1 Introduction

This API enables the API Consumer to request AI/ML model training, query AI/ML model training job status, and cancel AI/ML model training based on the procedures for "AI/ML model training services" as defined in R1GAP [5]. It also allows the API Producer to notify AI/ML model training job status change based on the procedure for "AI/ML model training services" defined in R1GAP [5].

10.3.2 API version

For the AI/ML model training API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0, and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v1".

The AI/ML model training API is under development and consequently the API version shall include the pre-release version "alpha.1".

10.3.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined in clause 5.2. The <apiName> resource URI variable shall be "ai-ml-model-training".

Figure 10.3.3-1 shows the overall resource URI structure defined for the AI/ML model training API.

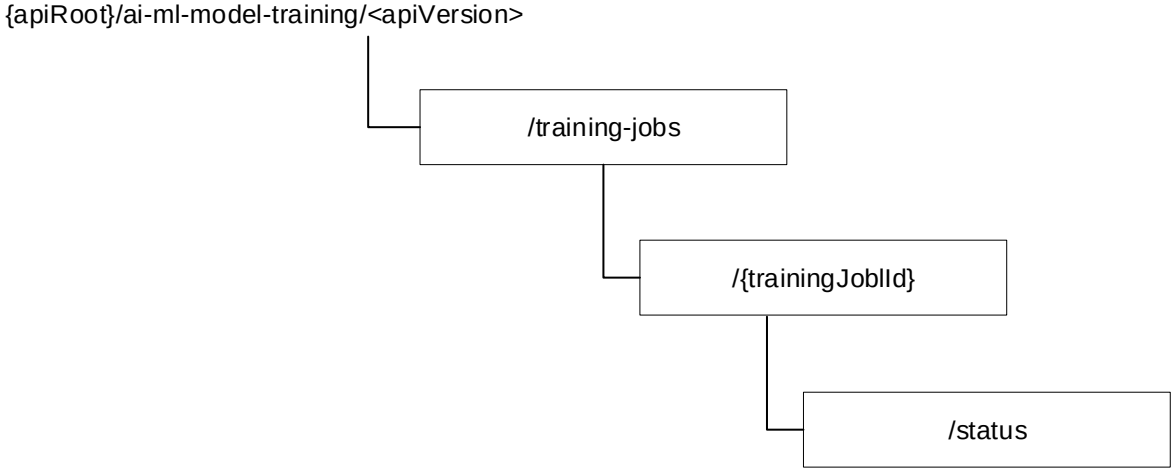

Figure 10.3.3-1: Resource URI structure of the AI/ML model training API

Table 10.3.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 10.3.3-1: Resource and methods overview of the AI/ML model training API

Resource name	Resource URI	HTTP method	Service Operation
All AI/ML model training jobs	…/training-jobs	POST	Request AI/ML model training.
Individual AI/ML model training job	…/training-jobs/{trainingJobId}	DELETE	Cancel AI/ML model training job.
Individual AI/ML model training job status	…/training-jobs/{trainingJobId}/status	GET	Query AI/ML model training job status.

10.3.4 Service operations

10.3.4.1 Request AI/ML model training

10.3.4.1.1 Operation definition

The API Consumer uses this operation to request AI/ML model training.

The operation to create AI/ML model training job is based on HTTP POST.

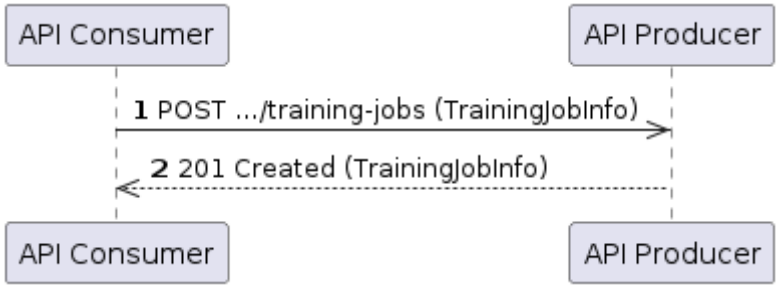

Figure 10.3.4.1.1-1: Request AI/ML model training operation

The service operation is as follows:

1) The API Consumer shall send an HTTP POST request to the API Producer. The target URI shall identify the resource "/training-jobs", the message content shall carry a TrainingJobInfo structure which includes information for training. The API Producer shall process the request received in the HTTP POST message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall generate the training job identifier and construct the URI for the created resource. The API Producer shall return the HTTP POST response. On success, "201 Created" shall be returned. The "Location" HTTP header shall be present and shall carry the URI for the newly created resource. The message content shall carry a TrainingJobInfo structure represents the new resource. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

10.3.4.1.2 Referenced procedures

10.3.4.1.2.1 Request AI/ML model training procedure

The request AI/ML model training operation illustrated in Figure 10.3.4.1.1-1 is based on the request AI/ML model training procedure defined in R1GAP [5].

10.3.4.2 Cancel AI/ML model training

10.3.4.2.1 Operation definition

The API Consumer uses this operation to cancel AI/ML model training.

The operation to delete AI/ML model training job is based on HTTP DELETE.

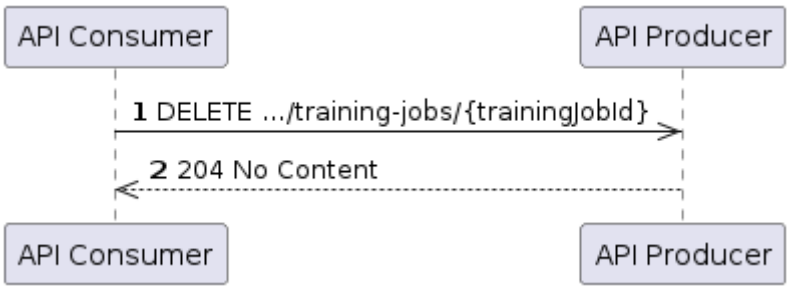

Figure 10.3.4.2.1-1: Cancel AI/ML model training operation

The service operation is as follows:

1) The API Consumer shall send an HTTP DELETE request to the API Producer. The target URI shall identify the training job resource to be deleted as "/training-jobs/{trainingJobId}", the message content shall be empty. The API Producer shall process the request received in the HTTP DELETE message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP DELETE response. On success, "204 No Content" shall be returned. The message content shall be empty. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

10.3.4.2.2 Referenced procedures

10.3.4.2.2.1 Cancel AI/ML model training procedure

The cancel AI/ML model training operation illustrated in Figure 10.3.4.2.1-1 is based on the cancel AI/ML model training procedure defined in R1GAP [5].

10.3.4.3 Query AI/ML model training job status

10.3.4.3.1 Operation definition

The API Consumer uses this operation to query the AI/ML model training job status.

The operation to query AI/ML model training job status is based on HTTP GET.

Figure 10.3.4.3.1-1: Query AI/ML model training job status operation

The service operation is as follows:

1) The API Consumer shall send an HTTP GET request to the API Producer. The target URI shall identify the resource "/training-jobs/{trainingJobId}/status", the message content shall be empty. The API Producer shall process the request received in the HTTP GET message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall return the HTTP GET response. On success, "200 OK" shall be returned. The message content shall carry a TrainingJobStatus representing the status of the training job, which is identified by the trainingJobId. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

10.3.4.3.2 Referenced procedures

10.3.4.3.2.1 Query AI/ML model training job status procedure

The query AI/ML model training job status operation illustrated in Figure 10.3.4.3.1-1 is based on the query AI/ML model training job status procedure defined in R1GAP [5].

10.3.4.4 Notify AI/ML model training job status change

10.3.4.4.1 Operation definition

The API Producer uses this operation to notify the status change of an AI/ML model training job.

The operation to notify AI/ML model training job status change is based on HTTP POST.

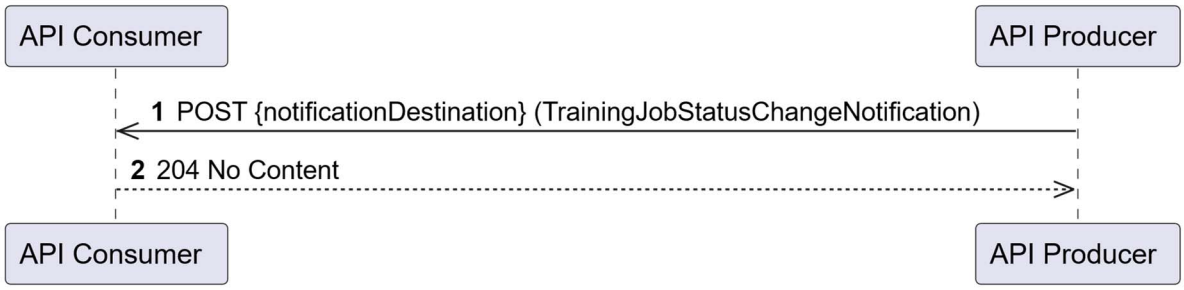

Figure 10.3.4.4.1-1: Notify AI/ML model training job status change operation

The service operation is as follows:

1) The API Producer shall send an HTTP POST request to the API Consumer. The target URI {notificationDestination} shall be the one provided by API Consumer during the creation of the training job, the message content shall carry a TrainingJobStatusChangeNotification which includes the updated training job status. The API Consumer shall process the request received in the HTTP POST message and determine if the request sent by the API Producer is authorized or not.

2) The API Consumer shall return the HTTP POST response. On success, "204 No Content" shall be returned. The message content shall be empty. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

10.3.4.4.2 Referenced procedures

10.3.4.4.2.1 Notify AI/ML model training job status change procedure

The notify AI/ML model training job status change operation illustrated in Figure 10.3.4.4.1-1 is based on the notify AI/ML model training job status change procedure defined in R1GAP [5].

10.3.5 Resources

10.3.5.1 Overview

This clause defines the resource for the AI/ML model training API.

10.3.5.2 Resource: "All AI/ML model training jobs"

10.3.5.2.1 Description

The resource represents all AI/ML model training jobs created in the AI/ML model training API producer.

The methods defined in clause 10.3.5.2.3 shall be supported by this resource.

10.3.5.2.2 Resource Definition

Resource URI: **{apiRoot}/ai-ml-model-training/<apiVersion>/training-jobs**

The resource URI variables supported by the resource is defined in Table 10.3.5.2.2-1.

Table 10.3.5.2.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 10.3.2.

10.3.5.2.3 Resource Standard Methods

10.3.5.2.3.1 POST

This method shall support the request data structures specified in Table 10.3.5.2.3.1-1 and the response data structures, and response codes specified in Table 10.3.5.2.3.1-2 and the HTTP headers specified in Table 10.3.5.2.3.1-3.

Table 10.3.5.2.3.1-1: Data structures supported by the HTTP POST request body on this resource

Data type	P	Cardinality	Description
TrainingJobInfo	M	1	Information related to the creation of the AI/ML model training job.

Table 10.3.5.2.3.1-2: Data structures supported by the HTTP POST response body on this resource

Data type	P	Cardinality	Response codes	Description
TrainingJobInfo	M	1	201 Created	Confirmation of creation of the AI/ML model training job.
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.

Table 10.3.5.2.3.1-3: Headers supported by the 201 Response Code on this resource

Name	Data type	P	Cardinality	Description
Location	string	M	1	Contains the URI of the newly created individual AI/ML model training job resource, as defined in clause 10.3.5.3, with the trainingJobId in the URI.

10.3.5.2.4 Resource Custom Methods

None.

10.3.5.3 Resource: "Individual AI/ML model training job"

10.3.5.3.1 Description

The resource represents an individual AI/ML model training job created in the AI/ML model training API producer.

The methods defined in clause 10.3.5.3.3 shall be supported by this resource.

10.3.5.3.2 Resource Definition

Resource URI: **{apiRoot}/ai-ml-model-training/<apiVersion>/training-jobs/{trainingJobId}**

The resource URI variables supported by the resource is defined in Table 10.3.5.3.2-1.

Table 10.3.5.3.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 10.3.2.
trainingJobId	The training job identifier assigned by the Service Producer.

10.3.5.3.3 Resource Standard Methods

10.3.5.3.3.1 DELETE

This method shall support the request data structures specified in Table 10.3.5.3.3.1-1 and the response data structures, and response codes specified in Table 10.3.5.3.3.1-2.

Table 10.3.5.3.3.1-1: Data structures supported by the HTTP DELETE request body on this resource

Data type	P	Cardinality	Description
N/A			There is no object in the message content of a DELETE request.

Table 10.3.5.3.3.1-2: Data structures supported by the HTTP DELETE response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No content	Confirmation of successful deletion.
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.

10.3.5.3.4 Resource Custom Methods

None.

10.3.5.4 Resource: "Individual AI/ML model training job status"

10.3.5.4.1 Description

The resource represents the status of an individual AI/ML model training job created in the Non-RT RIC.

The methods defined in clause 10.3.5.4.3 shall be supported by this resource.

10.3.5.4.2 Resource Definition

Resource URI: **{apiRoot}/ai-ml-model-training/<apiVersion>/training-jobs/{trainingJobId}/status**

The resource URI variables supported by the resource is defined in Table 10.3.5.4.2-1.

Table 10.3.5.4.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 10.3.2.
trainingJobId	The training job identifier assigned by the Service Producer.

10.3.5.4.3 Resource Standard Methods

10.3.5.4.3.1 GET

This method shall support the request data structures specified in Table 10.3.5.4.3.1-1 and the response data structures, and response codes specified in Table 10.3.5.4.3.1-2.

Table 10.3.5.4.3.1-1: Data structures supported by the HTTP GET request body on this resource

Data type	P	Cardinality	Description
N/A			There is no object in the message content of a GET request.

Table 10.3.5.4.3.1-2: Data structures supported by the HTTP GET response body on this resource

Data type	P	Cardinality	Response codes	Description
TrainingJobStatus	M	1	200 OK	The status of the AI/ML model training job.
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.

10.3.5.4.4 Resource Custom Methods

None.

10.3.6 Custom operation without associated resources

None.

10.3.7 Notifications

10.3.7.1 Notify training job status change

10.3.7.1.1 Description

The notification informs the receiver about the updated status of an AI/ML model training job.

10.3.7.1.2 Resource Definition

The Resource URI {notificationDestination} is a callback URI provided when creating an AI/ML model training job.

10.3.7.1.3 Resource Standard Methods

10.3.7.1.3.1 POST

This method shall support the request data structures specified in Table 10.3.7.1.3.1-1 and the response data structure and response codes specified in Table 10.3.7.1.3.1-2.

Table 10.3.7.1.3.1-1: Data structures supported by the HTTP POST request body on this resource

Data type	P	Cardinality	Description
TrainingJobStatusChangeNotification	M	1	Notify a status changes of an AI/ML model training job.

Table 10.3.7.1.3.1-2: Data structures supported by the HTTP POST response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No content	Confirmation of received notification.
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content.

10.3.8 Data Model

10.3.8.1 Structured data types

10.3.8.1.1 Overview

The following clauses define the data types and attributes to be used by the AI/ML model training API.

10.3.8.1.2 Data type: TrainingJobInfo

Table 10.3.8.1.2-1: Definition of type TrainingJobInfo

Attribute Name	Data type	P	Cardinality	Description
modelId	ModelId	M	1	The model identifier which contains model name, model version, and artifact version. See clause 10.1.8.1.3.
modelLocation	Uri	O	0..1	Location of the AI/ML model.
trainingDataset	Uri	O	0..1	Information for reference to the training dataset.
trainingConfig	Object	O	0..1	trainingConfig provided by the training service producer.
notificationDestination	Uri	O	0..1	Callback URI where the notification should be delivered to.
trainingProducerId	String	O	0..1	Identifier of training producer "apfId" as specified in SME.

10.3.8.2 Simple data types and enumerations

10.3.8.2.1 Overview

The following clauses define simple data types and enumerations that can be referenced from data structure defined in the previous clauses.

10.3.8.2.2 Simple data types

No simple data types are defined in the present document.

10.3.8.2.3 Enumerations

No enumerations are defined in the present document..

10.3.9 Error Handling

10.3.9.1 General

For the AI/ML model training API, HTTP error responses shall be supported as specified in ETSI TS 129 501 [1]. Protocol errors and application errors specified in ETSI TS 129 500 [2], Table 5.2.7.2-1, shall be supported for an HTTP method if the corresponding HTTP status codes are specified as mandatory for that HTTP method in ETSI TS 129 500 [2], Table 5.2.7.1-1.

In addition, the requirements in the following clauses are applicable for the AI/ML model training API.

10.3.9.2 Protocol Errors

No specific protocol errors are defined in the present document.

10.3.9.3 Application Errors

No additional application errors defined in the present document.

10.4 AI/ML model deployment API

10.4.1 Introduction

This API enables the API Consumer to request the deployment of an AI/ML model based on the AI/ML model deployment request service defined in R1GAP [5].

10.4.2 API version

For the AI/ML model deployment API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v1".

The AI/ML model deployment API is under development and consequently the API version shall include the pre-release version "alpha.1".

10.4.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined in clause 5.2. The <apiName> resource URI variable shall be " ai-ml-model-deployment". The <apiSpecificResourceUriPart> for each resource shall be set as described in clause 10.4.5.

Figure 10.4.3-1 shows the overall resource URI structure defined for the AI/ML model deployment API.

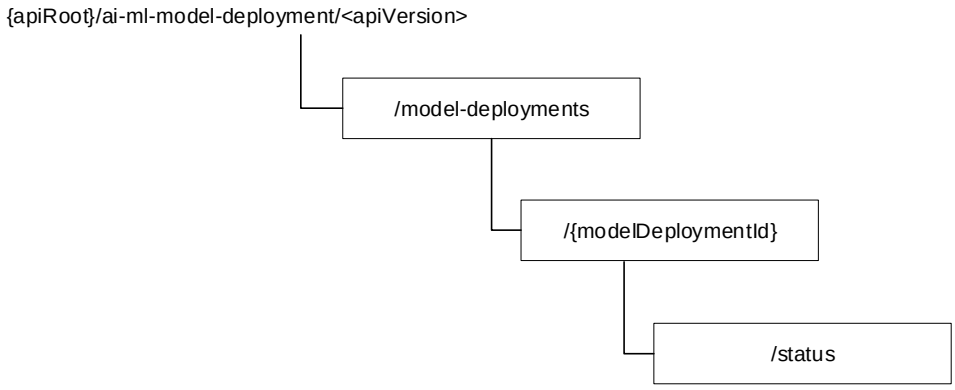

Figure 10.4.3-1: Resource URI structure of the AI/ML model deployment API

Table 10.4.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 10.4.3-1: Resource and methods overview of the AI/ML model deployment API

Resource name	Resource URI	HTTP method	Service Operation
All AI/ML model deployments	…/model-deployments	POST	Request AI/ML model deployment.
Individual AI/ML deployment status	…/model-deployments/{ modelDeploymentId }/status	GET	Query AI/ML model deployment status.

10.4.4 Service operations

10.4.4.1 Request AI/ML model deployment

10.4.4.1.1 Operation definition

The API Consumer uses this operation to request AI/ML model deployment.

The operation to request AI/ML model deployment is based on HTTP POST.

Figure 10.4.4.1.1-1: Request AI/ML model deployment operation

The service operation is as follows:

1) The API Consumer shall send an HTTP POST request to the API Producer. The target URI shall identify the resource "/modeldeployments", the message content shall carry a ModelDeploymentInformation which includes information for deployment. The API Producer shall process the request received in the HTTP POST message and determine if the request sent by the API Consumer is authorized or not.

2) The API Producer shall generate the deployment identifier and construct the URI for the created resource. The API Producer shall return the HTTP POST response. On success, "201 Created" shall be returned. The "Location" header shall be present and shall carry the URI of the new resource. The message content shall carry a ModelDeploymentInformation structure that represents the new resource. On failure, the appropriate error code shall be returned, and the message content may contain additional error information.

10.4.4.1.2 Referenced procedures

10.4.4.1.2.1 Request AI/ML model deployment procedure

The request AI/ML model deployment operation illustrated in Figure 10.4.4.1.1-1 is based on the request AI/ML model deployment procedure defined in R1GAP [5].

10.4.4.2 Notify AI/ML model deployment status change

10.4.4.2.1 Operation definition

The API Producer uses this operation to notify the change of an AI/ML model deployment status.

The operation to notify AI/ML model deployment status change is based on HTTP POST.

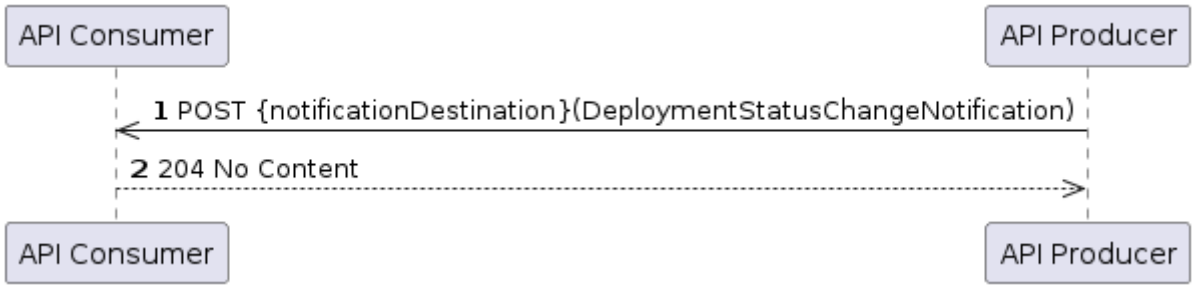

Figure 10.4.4.2.1-1: Notify AI/ML model deployment status change operation

The service operation is as follows:

1) The API Producer shall send an HTTP POST request to the API Consumer. The target URI {notificationDestination} may be provided by API Consumer during the ai/ml model deployment request, where the message content shall carry a DeploymentStatusChangeNotification which includes the updated AI/ML model deployment status. The API Consumer shall process the request received in the HTTP POST message and determine if the request sent by the API Producer is authorized or not.

2) The API Consumer shall return the HTTP POST response. On success, "204 No Content" shall be returned. The message content shall be empty. On failure, the appropriate error code shall be returned, and the response message content may contain additional error information.

10.4.4.2.2 Referenced procedures

10.4.4.2.2.1 Notify AI/ML model deployment status change procedure

The notify AI/ML model deployment status change operation illustrated in Figure 10.4.4.2.1-1 is based on the AI/ML model deployment procedure defined in R1GAP [5].

10.4.5 Resources

10.4.5.1 Overview

This clause defines the resource for the AI/ML model deployment API.

10.4.5.2 Resource: "All AI/ML model deployment"

10.4.5.2.1 Description

The resource represents all AI/ML model deployment in the Non-RT RIC.

The methods defined in clause 10.4.5.2.3 shall be supported by this resource.

10.4.5.2.2 Resource Definition

Resource URI: **{apiRoot}/ai-ml-model-deployment/<apiVersion>**

The resource URI variables supported by the resource is defined in Table 10.4.5.2.2-1.

Table 10.4.5.2.2-1: Resource URI variable for the resource

Name	Definition
apiRoot	See clause 5.2.
apiVersion	See clause 10.4.2.

10.4.5.2.3 Resource Standard Methods

10.4.5.2.3.1 POST

This method shall support the request data structures specified in Table 10.4.5.2.3.1-1 and the response data structures and response codes specified in Table 10.4.5.2.3.1-2.

Table 10.4.5.2.3.1-1: Data structures supported by the HTTP POST Request Body on this resource

Data type	P	Cardinality	Description
ModelDeploymentInformation	M	1	Information related to the deployment of the AI/ML model.

Table 10.4.5.2.3.1-2: Data structures supported by the HTTP POST Response Body on this resource

Data type	P	Cardinality	Response codes	Description
ModelDeploymentInformation			201 Created	The operation was successful. The message content of the POST response carries ModelDeploymentInformation.
ProblemDetails	O	0..1	4xx/5xx	Detailed problem description.

10.4.5.2.4 Resource Custom Methods

None.

10.4.6 Custom operation without associated resources

None.

10.4.7 Notifications

10.4.7.1 Notify AI/ML model deployment status change

10.4.7.1.1 Overview

The notification informs the receiver about the AI/ML model deployment status.

10.4.7.1.2 Resource Definition

The Resource URI {notificationDestination} is a callback URI provided when requesting deployment of an AI/ML model.

10.4.7.1.3 Resource Standard Methods

10.4.7.1.3.1 POST

This method shall support the request data structures specified in Table 10.4.7.1.3.1-1 and the response data structure and response codes specified in Table 10.4.7.1.3.1-2.

Table 10.4.7.1.3.1-1: Data structures supported by the HTTP POST request body on this resource

Data type	P	Cardinality	Description
DeploymentStatusChangeNotification	M	1	Notify a status changes of an AI/ML model deployment status.

Table 10.4.7.1.3.1-2: Data structures supported by the HTTP POST response body on this resource

Data type	P	Cardinality	Response codes	Description
N/A			204 No Content	Confirmation of received notification
ProblemDetails	O	0..1	4xx/5xx	The operation was unsuccessful. Detailed problem description may be carried in the response message content

10.4.8 Data Model

10.4.8.1 Structured data types

10.4.8.1.1 Overview

The following clauses define the structured data types and their attributes to be used by the AI/ML model deployment API.

10.4.8.1.2 Data type: ModelDeploymentInformation

The ModelDeploymentInformation data type represents deployment information for an AI/ML model. It contains the attributes defined in Table 10.4.8.1.2-1.

Table 10.4.8.1.2-1: Definition of type ModelDeploymentInformation

Attribute Name	Data type	P	Cardinality	Description
modelId	ModelId	M	1	Identifier of a model. See clause 10.1.8.1.3.
notificationDestination	Uri	O	0..1	Callback URI where the notification would be delivered to, once the deployment status changes.

10.5 AI/ML model retrieve API

10.5.1 Introduction

This API enables the API Consumer to request the retrieve of an AI/ML model based on the AI/ML model retrieve request service defined in R1GAP [5].

10.5.2 API version

For the AI/ML model retrieve API as specified in the present document, the MAJOR version field shall be 1, the MINOR version field shall be 0 and the PATCH version field shall be 0 (see clause 4.3.1.1 of ETSI TS 129 501 [1] for a definition of the version fields). Consequently, the <apiVersion> URI path segment shall be set to "v1".

The AI/ML model retrieve API is under development and consequently the API version shall include the pre-release version "alpha.1".

10.5.3 Resource structure and methods

The request URIs used in HTTP requests from the API Consumer towards the API Producer shall have the resource URI structure as defined in clause 5.2. The <apiName> resource URI variable shall be "ai-ml-model-retrieve".

Figure 10.5.3-1 shows the overall resource URI structure defined for the AI/ML model retrieve API.

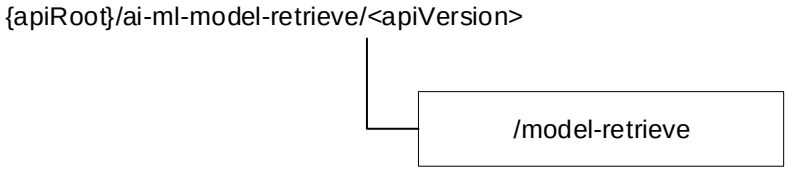

Figure 10.5.3-1: Resource URI structure of AI/ML model retrieve API

Table 10.5.3-1 lists the individual resources defined for the API, the applicable HTTP methods, and the associated service operations.

Table 10.5.3-1: Resource and methods overview of the AI/ML model retrieve API

Resource name	Resource URI	HTTP method	Service Operation
AI/ML model retrieve	…/model-retrieve	GET	Retrieve AI/ML model

Annex A (normative): OpenAPI specifications

A.1 General

A.1.1 Overview

This annex formally specifies the RESTful R1 service APIs by defining OpenAPI documents in YAML format that comply with the OpenAPI Specification v3.0.3 [3].

The Open API specifications of the RESTful R1 service APIs provided in this annex are versioned as described in clause 5.2.

The OpenAPIs defined in this annex has references to the common definitions defined in clause A.1.2 of the present document.

A.1.2 Common schemas for general use

A.1.2.1 Introduction

The Open API specified in clause A.1.2.2 provides schemas for general data types and responses for usage across the R1 APIs.

A.1.2.2 Common definitions

```
openapi: 3.0.3
info:
  title: 'R1 Common definitions'
  version: 1.0.0
  description: |
    R1 Common definitions - O-RAN.WG2.R1AP_Common.yaml.
    © 2025, O-RAN ALLIANCE.
    All rights reserved.
externalDocs:
  description: 'O-RAN.WG2.R1AP-v08.00'
  url: 'https://www.o-ran.org/specifications'
paths: {}
components:
  schemas:
    Uri:
      description: 'A string formatted according to IETF RFC 3986 [8].'
      type: string
    ProblemDetails:
      description: 'A problem detail to carry details in an HTTP response according to IETF RFC
7807'
      type: object
      properties:
        type:
          description: 'a URI reference according to IETF RFC 3986 that identifies the problem type'
          type: string
        title:
          description: 'human-readable summary of the problem type'
          type: string
        status:
          description: 'the HTTP status code'
          type: number
        detail:
          description: 'human-readable explanation '
          type: string
        instance:
          description: 'URI reference that identifies the specific occurrence of the problem'
          type: string
  responses:
```

```
  '400':
     description: 'Bad Request'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
  '401':
     description: 'Unauthorized'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
  '403':
     description: 'Forbidden'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
  '404':
     description: 'Not Found'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
  '405':
     description: 'Method Not Allowed'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
  '406':
     description: 'Not Acceptable'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
  '409':
     description: 'Conflict'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
  '411':
     description: 'Length Required'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
  '413':
     description: 'Payload Too Large'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
  '414':
     description: 'URI Too Large'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
  '415':
     description: 'Unsupported Media Type'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
  '429':
     description: 'Too Many Requests'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
  '500':
     description: 'Internal Server Error'
     content:
       application/problem+json:
         schema:
           $ref: '#/components/schemas/ProblemDetails'
```

```
   '502':
      description: 'Bad Gateway'
      content:
        application/problem+json:
          schema:
            $ref: '#/components/schemas/ProblemDetails'
   '503':
      description: 'Service Unavailable'
      content:
        application/problem+json:
          schema:
            $ref: '#/components/schemas/ProblemDetails'
```

A.2 Service management and exposure service

A.2.1 Service registration API

A.2.1.1 Introduction

The Open API for this service is reusing the CAPIF_Publish_Service_API as specified in clause A.2.1.2 with exceptions specified in clause A.2.1.3 below.

A.2.1.2 CAPIF_Publish_Service_API

The Open API for the SME service registration API specified in clause 6.1 reuses the CAPIF_Publish_Service_API as specified in ETSI TS 129 222 [9], clause A.3.

The API version of the Service registration API as specified in clause 6.1.2 shall use the CAPIF_Publish_Service_API OpenAPI version as specified in Table A.2.1.2-1.

Table A.2.1.2-1

API name	API version	CAPIF OpenAPI version
Service registration API	1.2.0	1.3.0

A.2.1.3 Adaptations and Exceptions

The OpenAPI code below represents the "VersionExtensions" data type which needs to be added to the OpenAPI definitions of the CAPIF_Publish_Service_API defined in ETSI TS 129 222 [9].

```
components:
  schemas:
    VersionExtensions:
      Description: ' The VersionExtensions data structure specified in table B.3.4.1-1 in R1AP
defines O-RAN extensions to the CAPIF Version data type which allows to signal the versions
supported for an interface endpoint.'
      type: object
      properties:
        fullApiVersions:
          description: ' List of version strings, as defined in R1GAP clause 5.2, to signal the API
versions supported by the API Producer for a particular major API version.'
          type: array
          items:
            type: string
          minItems: 1
```

The OpenAPI code below represents the "ServiceProperties" data type which needs to be added to the OpenAPI definitions of the CAPIF_Publish_Service_API defined in ETSI TS 129 222 [9].

```
components:
  schemas:
    ServiceProperties:
      description: 'Defines a container that can be used by individual service APIs to register and
discover service-specific properties.'
```

```
      type: object
      properties:
        serviceCapabilities:
          description: 'Service capabilities. The content of this attribute is service-specific and
is defined per service.'
          type: object
```

Table A.2.1.3-1 below lists exceptions and adaptations related to certain attributes in certain CAPIF data types when re-using the CAPIF_Publish_Service_API in the context of the present document.

Table A.2.1.3-1: Message content exceptions and adaptations when reusing the CAPIF_Publish_Service_API

Data type	Reference	Attributes	Adaptations/Exceptions
ServiceAPIDescription	ETSI TS 129 222 [9], clause A.3	supportedFeatures	Not required to be supported.
		shareableInfo	Not required to be supported.
		serviceAPICategory	Not required to be supported.
		ccfId	Not required to be supported.
		apiSuppFeats	Not required to be supported.
			Not required to be supported.
		vendorSpecific-o-ran.org	This additional attribute shall be present if service-specific registration information is available for the service API. It shall have the type ServiceProperties as specified in clause B.3.4.2 and defined in clause A.2.1.3.
AefProfile	ETSI TS 129 222 [9], clause A.3	aefId	Shall be set to the value of "rAppId" if an rApp produces the API.
		aefLocation	Not required to be supported.
		domainName	Not required to be supported.
		serviceKpis	Not required to be supported.
		ueIpRange	Not required to be supported.
Version	ETSI TS 129 222 [9] clause A.3	vendorSpecific-o-ran.org	This additional attribute shall be supported and shall have the type VersionExtensions as specified in clause B.3.4.1 and defined above.

A.2.2 Service discovery API

A.2.2.1 Introduction

The Open API for this service is reusing the CAPIF_Discover_Service_API in as specified in clause A.2.2.2 with exceptions specified in clause A.2.2.3 below.

A.2.2.2 CAPIF_Discovery_Service_API

The Open API for the SME service discovery API specified in clause 6.2reuses the CAPIF_Discover_Service_API as specified in ETSI TS 129 222 [9], clause A.2.

The API version of the Service discovery API as specified in clause 6.2.2 shall use the CAPIF_Discover_Service_API OpenAPI version as specified in Table A.2.2.2-1.

Table A.2.2.2-1

API name	API version	CAPIF OpenAPI version
Service discover API	1.2.0	1.3.0

A.2.2.3 Adaptations and Exceptions

Table A.2.2.3-1 lists exceptions and adaptations related to CAPIF message content when re-using the CAPIF_Discover_Service _API in the context of the present document.

Table A.2.2.3-1 lists exceptions and adaptations related to CAPIF URI query parameters when re-using the CAPIF_Discover_Service_API in the context of the present document.

Table A.2.2.3-1: Message content exceptions and adaptations when reusing the CAPIF_Discover_Service_API

Data type	Reference	Attributes	Exceptions/Adaptations
ServiceAPIDescription	ETSI TS 129 222 [9], clause A.3	supportedFeatures	Not required to be supported.
		shareableInfo	Not required to be supported.
		serviceAPICategory	Not required to be supported.
		ccfId	Not required to be supported.
		apiSuppFeats	Not required to be supported.
		pubApiPath	Not required to be supported.
		vendorSpecific-o-ran.org	This additional attribute shall be present if service-specific registration information is available for the service API. It shall have the type ServiceProperties as specified in clause B.3.4.2 and defined in clause A.2.1.3.
		ue-ip-addr	Not required to be supported.
		service-kpis	Not required to be supported.
AefProfile	ETSI TS 129 222 [9], clause A.3	aefId	Shall be set to the value of "rAppId" if an rApp produces the API. Shall be set to an identifier related to the SMO/Non-RT RIC functions if these produce the API. See note.
		aefLocation	Not required to be supported.
		domainName	Not required to be supported.
Version	ETSI TS 129 222 [9], clause A.3	vendorSpecific-o-ran.org	This additional attribute shall be supported and shall have the type VersionExtensions as specified in clause B.3.4.1 and defined in clause A.2.1.3.
NOTE: It is out of scope of the present document whether each SMO/Non-RT RIC framework function is identifiable by a separate identifier value or whether such decomposition information is hidden from the rApps. However, the SMO/Non-RT RIC framework shall ensure that the identifiers used for rApps as AEFs and the identifiers used for SMO/Non-RT RIC framework functions as AEFs do not collide.			

The Query parameters in Table A.2.2.3-2 below are not required to be supported when the CAPIF_Discover_Service_API is reused.

Table A.2.2.3-2: Query parameters exceptions and adaptations when reusing the CAPIF_Discover_Service_API

Resource	Reference	Exceptions	Comment
/allServiceAPIs	ETSI TS 129 222 [9], clause A.2	api-invoker-id	Shall be set to the value of "rAppId" for an API-consuming rApp.
	ETSI TS 129 222 [9], clause A.2	comm-type	Not required to be supported.
	ETSI TS 129 222 [9], clause A.2	protocol	Not required to be supported.
	ETSI TS 129 222 [9], clause A.2	aef-id	Not required to be supported.
	ETSI TS 129 222 [9], clause A.2	data-format	Not required to be supported.
	ETSI TS 129 222 [9], clause A.2	api-cat	Not required to be supported.
	ETSI TS 129 222 [9], clause A.2	preferred-aef-loc	Not required to be supported.
	ETSI TS 129 222 [9], clause A.2	supported-features	Not required to be supported.
	ETSI TS 129 222 [9], clause A.2	api-supported-features	Not required to be supported.
	ETSI TS 129 222 [9], clause A.2	ue-ip-addr	Not required to be supported.
	ETSI TS 129 222 [9], clause A.2	service-kpis	Not required to be supported.

A.2.3 Service events subscription API

A.2.3.1 Introduction

The Open API for this service is reusing the CAPIF_Events_API as specified in clause A.2.3.2 with exceptions specified in clause A.2.3.3 below.

A.2.3.2 CAPIF_Events_API

The Open API for the SME service events subscription API specified in clause 6.3 reuses the CAPIF_Events_API as specified in ETSI TS 129 222 [9], clause A.4.

The API version of the SME service events subscription API as specified in clause 6.3.2 shall use the CAPIF_Events_API OpenAPI version as specified in Table A.2.3.2-1

Table A.2.3.2-1

API name	API version	CAPIF OpenAPI version
Service events subscription API	1.2.0	1.3.0

A.2.3.3 Adaptations and Exceptions

Table A.2.3.3-1 lists exceptions and adaptations related to CAPIF message content when re-using the CAPIF_Events_API in the context of the present document.

Table A.2.3.3-1: Message content exceptions and adaptations when reusing the CAPIF_Events_API

Data type	Reference	Attributes	Exceptions/Adaptations
AccessControlPolicyListExt	ETSI TS 129 222 [9], clause A.4		Not required to be supported.
CAPIFEvent	ETSI TS 129 222 [9], clause A.4		See clause B.3.5.
CAPIFEventDetail	ETSI TS 129 222 [9], clause A.4	accCtrlPolList	Not required to be supported.
		invocationLogs	Not required to be supported.
		apiTopoHide	Not required to be supported.
CAPIFEventFilter	ETSI TS 129 222 [9], clause A.4	apiInvokerIds	Shall be set to the value of "rAppId" for an API-consuming rApp.
		aefIds	Shall be set to the value of "rAppId" if an rApp -produces the API. Shall be set to an identifier related to the SMO/Non-RT RIC framework functions if these produce the API. See Table A.2.2.3-1.
		ReportingInformation	Not required to be supported.
EventNotification	ETSI TS 129 222 [9], clause A.4	invocationLogs	Not required to be supported.
		apiTopoHide	Not required to be supported.
EventSubscription	ETSI TS 129 222 [9], clause A.4	supportedFeatures	Not required to be supported.
		eventReq	Not required to be supported.
TopologyHiding	ETSI TS 129 222 [9], clause A.4		Not required to be supported.

A.2.4 Bootstrap API

A.2.4.1 Introduction

The Open API for this service as specified in clause A.2.4.2.

A.2.4.2 Bootstrap API

```
openapi: 3.0.3
info:
  title: 'BootStrap'
  version: 1.0.0
  description: |
    API for BootStrap service.
    © 2025, O-RAN ALLIANCE.
    All rights reserved.
externalDocs:
  description: 'O-RAN.WG2.R1AP-v08.00'
  url: 'https://www.o-ran.org/specifications'
servers:
  - url: '{apiRoot}/bootstrap/v1/'
    variables:
      apiRoot:
        description: 'apiRoot as defined in clause 5.3 in O-RAN.WG2.R1AP'
        default: 'https://example.com'
paths:
  '/bootstrap-info':
    get:
      description: 'To discover the entry points into Service management and exposure '
      responses:
        '200':
          description: '.'
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/BootstrapInformation'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
components:
  schemas:
    BootstrapInformation:
      type: object
      properties:
        apiEndpoints:
          type: array
          items:
            $ref: '#/components/schemas/ApiEndpointInformation'
    ApiEndpointInformation:
      type: object
      properties:
        apiName:
          type: string
          description: 'Name of the API ("service-apis" or "published-apis")'
        tokenEndpoint:
          $ref: 'TS29122_CAPIF_publish_Service_API.yaml#/components/schemas/InterfaceDescription'
          description: 'Token endpoint shall be provided if the API requires authorization over
OAuth2.0'
          nullable: true
        apiEndPoint:
          $ref: 'TS29122_CAPIF_publish_Service_API.yaml#/components/schemas/InterfaceDescription'
          description: 'End point of the API'
          nullable: true
      required:
        - apiName
        - apiEndPoint
```

A.3 Data management and exposure service

A.3.1 Data registration API

A.3.1.1 Introduction

The Open API for the Data Registration API is specified in clause A.3.1.2.

A.3.1.2 Data registration API

```
openapi: 3.0.3
info:
  title: 'Data registration service'
  version: 2.0.0-alpha.2
  description: |
    API for Data registration service.
    © 2025, O-RAN ALLIANCE.
    All rights reserved.
externalDocs:
  description: 'O-RAN.WG2.R1AP-v08.00'
  url: 'https://www.o-ran.org/specifications'
servers:
  - url: '{apiRoot}/data-registration/v2/'
    variables:
      apiRoot:
        description: 'apiRoot as defined in clause 5.3 in O-RAN.WG2.R1AP'
        default: 'https://example.com'
      apiConsumerId:
        description: Identifier of the API consumer that registers its data production capabilities
        default: ''
paths:
  '/production-capabilities':
    post:
      description: 'To register DME type production capabilities'
      tags:
      - Registered DME type production capabilities
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/DmeTypeRelatedCapabilities'
      responses:
        '201':
          description: 'Success case 201 created'
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/DmeTypeRelatedCapabilities'
          headers:
            Location:
              description: 'Contains the URI of the newly created resource'
              required: true
              schema:
               type: string
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '405':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/405'
        '409':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/409'
        '413':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
        '415':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
```

```
      '429':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
      '500':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
      '503':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'

'/production-capabilities/{registrationId}':
  parameters:
    - name: registrationId
      in: path
      required: true
      schema:
        $ref: '#/components/schemas/registrationId'
  put:
    description: 'To update DME type production capabilities that it has previously registered'
    tags:
    - Individual registered DME type production capability
    requestBody:
      required: true
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/DmeTypeRelatedCapabilities'
    responses:
      '200':
        description: 'Success case 200 with updated information'
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/DmeTypeRelatedCapabilities'
      '400':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
      '401':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
      '403':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
      '404':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
      '406':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
      '411':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/411'
      '413':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
      '415':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
      '429':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
      '500':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
      '503':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
  get:
    description: 'To query DME type production capabilities that it has previously registered'
    tags:
    - Individual registered DME type production capability
    responses:
      '200':
        description: 'Success case 200 with queried information'
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/DmeTypeRelatedCapabilities'
      '400':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
      '401':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
      '403':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
      '404':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
      '406':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
```

```
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
    delete:
      description: 'To deregister DME type production capabilities'
      tags:
      - Individual registered DME type production capability
      responses:
        '204':
          description: 'The registration was deleted'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
components:
  schemas:
   registrationId:
     description: 'A successful registration identified by registrationId '
     type: string
   DmeTypeRelatedCapabilities:
     description: 'Information related to the registration as producer of a DME type'
     type: object
     properties:
       dmeTypeDefinition:
         $ref: '#/components/schemas/DmeTypeDefinition'
       constraints:
         description: 'Formulates producer constraints or constraints applicable to the consumption
related to the DME type based on the dataProductionSchema'
         type: object
       dataAccessEndpoint:
         $ref: 'TS29222_CAPIF_Publish_Service_API.yaml#/components/schemas/InterfaceDescription'
       dataDeliveryModes:
        type: array
        items:
          $ref: 'O-RAN.WG2.R1AP_DataAccess.yaml#/components/schemas/DataDeliveryMode'
     required: [dmeTypeDefinition, dataDeliveryModes, dataAccessEndpoint]
   DmeTypeDefinition:
     description: 'Information of the DME type'
     type: object
     properties:
       dmeTypeId:
         $ref: '#/components/schemas/DmeTypeIdStruct'
       metadata:
         $ref: '#/components/schemas/Metadata'
       dataProductionSchema:
         type: object
         description: 'Schema that defines the information necessary to formulate a data request or
data subscription.   If this attribute is not present, the schema is assumed to be known from the
DME type definition that is referenced by dmeTypeId'
       dataDeliverySchemas:
        description: 'List of delivery schemas supported by the producer for the DME type being
registered.'
        type: array
        items:
          $ref: '#/components/schemas/DeliverySchema'
       dataDeliveryMechanisms:
        description: 'Defining the delivery mechanism supported by Data Producer '
        type: array
        items:
          $ref: '#/components/schemas/DataDeliveryMechanism'
     required: ["dmeTypeId", "metadata", "dataDeliverySchemas", "dataDeliveryMechanisms"]
```

```
   DmeTypeIdStruct:
     description: 'Defining the attributes of DME type identifier'
     type: object
     properties:
       namespace:
         type: string
         description: 'Indicating the entity responsible for the DME type definition.'
       name:
         type: string
         description: 'Name of the DME type. The string can be any character except ":" (colon)'
         pattern: '^[^:]{1,}$'
       version:
         type: string
         description: 'Version of the DME type. The versioning and allowed characters are according
to SemVer [11]'
     required: ["namespace","name","version"]
   DeliverySchema:
    description: 'Delivery schema for a DME type'
    type: object
    properties:
       type:
         $ref: '#/components/schemas/SchemaTypes'
       deliverySchemaId:
         type: string
         description: A Data Producer may support one or more delivery schemas and for each
supported schema type a delivery schema identifier is assigned. A Data Consumer uses this attribute
while creating a data job and request to deliver the data using specific schema type which is
identified by this attribute.
       schema:
         type: string
         description: 'The schema serialized to string. If this attribute is not present, the schema
is assumed to be known from the DME type definition that is referenced by dmeTypeId'
    required: ["type","deliverySchemaId"]
   DataDeliveryMechanism:
     description: 'Defining the attributes of delivery mechanism supported'
     type: object
     properties:
      dataDeliveryMethod:
        description: 'Delivery Method supported'
        ref: 'O-RAN.WG2.R1AP_DataAccess.yaml#/components/schemas/DataDeliveryMethod'
      kafkaDeliveryConfiguration:
        $ref: '#/components/schemas/KafkaDeliveryConfiguration'
     required:
     - dataDeliveryMethod
     oneOf:
      - required: ["kafkaDeliveryConfiguration"]
   Metadata:
     description: 'Metadata that can be used in discovering the DME type'
     properties:
       dataCategory:
         description: 'Defines the category of the DME type e.g. PM counters'
         type: array
         items:
          type: string
         minItems: 1
       rat:
         description: 'Defines the radio access technology e.g. 5G'
         type: array
         items:
          type: string
         minItems: 1
     required: ["dataCategory"]
   SchemaTypes:
     description: 'Type of the schema supported by Data Producers'
     type: string
     enum:
      - JSON_SCHEMA
      - XML_SCHEMA
   KafkaDeliveryConfiguration:
     description: 'These configuration will be applied if STREAMING_KAFKA is selected as delivery
method'
     type: object
     properties:
      numPartitions:
        description: 'Number of partitions'
        type: integer
      cleanUpPolicy:
```

```
        description: 'cleanUpPolicy is based on cleanup.policy defined in the Kafka Documentation
[15]. '
        type: string
      compressionType:
        description: ' compressionType is based on compression.type defined in the Kafka
Documentation [15] .'
        type: string
      retentionBytes:
        description: ' retentionBytes is based on retention.bytes defined in the Kafka Documentation
[15] . This attribute is applicable ONLY when cleanUpPolicy is set to DELETE'
        type: integer
      retentionMs:
        description: ' retentionMs is based on retention.ms defined in the Kafka Documentation [15]
. This attribute is applicable ONLY when cleanUpPolicy is set to DELETE'
        type: integer
     required: ["cleanUpPolicy", "compressionType"]
```

A.3.2 Data discovery API

A.3.2.1 Introduction

The Open API for the Data Discovery API is specified in clause A.3.2.2.

A.3.2.2 Data discovery API

```
openapi: 3.0.3
info:
  title: 'Data discovery service'
  version: 2.0.0
  description: |
    API for Data discovery service.
    © 2025, O-RAN ALLIANCE.
    All rights reserved.
externalDocs:
  description: 'O-RAN.WG2.R1AP-v08.00'
  url: 'https://www.o-ran.org/specifications'
servers:
  - url: '{apiRoot}/data-discovery/v2/'
    variables:
      apiRoot:
        description: 'apiRoot as defined in clause 5.3 in O-RAN.WG2.R1AP'
        default: 'https://example.com'
paths:
  /dme-types:
    get:
      description: 'To discover the available DME types'
      parameters:
        - name: identity-namespace
          in: query
          description: 'Identity namespace to match the "namespace" part of the "dmeTypeId"
attribute'
          schema:
            type: string
        - name: identity-name
          in: query
          description: 'Identity name to match the "name" part of the "dmeTypeId" attribute.'
          schema:
            type: string
        - name: data-category
          in: query
          description: 'Set of data category entries, all of which to match entries of the
"dataCategory" attribute.'
          schema:
            type: array
            explode: false
            items:
              type: string
      responses:
        '200':
          description: 'The response body contains the result of the search over the list of
registered APIs.'
          content:
            application/json:
```

```
              schema:
                type: array
                items:
                  $ref: 'O-
RAN.WG2.R1AP_DataRegistration.yaml#/components/schemas/DmeTypeRelatedCapabilities'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '406':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
        '414':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/414'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
  /dme-types/{dmeTypeId}:
    get:
      description: To obtain information about an individual DME type.
      parameters:
        - name: dmeTypeId
          in: path
          required: true
          schema:
           $ref: '#/components/schemas/dmeTypeId'
      responses:
        '200':
          description: The response body contains information about the DME type.
          content:
            application/json:
              schema:
                $ref: 'O-
RAN.WG2.R1AP_DataRegistration.yaml#/components/schemas/DmeTypeRelatedCapabilities'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '406':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
        '414':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/414'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
components:
  schemas:
   dmeTypeId:
     type: string
     description: 'The DmeTypeId is constructed based on the three parts separated by ":" (colon)
{dmeTypeId} = {namespace}:{name}:{version}. See O-
RAN.WG2.R1AP_DataRegistration.yaml#/components/schemas/DmeTypeIdStruct for the definition of
"namespace", "name" and "version".'
```

A.3.3 Data access API

A.3.3.1 Introduction

The Open API for the Data access API is specified in clause A.3.3.2.

A.3.3.2 Data access API

```
openapi: 3.0.3
info:
  title: 'Data access service'
  version: 2.0.0-alpha.2
  description: |
    API for Data access service.
    © 2025, O-RAN ALLIANCE.
    All rights reserved.
externalDocs:
  description: 'O-RAN.WG2.R1AP-v08.00'
  url: 'https://www.o-ran.org/specifications'
servers:
  - url: '{apiRoot}/data-access/v1/'
    variables:
      apiRoot:
        description: 'apiRoot as defined in clause 5.3 in O-RAN.WG2.R1AP'
        default: 'https://example.com'
      apiConsumerId:
        description: 'Identifier of the API consumer '
        default: ''
paths:
  '/data-jobs':
    post:
      description: 'To create a data job'
      tags:
      - Create all data jobs
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/DataJobInfo'
      responses:
        '201':
          description: 'Success case 201 created'
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/DataJobInfo'
          headers:
            Location:
              description: 'Contains the URI of the newly created resource'
              required: true
              schema:
               type: string
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '405':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/405'
        '409':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/409'
        '413':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
        '415':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
```

```
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
      callbacks:
        DataAvailabilityNotification:
          '{$request.body.dataAvailabilityNotificationUri}':
           post:
            description: 'Notification on the availability of requested data'
            requestBody:
              required: true
              content:
                application/json:
                  schema:
                    $ref: '#/components/schemas/DataAvailabilityNotification'
            responses:
              '204':
                description: 'The notification was delivered'
              '400':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
              '401':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
              '403':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
              '404':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
              '429':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
              '500':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
              '502':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
              '503':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
  '/data-jobs/{dataJobId}':
    parameters:
      - name: dataJobId
        in: path
        required: true
        schema:
          $ref: '#/components/schemas/dataJobId'
    delete:
      description: 'To delete the created data job'
      tags:
      - Individual data job
      responses:
        '204':
          description: 'The data job was deleted'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
components:
  schemas:
    dataJobId:
      description: 'A successful created data job is identified by dataJobId '
      type: string
    DataJobInfo:
      description: 'Information related to a data job'
      allOf:
        - type: object
          properties:
            dataDeliveryMode:
              $ref: '#/components/schemas/DataDeliveryMode'
            dmeTypeId:
              $ref: 'O-RAN.WG2.R1AP_DataDiscovery.yaml#/components/schemas/dmetypeId'
            productionJobDefinition:
```

```
              description: 'Job description based on the DME type specific dataProductionSchema'
              type: object
            dataDeliveryMethod:
              $ref: '#/components/schemas/DeliveryMethod'
            dataDeliverySchemaId:
              description: 'A delivery schema identifier provided by a Data Producer during the data
registration procedure'
              type: string
          required: [dataDeliveryMode, dmeTypeId, productionJobDefinition, dataDeliveryMethod,
dataDeliverySchemaId]
        - type: object
          oneOf:
            - properties:
                pullDeliveryDetailsHttp:
                  $ref: '#/components/schemas/PullDeliveryDetailsHttp'
              required: [pullDeliveryDetailsHttp]
            - properties:
                dataAvailabilityNotificationUri:
                  $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/schemas/Uri'
              required: [dataAvailabilityNotificationUri]
            - properties:
                pushDeliveryDetailsHttp:
                  $ref: '#/components/schemas/PushDeliveryDetailsHttp'
              required: [pushDeliveryDetailsHttp]
            - properties:
                streamingConfigurationKafka:
                  $ref: '#/components/schemas/StreamingConfigurationKafka'
              required: [streamingConfigurationKafka]
    DataDeliveryMode:
      description: 'This indicates whether the data instance is created in a one-time data delivery
(data request) or continuously (data subscription)'
      type: string
      enum:
        - ONE_TIME
        - CONTINUOUS
    DataDeliveryMethod:
      description: 'This indicates supported delivery method'
      type: string
      enum:
        - PULL_HTTP
        - PUSH_HTTP
        - STREAMING_KAFKA
    PullDeliveryDetailsHttp:
      description: 'The PullDeliveryDetailsHttp data type signals how to pull data using the HTTP
protocol.'
      readOnly: true
      type: object
      properties:
        dataPullUri:
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/schemas/Uri'
      required: [dataPullUri]
    PushDeliveryDetailsHttp:
      description: 'The PushDeliveryDetailsHttp data type signals how to push data using the HTTP
protocol.'
      type: object
      properties:
        dataPushUri:
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/schemas/Uri'
      required: [dataPushUri]
    StreamingConfigurationKafka:
      description: 'The StreamingConfigurationKafka data type signals a data streaming configuration
for the Kafka protocol.'
      type: object
      properties:
        topicName:
          description: 'Name of the Kafka topic'
          type: string
        kafkaBootstrapServers:
          description: 'Server configuration'
          type: array
          items:
            $ref: '#/components/schemas/ServerAddressWithPort'
      required: [topicName, kafkaBootstrapServers]
    ServerAddressWithPort:
      description: 'Server configuration'
      type: object
      properties:
        hostname:
```

```
          description: 'string identifying a hostname shall be formatted according to clause 2.3.1
as defined in IETF RFC 1035 [19]'
          type: string
        portAddress:
          description: 'Port address, e.g. 9092'
          type: integer
          minimum: 1
          maximum: 65535
      required: [hostname, portAddress]
    DataAvailabilityNotification:
      description: 'Availability of the data'
      type: object
      properties:
        dataJobId:
          description: 'data job identifier'
          type: string
        pullDeliveryDetailsHttp:
          $ref: '#/components/schemas/PullDeliveryDetailsHttp'
```

A.3.4 HTTP based Push data API

Deliberately, no OpenAPI is specified for this API in the present document.

NOTE: OpenAPI requires the definition of the valid content types for the request message content. However, this API is agnostic with respect to the content type of the data carried in the request message. Within the DME services, the valid content types are defined as part of separate data message schemas which specify the structure of the data to be carried over the API. Because of the requirement to fix the content types that can be carried over the API, defining an OpenAPI would restrict the versatility of the API.

A.3.5 HTTP based Pull data API

Deliberately, no OpenAPI is specified for this API in the present document.

NOTE: OpenAPI requires the definition of the valid content types for the response message content. However, this API is agnostic with respect to the content type of the data carried in the response message. Within the DME services, the valid content types are defined as part of a separate data message schema which specifies the structure of the data to be carried over the API. Because of the requirement to fix the content types that can be carried over the API, defining an OpenAPI would restrict the versatility of the API.

A.3.6 Data offer API

A.3.6.1 Introduction

The Open API for the Data offer API is specified in clause A.3.6.2.

A.3.6.2 Data offer API

```
openapi: 3.0.3
info:
  title: 'Data offer service'
  version: 1.0.0
  description: |
    API for Data offer service.
    © 2025, O-RAN ALLIANCE.
    All rights reserved.
externalDocs:
  description: 'O-RAN.WG2.R1AP-v08.00'
  url: 'https://www.o-ran.org/specifications'
servers:
  - url: '{apiRoot}/data-offer/v1'
    variables:
      apiRoot:
        description: 'apiRoot as defined in clause 5.3 in O-RAN.WG2.R1AP'
        default: 'https://example.com'
      apiConsumerId:
        description: 'Identifier of API Consumer'
```

```
        default: ''
paths:
  '/offers':
    post:
      description: 'Allows to create a new data offer'
      tags:
      - All data offers
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/DataOfferInfo'
      responses:
        '201':
          description: 'Success case 201 created'
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/DataOfferInfo'
          headers:
            Location:
              description: 'Contains the URI of the newly created resource'
              required: true
              schema:
               type: string
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '405':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/405'
        '409':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/409'
        '413':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
        '415':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
      callbacks:
        DataAvailabilityNotification:
          '{$request.body.dataAvailabilityNotificationUri}':
           post:
            description: 'Notification on the availability of offered data'
            requestBody:
              required: true
              content:
                application/json:
                  schema:
                    $ref: ' O-RAN.WG2.R1AP_DataAccess.yaml
#/components/schemas/DataAvailabilityNotification'
            responses:
              '204':
                description: 'The notification was delivered'
              '400':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
              '401':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
              '403':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
              '404':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
              '429':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
              '500':
                $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
              '502':
```

```
              $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
            '503':
              $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
      DataOfferTerminationNotification:
        '{$request.body.dataOfferTerminationNotificationUri}':
         post:
          description: 'Notification on termination of data offer by the API producer'
          requestBody:
            required: true
            content:
              application/json:
                schema:
                  $ref: '#/components/schemas/DataOfferTerminationNotification'
          responses:
            '204':
              description: 'The notification was delivered'
            '400':
              $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
            '401':
              $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
            '403':
              $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
            '404':
              $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
            '429':
              $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
            '500':
              $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
            '502':
              $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
            '503':
              $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
  '/offers/{dataOfferId}':
    parameters:
      - name: dataOfferId
        in: path
        required: true
        schema:
          type: string
    delete:
      description: 'To delete the data offer'
      tags:
      - Individual data offer
      responses:
        '204':
          description: 'The data offer was deleted.'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
components:
  schemas:
    DataOfferInfo:
      description: 'Information related to a data offer'
      type: object
      properties:
       dataDeliveryMode:
         $ref: 'O-RAN.WG2.R1AP_DataAccess.yaml#/components/schemas/DataDeliveryMode'
       dmeTypeId:
         $ref: 'O-RAN.WG2.R1AP_DataDiscovery.yaml#/components/schemas/DmeTypeId'
       productionJobDefinition:
         description: 'Job description based on the DME type specific dataProductionSchema'
         type: object
       dataDeliveryMethods:
         type: array
         items:
```

```
          $ref: ' O-RAN.WG2.R1AP_DataAccess.yaml#/components/schemas/DataDeliveryMethod'
      dataDeliverySchemaIds:
        description: 'A delivery schema identifier provided by a Data Producer during the data
registration procedure'
        type: array
        items:
          type: string
      pullDeliveryDetailsHttp:
        $ref: ' O-RAN.WG2.R1AP_DataAccess.yaml#/components/schemas/PullDeliveryDetailsHttp'
      dataAvailabilityNotificationUri:
        $ref: '/components/schemas/DataOfferAvailabilityNotification'
      dataOfferTerminationNotificationUri:
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/schemas/Uri'
      pushDeliveryDetailsHttp:
        $ref: 'O-RAN.WG2.R1AP_DataAccess.yaml#/components/schemas/PushDeliveryDetailsHttp'
      streamingConfigurationKafka:
        $ref: 'O-RAN.WG2.R1AP_DataAccess.yaml#/components/schemas/StreamingConfigurationKafka'
     required: [ "dataDeliveryMode", "dmeTypeId", "productionJobDefinition", " dataDeliveryMethods
", "dataDeliverySchemaIds", "dataOfferTerminationNotificationUri"]
   DataOfferTerminationNotification:
    description: 'Termination of a data offer by the API producer'
    type: object
    properties:
      dataOfferId:
        description: 'Identifies the deleted data offer'
        type: string
    required: ["dataOfferId"]
DataOfferAvailabilityNotification:
      description: 'Availability of the data'
      type: object
      properties:
        dataOfferId:
          description: 'data offer identifier'
          type: string
        pullDeliveryDetailsHttp:
          $ref: 'O-RAN.WG2.R1AP_DataAccess.yaml#/components/schemas/PullDeliveryDetailsHttp'
      required: ['dataOfferId', 'pullDeliveryDetailsHttp']:
        description: 'Identifies the deleted data offer'
        type: string
    required: ["dataOfferId"]
```

A.4 RAN OAM related services

A.4.1 Configuration management API

A.4.1.1 Introduction

The Open API for this service is reusing the provisioning management service API as specified in clause A.4.1.2 with exceptions specified in clause A.4.1.3 below.

A.4.1.2 Configuration management API

The Open API for the Configuration management API specified in clause 8.1 reuses the Provisioning management service API (TS28532_ProvMnS.yaml) as specified in ETSI TS 128 532 [20], clause A.1.1.

The API version of the Configuration management API as specified in clause 8.1.2 shall use the Provisioning management service OpenAPI version as specified in Table A.4.1.2-1.

Table A.4.1.2-1

API name	API version	ProvMnS OpenAPI version
Configuration management API	1.0.0	17.5.0

A.4.1.3 Adaptations and Exceptions

Table A.4.1.3-1 lists exceptions and adaptations related to HTTP methods when re-using the TS28532_ProvMnS_API in the context of the present document.

Table A.4.1.3-1: HTTP methods exceptions and adaptations when reusing the TS28532_ProvMnS_API

HTTP Methods	Reference	Attributes	Exceptions/Adaptations
PUT	ETSI TS 128 532 [20], clause A.1.1		HTTP operation not required to be supported in current version.
POST	ETSI TS 128 532 [20], clause A.1.1		HTTP operation not required to be supported in current version.
DELETE	ETSI TS 128 532 [20], clause A.1.1		HTTP operation not required to be supported in current version.

A.4.2 Fault management API

A.4.2.1 Introduction

The Open API for this service is reusing the provisioning management service API as specified in clause A.4.2.2 with exceptions specified in clause A.4.2.3 below.

A.4.2.2 Fault management API

The Open API for the fault management API specified in clause 8.1 reuses the Provisioning management service API (TS28111_FaultNrm.yaml) as specified in ETSI TS 128 111 [26], clause A.1.3.

The API version of the fault management API as specified in clause 8.2.2 shall use the FaultNRM service OpenAPI version as specified in Table A.4.2.2-1.

Table A.4.2.2-1

API name	API version	FaultNrm OpenAPI
Fault management API	1.0.0	18.1.0

A.4.2.3 Adaptations and Exceptions

NOTE: The adaptations and exceptions are not specified in the present document.

A.4.3 Configuration schema information API

A.4.3.1 Introduction

The Open API for the Configuration schema information API is specified in clause A.4.3.2.

A.4.3.2 Configuration schema information API

```
openapi: 3.0.3
info:
  title: 'Configuration Schema information API'
  version: 1.0.0-alpha.1
  description:This API enables the API Consumer to retrieve configuration schema information  based
on the procedure defined in R1GAP [5].
    © 2025, O-RAN ALLIANCE.
    All rights reserved.
externalDocs:
  description: 'O-RAN.WG2.R1AP-v08.00'
```

```
  url: 'https://www.o-ran.org/specifications'
servers:
  - url: '{apiRoot}/ran-oam-cm-schema-info/v1'
    variables:
      apiRoot:
        description: 'apiRoot as defined in clause 10.3 in O-RAN.WG2.R1AP'
        default: 'https://example.com'
paths:
  '/schemas':
    get:
      summary: Get all schema information
      description: 'Retrieves configuration schema information'
      responses:
        '200':
          description: 'The results with the schema information'
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref:'#/components/schemas/Schema'

        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '405':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/405'
        '409':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/409'
        '413':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
        '415':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
  '/schemas/{schemaId}':
    get:
      summary: Get individual schema information
      parameters:
        - name: schemaId
          in: path
          required: true
          description: Where schemaId = "name" (for latest revision or if revision is not available)
or "name@revision" for a specific revision.
          schema:
            type: string
      responses:
        '200':
          description: The results with the schema information matching the request.
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Schema'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '405':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/405'
        '409':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/409'
        '413':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
```

```
        '415':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
components:
  schemas:
    Schema:
      type: object
      required:
        - name
        - location
        - type
      properties:
        name:
          type: string
          description: The unique name of the schema.
        revision:
          type: string
          description: The unique revision of the schema.
        location:
          type: string
          format: uri
          description: Location of where to find the schema content.
        type:
          $ref: '#/components/schemas/SchemaType'
          description:This data type is designed to provide extensibility. As specified in clause
8.3.8.2.3.1, Only YANG is supported in the present version of the present document.
        namespace:
          type: string
          description: The namespace of the schema, (See NOTE 1, clause 8.3.8.2.3.1, only applicable
if type is YANG).
        features:
          type: array
          items:
            type: string
          description: List of supported features (See NOTE 1,clause 8.3.8.2.3.1, only applicable if
type is YANG).
        deviations:
          type: array
          items:
            type: string
          description: List of schema deviation(s) (See NOTE 1,clause 8.3.8.2.3.1, only applicable
if type is YANG).
    SchemaType:
      type: string
      enum:
        - YANG
      description: Enumeration of schema types.
```

A.5 A1 related service

A.5.1 A1 policy management API

A.5.1.1 Introduction

The Open API for the A1 policy management API is specified in clause A.5.1.2.

A.5.1.2 A1 policy management API

```
openapi: 3.0.3
info:
  title: 'A1 policy management API'
  version: 1.0.0
  description: |
    API for A1 policy management service.
    © 2025, O-RAN ALLIANCE.
    All rights reserved.
externalDocs:
  description: 'O-RAN.WG2.R1AP-v08.00'
  url: 'https://www.o-ran.org/specifications'
servers:
  - url: '{apiRoot}/a1-policy-management/v1'
    variables:
      apiRoot:
        description: 'apiRoot as defined in clause 5.3 in O-RAN.WG2.R1AP'
        default: 'https://example.com'
paths:
  '/policy-types':
    get:
      description: 'To query A1 policy type identifier'
      tags:
      - All A1 policy types
      parameters:
        - name: nearRtRicId
          in: query
          description: 'The identifier of Near-RT RIC'
          schema:
            type: string
        - name: typeName
          in: query
          description: 'The unique label of the policy type'
          schema:
            type: string
      responses:
        '200':
          description: 'Success case 200 with queried information'
          content:
            application/json:
              schema:
                 type: array
                 items:
                    $ref: '#/components/schemas/PolicyTypeInformation'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '406':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
  '/policy-types/{policyTypeId}':
    parameters:
      - name: policyTypeId
        in: path
        required: true
        schema:
          $ref: '#/components/schemas/policyTypeId'
    get:
      description: 'To query A1 policy type'
      tags:
      - Individual A1 policy type
      responses:
        '200':
          description: 'Success case 200 with queried information'
          content:
```

```
          application/json:
            schema:
              $ref: '#/components/schemas/PolicyTypeObject'
      '400':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
      '401':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
      '403':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
      '404':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
      '406':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
      '429':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
      '500':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
      '503':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
'/policies':
  get:
    description: 'To query A1 policy type'
    tags:
    - All A1 policies
    parameters:
      - name: nearRtRicId
        in: query
        description: 'The identifier of Near-RT RIC'
        schema:
          type: string
      - name: policyTypeId
        in: query
        description: 'The identifier of the policy'
        schema:
          type: string
    responses:
      '200':
        description: 'Success case 200 with queried information'
        content:
          application/json:
            schema:
               type: array
               items:
                  $ref: '#/components/schemas/PolicyInformation'
      '400':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
      '401':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
      '403':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
      '404':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
      '406':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
      '429':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
      '500':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
      '503':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
  post:
    description: 'To create A1 policies'
    tags:
    - All A1 policies
    requestBody:
      required: true
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/PolicyObjectInformation'
    responses:
      '201':
        description: 'Success case 201 created'
        content:
```

```
          application/json:
            schema:
              $ref: '#/components/schemas/PolicyObjectInformation'
        headers:
          Location:
            description: 'Contains the URI of the newly created resource'
            required: true
            schema:
             type: string
      '400':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
      '401':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
      '403':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
      '404':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
      '405':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/405'
      '409':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/409'
      '413':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
      '415':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
      '429':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
      '500':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
      '503':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
'/policies/{policyId}':
  parameters:
    - name: policyId
      in: path
      required: true
      schema:
        $ref: '#/components/schemas/policyId'
  put:
    description: 'To update a created policy'
    tags:
    - Individual A1 policy
    requestBody:
      required: true
      content:
        application/json:
          schema:
            "$ref": "#/components/schemas/PolicyObject"
    responses:
      200:
        description: 'The policy was updated'
        content:
          application/json:
            schema:
              "$ref": "#/components/schemas/PolicyObject"
      '400':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
      '401':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
      '403':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
      '404':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
      '406':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
      '411':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/411'
      '413':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
      '415':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
      '429':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
      '500':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
```

```
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
    get:
      description: 'To query created A1 policy'
      tags:
      - Individual A1 policy
      responses:
        '200':
          description: 'Success case 200 with queried information'
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/PolicyObject'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '406':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
    delete:
      description: 'To delete the created A1 policy'
      tags:
      - Individual A1 policy
      responses:
        '204':
          description: 'The created A1 policy was deleted'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '406':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
  '/policies/subscriptions':
    post:
      summary: Create a new A1 policy status subscription
      description: This operation creates a new subscription for receiving A1 policy status
notifications.
      tags:
      - A1 policy status subscription
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/PolicyStatusSubscription'
      responses:
        '201':
          description: 'Success case 201 created'
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/PolicyStatusSubscription'
          headers:
```

```
          Location:
            description: 'Contains the URI of the newly created resource'
            required: true
            schema:
             type: string
      '400':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
      '401':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
      '403':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
      '404':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
      '406':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
      '429':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
      '500':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
      '503':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
'/policies/subscriptions/{subscriptionId}':
  parameters:
    - name: subscriptionId
      in: path
      required: true
      schema:
        $ref: '#/components/schemas/subscriptionId'
  get:
    summary: Get details of an A1 policy status subscription
    description: This operation retrieves information about a specific subscription.
    tags:
    - A1 policy status subscription
    responses:
      '200':
        description: OK
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/PolicyStatusSubscription'
      '400':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
      '401':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
      '403':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
      '404':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
      '406':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
      '429':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
      '500':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
      '503':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
  put:
    description: 'This operation modifies an existing subscription.'
    tags:
    - A1 policy status subscription
    requestBody:
      required: true
      content:
        application/json:
          schema:
            "$ref": "#/components/schemas/PolicyStatusSubscription"
    responses:
      200:
        description: 'The policy was updated'
        content:
          application/json:
            schema:
              "$ref": "#/components/schemas/PolicyObject"
      '400':
```

```
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '406':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
        '411':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/411'
        '413':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
        '415':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'

    delete:
      summary: Delete an A1 policy status subscription
      description: This operation removes a subscription.
      tags:
      - A1 policy status subscription
      responses:
        '204':
          description: No Content
components:
  schemas:
   policyTypeId:
    description: 'Policy Type identifier as defined in A1AP [23], clause 6.2.3.1.3'
    type: string
   policyId:
    description: 'Policy Identifier of a policy'
    type: string
   NearRtRicId:
    description: 'Near-RT RIC identifier'
    type: string
   subscriptionId:
    description: 'subscription identity of the policy'
    type: string
   PolicyObject:
    description: 'Policy Object is a JSON representation of an A1 policy; the A1 policies are
specified in A1TD [24]'
    type: object
   PolicyTypeObject:
    description: 'policy type object as defined in A1TD'
    type: object
   PolicyTypeInformation:
    description: 'Available policy types and for each policy type identifier the Near-RT RIC
identifiers of those Near-RT RICs that support the related A1 policy type'
    type: object
    properties:
      policyTypeId:
        description: 'Identity of the policy type'
        type: string
      nearRtRicId:
        $ref: '#/components/schemas/NearRtRicId'
    required: ["policyTypeId","nearRtRicId"]
   PolicyInformation:
    description: 'Near-RT RIC identifiers where A1 policies exist and for each Near-RT RIC
identifier the policy identifiers of those policies that exist in that Near-RT RIC'
    type: object
    properties:
      policyId:
        description:
        $ref: '#/components/schemas/policyId'

      nearRtRicId:
        $ref: '#/components/schemas/NearRtRicId'
    required: ["policyId","nearRtRicId"]
   PolicyObjectInformation:
    description: 'Information related to the creation of the policy'
```

```
    type: object
    properties:
      policyObject:
        description: 'Policy Object is a JSON representation of an A1 policy; the A1 policies are
specified in A1TD [24]'
        type: object
      nearRtRicId:
        $ref: '#/components/schemas/NearRtRicId'
      policyTypeId:
          $ref: "#/components/schemas/policyTypeId"
    required: ["policyObject","nearRtRicId"]
   PolicyStatusSubscription:
      description: 'PolicyStatusSubscription data type represents the subscription information of A1
policy status'
      type: object
      properties:
        subscriptionScope:
          $ref: '#/components/schemas/QueryFilter'
        notificationDestination:
          type: URI
          description: URI for policy status notifications
        policyIdList:
          type: array
          items:
            $ref: '#/components/schemas/policyId'
        policyTypeIdList:
          type: array
          items:
            $ref: '#/components/schemas/ policyTypeId
        nearRtRicIdList:
          type: array
          items:
            $ref: '#/components/schemas/ nearRtRicId
   QueryFilter:
          type: string
          enum:
              - OWN #'indicate the A1 policies created by API Consumer '
              - OTHERS #'indicate the A1 policies created other API Consumers'
              - ALL #'indicate the A1 policies created by any API Consumers'
```

A.6 AI/ML workflow service

A.6.1 AI/ML model registration API

A.6.1.1 Introduction

The Open API for the AI/ML model registration API is specified in clause A.6.1.2.

A.6.1.2 AI/ML model registration API

```
openapi: 3.0.3
info:
  title: 'AI/ML Model registration API '
  version: 1.0.0
  description: API for registering an AI/ML model|
    API for AI/ML Model registration service.
    © 2025, O-RAN ALLIANCE.
    All rights reserved.
externalDocs:
  description: 'O-RAN.WG2.R1AP-v08.00'
  url: 'https://www.o-ran.org/specifications'
servers:
  - url: '{apiRoot}/ai-ml-model-registration/{apiVersion}'
paths:
  '/model-registrations':
    post:
      description: 'Register a new AI/ML model'
      tags:
      -  Registered AI/ML Model registration details
      requestBody:
```

```
      required: true
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/ModelRelatedInformation'
    responses:
      '201':
        description: 'Success case 201 created'
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/ModelRelatedInformation'
        headers:
          Location:
            description: 'Contains the URI of the newly created resource'
            required: true
            schema:
             type: string
      '400':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
      '401':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
      '403':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
      '404':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
      '405':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/405'
      '409':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/409'
      '413':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
      '415':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
      '429':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
      '500':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
      '503':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'

  '/model-registrations/{modelRegistrationId}':
    get:
      summary: Get details of a registered AI/ML model
      parameters:
        - in: path
          name: modelRegistrationId
          required: true
          schema:
            type: string
      responses:
        '200':
          description: AI/ML model details retrieved successfully
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/ModelRelatedInformation'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '405':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/405'
        '409':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/409'
        '413':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
        '415':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
```

```
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
      '503':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
  put:
    summary: Update information of a registered AI/Ml model
    parameters:
      - in: path
        name: modelRegistrationId
        required: true
        schema:
          type: string
    requestBody:
      required: true
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/ModelRelatedInformation'
    responses:
      '200':
        description: AI/ML model information updated successfully
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/ModelRelatedInformation'
      '400':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
      '401':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
      '403':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
      '404':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
      '405':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/405'
      '409':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/409'
      '413':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
      '415':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
      '429':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
      '500':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
      '503':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'

  delete:
    summary: Deregister an AI/ML model
    parameters:
      - in: path
        name: modelRegistrationId
        required: true
        schema:
          type: string
    responses:
      '204':
        description: AI/ML model deregistered successfully
      '400':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
      '401':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
      '403':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
      '404':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
      '429':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
      '500':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
      '503':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
```

```
components:
  schemas:
    ModelRelatedInformation:
      type: object
      properties:
        modelId:
          $ref: '#/components/schemas/ModelId'
        description:
          type: string
        modelInformation:
          $ref: '#/components/schemas/ModelInformation'
        modelLocation:
          type: string
          format: uri
      required:
        - modelId
        - description
        - modelInformation
    ModelId:
      type: object
      description: 'Identifier of a model'
      properties:
        modelName:
          type: string
          description: 'name of AI/ML model as specified in R1GAP'
        modelVersion:
          type: string
          description: 'version of AI/ML model as specified in R1GAP'
        artifactVersion:
          type: string
          format: uri
          description: 'artifact version of AI/ML model as specified in R1GAP'
      required:
        - modelName
        - modelVersion
    ModelInformation:
      type: object
      properties:
        metadata:
          description: 'Meta data of AI/Ml Model'
          $ref: '#/components/schemas/MetaData'
        inputDataType:
          type: array
          description: 'Input data type for the model, the structure of  dataTypeId is specified in
clause 7.1.8'
          items:
            $ref: 'O-RAN.WG2.R1AP_DataRegistration.yaml#/components/schemas/DataTypeId'  #
DataTypeId is specified (clause 7.1.8)
        outputDataType:
          type: array
          description: 'Output data type for the model, the structure of  dataTypeId is specified in
(clause 7.1.8)'
          items:
            $ref: 'O-RAN.WG2.R1AP_DataRegistration.yaml#/components/schemas/DataTypeId'  #
DataTypeId is specified (clause 7.1.8)
        targetEnvironment:
          type: array
          description: 'Information on the target environment is required for deployment of an AI/ML
model'
          items:
            $ref: '#/components/schemas/TargetEnvironment'
      required:
        - metadata
        - inputDataType
        - outputDataType
    MetaData:
      type: object
      properties:
        author:
          type: string
          description: 'Author of an AI/ML model'
        owner:
          type: string
      required:
        - author
    TargetEnvironment:
      type: object
      properties:
```

```
        platformName:
          type: string
          description: 'Name of the platform'
        environmentType:
          type: string
          description: 'Name of the platform execution service  type, and this is dependent on the
platformName'
        dependencyList:
          type: string
          format: uri
          description: 'Location to the template that has all the list of dependencies platform must
provide needs to be installed for the model. (for example, scikit-learn 0.21.3)'
      required:
        - platformName
        - environmentType
        - dependencyList
```

A.6.2 AI/ML model discovery API

A.6.2.1 Introduction

The Open API for the AI/ML model discovery API is specified in clause A.6.2.2.

A.6.2.2 AI/ML model discovery API

```
openapi: 3.0.3
info:
  title: 'AI/ML Model discovery API'
  version: 1.0.0
  description: |
    API for AI/Ml Model discovery service.
    © 2025, O-RAN ALLIANCE.
    All rights reserved.
externalDocs:
  description: 'O-RAN.WG2.R1AP-v08.00'
  url: 'https://www.o-ran.org/specifications'
servers:
  - url: '{apiRoot}/ai-ml-model-discovery/v1'
    variables:
      apiRoot:
        description: 'apiRoot as defined in clause 10.3 in O-RAN.WG2.R1AP'
        default: 'https://example.com'
paths:
  '/models':
    get:
      description: 'This operation retrieves all registered AI/ML models'
      parameters:
        - name: model-name
          in: query
          description: 'name of the model as specified in R1GAP[5]'
          schema:
            type: string
        - name: model-version
          in: query
          description: 'name of the model as specified in R1GAP[5]'
          schema:
            type: string
      responses:
        '200':
          description: 'The response body contains the result of the search over the list of
ModelRelatedInformation that includes the modelId and metadata..'
          content:
            application/json:
              schema:
                type: array
                items:
                $ref: '#/components/schemas/ModelRelatedInformation'

        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
```

```
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '406':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
        '414':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/414'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
        tags:
            - AI/ML Model Discovery

components:
  schemas:
    ModelRelatedInformation:
      type: object
      description: "The ModelRelatedInformation data type represents registration information for an
AI/ML model."
      required:
        - modelId
        - metadata
      properties:
        modelId:
          $ref: 'O-RAN.WG2.R1AP_AI/Ml_model_registrtaion_API.yaml/components/schemas/ModelId'
          description: "Identifier of a model as specified in clause 10.1.8.1.3"
        metadata:
          type: string
          description: "Description of the AIML model that includes the AI/ML model related
information and training related information."
```

A.6.3 AI/ML model training API

A.6.3.1 Introduction

The Open API for the AI/Ml model API is specified in clause A.6.3.2.

A.6.3.2 AI/ML model training API

```
openapi: 3.0.3
info:
  title: 'AI/ML Model Training API'
  version: 1.0.0-alpha.1
  description: |
    API for AI/Ml Model training service.
    © 2025, O-RAN ALLIANCE.
    All rights reserved.
externalDocs:
  description: 'O-RAN.WG2.R1AP-v08.00'
  url: 'https://www.o-ran.org/specifications'
servers:
  - url: '{apiRoot}/ai-ml-model-training/v1'
    variables:
      apiRoot:
        description: 'apiRoot as defined in clause 10.3 in O-RAN.WG2.R1AP'
        default: 'https://example.com'
paths:
  '/training-jobs':
    post:
      summary: Create a new AI/ML model training job
      tags:
        - Creation of AI/ML model training job
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/TrainingJobDescription"
```

```
    responses:
      '201':
        description: 'Success case 201 created'
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/TrainingJobInfo'
        headers:
          Location:
            description: 'Contains the URI of the newly created resource'
            required: true
            schema:
             type: string
      '400':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
      '401':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
      '403':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
      '404':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
      '405':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/405'
      '409':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/409'
      '413':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/413'
      '415':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/415'
      '429':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
      '500':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
      '502':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
      '503':
        $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'

'/training-jobs/{trainingJobId}':
    delete:
      summary: 'Delete an AI/ML model training job'
      tags:
        - Delete AI/ML model training job
      responses:
        '204':
          description:'The AI/ML model training job wasI/ML deleted'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '406':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
'/training-jobs/{trainingJobId}/status':
  get:
    summary: Get the status of an AI/ML model training job
    tags:
      - Status of AI/ML model training job
    parameters:
      - in: path
        name: trainingJobId
        required: true
        schema:
          type: string
    responses:
      '200':
        description: 'Training job status retrieved successfully'
```

```
          content:
            application/json:
              schema:
                 type: array
                 items:
                    $ref: '#/components/schemas/TrainingJobStatus'
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '406':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/406'
        '429':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/429'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'
        '502':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/502'
        '503':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/503'
components:
  schemas:
    TrainingJobDescription:
      type: object
      properties:
        modelId:
          $ref: "#O-RAN.WG2.R1AP_ai_ml_model_registration.yaml/components/schemas/modelId"
          description: "The model identifier which contains model name, model version, and artifact
version see clause 10.1.8.1.3-1 in R1AP."
        modelLocation:
          type: string
          format: uri
          description: "Location of the AI/ML model."
        trainingDataset:
          type: string
          format: uri
          description: "Information for reference to the training dataset."
        validationDataset:
          type: string
          format: uri
          description: "Information for reference to the validation dataset."
        trainingConfig:
          type: object
          description: "Configuration of AI/ML model training based on training service producer
specific trainingConfigSchema."
        notificationDestination:
          type: string
          format: uri
          description: "Callback URI where the notification should be delivered to."
        consumerRAppId:
          type: string
          description: "rAppId of the training service consumer rApp."
        producerRAppId:
          type: string
          description: "rAppId of the training service producer rApp."
      required:
        - modelId
```

A.6.4 AI/ML model deployment API

A.6.4.1 Introduction

The Open API for the AI/Ml model deployment API is specified in clause A.6.4.2.

A.6.4.2 AI/ML model deployment API

```
openapi: 3.0.3
info:
  title: 'AI/ML Model Deployment API'
  version: 1.0.0-alpha.1
  description: |
    API for AI/ML Model Deployment service.
    © 2025, O-RAN ALLIANCE.
    All rights reserved.
externalDocs:
  description: 'O-RAN.WG2.R1AP-v08.00'
  url: 'https://www.o-ran.org/specifications'
servers:
  - url: '{apiRoot}/ai-ml-model-deployment/v1'
    variables:
      apiRoot:
        description: 'apiRoot as defined in clause 5.2 in O-RAN.WG2.R1AP'
        default: 'https://example.com'
paths:
  '/model-deployments':
    post:
      summary: Request AI/ML model deployment
      tags:
        - Request AI/ML model deployment
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/ModelDeploymentInformation'
      responses:
        '201':
          description: 'AI/ML model deployment created successfully'
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/ModelDeploymentInformation'
          headers:
            Location:
              description: 'URI of the created deployment resource'
              required: true
              schema:
                type: string
        '400':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/400'
        '401':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/401'
        '403':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/403'
        '404':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/404'
        '500':
          $ref: 'O-RAN.WG2.R1AP_Common.yaml#/components/responses/500'


components:
  schemas:
    ModelDeploymentInformation:
      type: object
      properties:
        modelId:
          $ref: '#/components/schemas/ModelId'
          description: 'Unique identifier for the AI/ML model.'
        deploymentLocation:
          type: string
          format: uri
          description: 'URI for the deployment location.'
        deploymentConfig:
          type: object
          description: 'Configuration parameters for AI/ML model deployment.
      required:
        - modelId
        - deploymentLocation

    DeploymentStatus:
      type: object
      properties:
```

```
        deploymentId:
          type: string
          description: 'Identifier for the AI/ML model deployment.'
        status:
          type: string
          description: 'Current status of the deployment.'
        details:
          type: string
          description: 'Additional details about the deployment status.'
```

Annex B (normative): Common data types for R1 service APIs

B.1 Introduction

In the subsequent clauses, common data types for the following areas are defined:

- Generic usage.
- Service management and exposure.
- Data management and exposure.
- RAN OAM.

B.2 Common data types for generic usage

B.2.1 Introduction

This clause defines common data types for generic usage.

B.2.2 Simple data types

Table B.2.2-1: Simple data types for generic use

Type name	Type Definition	Description
Uri	string	A string formatted according to IETF RFC 3986 [8].

B.2.3 Enumeration

B.2.3.1 Void

B.2.4 Structured data types

B.2.4.1 Data type: ProblemDetails

The ProblemDetails structure is specified in Table B.2.4.1-1. It is based on IETF RFC 7807 [10] and shall comply with the provisions defined there.

Table B.2.4.1-1: Definition of type ProblemDetails

Attribute name	Data type	P	Cardinality	Description
type	string	O	0..1	URI reference according to IETF RFC 3986 [8] that identifies the problem type.
title	string	O	0..1	Human-readable summary of the problem type.
status	number	O	0..1	The HTTP status code.
detail	string	O	0..1	Human-readable explanation.
instance	string	O	0..1	URI reference that identifies the specific occurrence of the problem.

B.2.4.2 Void

B.2.4.3 Void

B.2.4.4 Void

B.2.5 Re-used data types

Re-used data types are not defined in the present document.

B.3 Common data types for Service management and exposure

B.3.1 Introduction

This clause defines common data types for usage in the service management and exposure APIs.

Clause B.3.5 references and profiles data types defined in external specifications and defines restrictions on and adaptations of these data type for their usage within the context of the service management and exposure APIs.

B.3.2 Simple data types

None.

B.3.3 Enumerations

None.

B.3.4 Structured data types

B.3.4.1 Data type: VersionExtensions

The VersionExtensions data structure specified in Table B.3.4.1-1 defines extensions to the CAPIF "Version" data type which allows to signal the versions supported for an interface endpoint.

Table B.3.4.1-1: Definition of type VersionExtensions

Attribute name	Data type	P	Cardinality	Description
fullApiVersions	array(string)	M	1..N	List of version strings, as defined in clause 5.2, to signal the API versions supported by the API Producer for a particular major API version.

B.3.4.2 Data type: ServiceProperties

The ServiceProperties data structure specified in Table B.3.4.2-1 defines a container to be used to register and discover service-specific properties of individual service APIs.

Table B.3.4.2-1: Definition of type ServiceProperties

Attribute name	Data type	P	Cardinality	Description
serviceCapabilities	object	M	0..1	Service capabilities. The content of this attribute is service-specific and is defined per service.

B.3.5 Re-used data types

Table B.3.5-1 references data types that are reused from external documents by the service management and exposure APIs and defines restrictions on and adaptations of these data type for their usage within the context of the service management and exposure APIs.

NOTE: All externally-defined data types that are directly used in request or response message content of the service management and exposure APIs (such as ServiceAPIDescription) are listed below. In addition, those externally-defined data types are listed that are descendants of a directly used data type and for which restrictions and adaptations are defined in the present document. Other external descendant data types are not listed, as they are referenced directly in the external specification.

Table B.3.5-1: Re-used data types

Data type	Reference	Comments	Applicability
ServiceAPIDescription	ETSI TS 129 222 [9], clause 8.2.4.2.2	The following attributes are not applicable in the context of the R1 SME services and therefore need not be supported: "supportedFeatures", "shareableInfo", "serviceAPICategory", "ccfId", "apiSuppFeats", "pubApiPath".	
InterfaceDescription	ETSI TS 129 222 [9], clause 8.2.4.2.3	The CAPIF feature "ExtendedIntfDesc" shall be supported.	
AefProfile	ETSI TS 129 222 [9], clause 8.2.4.2.4	The AEF profile holds information related to the discoverable APIs produced by a single API Producer. In case the API Producer is an rApp, the "aefId" attribute shall contain the value of the rAppID. The following attributes are not applicable in the context of the R1 SME services and need therefore not be supported: "aefLocation", "domainName""serviceKpis", and "ueIpRange".	
Version	ETSI TS 129 222 [9], clause 8.2.4.2.5	The following additional attribute shall be supported: "vendorSpecific-o-ran.org" of type VersionsList as specified in clause B.3.4.1, to signal the full API versions supported.	
CAPIFEvent	ETSI TS 129 222 [9], clause 8.3.4.3.3	The enumeration values "SERVICE_API_AVAILABLE", "SERVICE_API_UNAVAILABLE" and "SERVICE_API_UPDATE" shall be supported. The remaining enumeration values are not applicable in the context of the R1 SME services and therefore need not be supported.	

B.4 Common data types for Data Management and Exposure

B.4.1 Introduction

This clause defines common data types for DME usage.

B.4.2 Simple data types

Table B.4.2-1: Simple data types

Type Name	Type Definition	Description
DmeTypeId	string	A DME type Id is constructed based on the three parts separated by ":" (colon): {dataTypeId} = {namespace}:{name}:{version}. See clause B.4.4.1 for the definition of "namespace", "name" and "version".

B.4.3 Enumerations

B.4.3.1 Enumeration: DataDeliveryMethod

Table B.4.3.1-1: Enumeration: DataDeliveryMethod

Enumeration value	Description
STREAMING_KAFKA	Kafka based streaming delivery mechanism as defined in R1TP [7], clause 6.
PULL_HTTP	HTTP based pull delivery mechanism as defined in clause 7.5.
PUSH_HTTP	HTTP based push delivery mechanism as defined in clause 7.4.

B.4.4 Structured data types

B.4.4.1 Data type: DmeTypeIdStruct

The DmeTypeIdStruct data type contains the attributes defined in Table B.4.4.1-1.

Table B.4.4.1-1: Definition of type DmeTypeIdStruct

Attribute Name	Data type	P	Cardinality	Description
namespace	string	M	1	Indicating the entity responsible for the DME type definition.
name	string	M	1	Name of the DME type. The string shall not contain the colon ":" character.
version	string	M	1	Version of the DME type. The versioning and allowed characters are according to SemVer [11].

B.4.4.2 Data type: DataDeliveryMechanism

The DataDeliveryMechanism data type contains the attributes defined in Table B.4.4.2-1.

Table B.4.4.2-1: Definition of type DataDeliveryMechanism

Attribute Name	Data type	P	Cardinality	Description
dataDeliveryMethod	DataDeliveryMethod	M	1	Delivery method supported by a Data Producer. See clause B.4.3.1.
kafkaDeliveryConfiguration	KafkaDeliveryConfiguration	C	0..1	See clause B.4.4.3 (see note).
NOTE: This attribute shall be presented if the "deliveryMethod" attribute is set to STREAMING_KAFKA.				

B.4.4.3 Data type: KafkaDeliveryConfiguration

The KafkaDeliveryConfiguration data type contains the attributes defined in Table B.4.4.3-1.

Table B.4.4.3-1: Definition of type KafkaDeliveryConfiguration

Attribute Name	Data type	P	Cardinality	Description
numPartitions	integer	O	0..1	Number of partitions.
cleanUpPolicy	string	M	1	cleanUpPolicy is based on cleanup.policy defined in the Kafka Documentation [15].
compressionType	string	M	1	compressionType is based on compression.type defined in the Kafka Documentation [15].
retentionBytes	integer	C	0..1	retentionBytes is based on retention.bytes defined in the Kafka Documentation [15].
retentionMs	integer	C	0..1	retentionMs is based on retention.ms defined in the Kafka Documentation [15].
NOTE: Presence condition "C" this attribute may be included when cleanUpPolicy is set to DELETE.				

B.4.5 Re-used data types

Re-used data types are not defined in the present document.

B.5 Common data types for RAN OAM related services

B.5.1 Introduction

This clause defines common data types for usage in the RAN OAM related service APIs.

Clause B.5.5 references and profiles data types defined in external specifications and defines restrictions on and adaptations of these data type for their usage within the context of the RAN OAM related APIs.

B.5.2 Simple data types

None.

B.5.3 Enumerations

None.

B.5.4 Structured data types

None.

B.5.5 Re-used data types

Table B.5.5-1 references data types that are reused from external documents by the RAN OAM related service APIs and defines restrictions on and adaptations of these data type for their usage within the context of the RAN OAM related APIs.

NOTE: All externally defined data types that are directly used in request or response message content of the RAN OAM related service APIs are listed below. In addition, those externally defined data types are listed that are descendant of a directly used data type and for which restrictions and adaptations are defined in the present document. Other external ancestor data types are not listed, as they are referenced directly in the external specification.

Table B.5.5-1: Re-used data types

Data type	Reference	Comments	Applicability
Resource	ETSI TS 128 532 [20], clause 12.1.1.4.1a.1	All the attributes are supported.	
Scope	ETSI TS 128 532 [20], clause 12.1.1.4.1a.2	All the attributes are supported.	
PatchItem	ETSI TS 128 532 [20], clause 12.1.1.4.1a.9	All the attributes are supported.	

Annex C (informative): Bibliography

- ETSI GS NFV-SOL 015 (V1.2.1): "Protocols and Data Models; Specification of Patterns and Conventions for RESTful NFV-MANO APIs".

Annex D (informative): Change history

Date	Version	Information about changes
2025.03.14	V08.00	Published with updating the Bootstrap API, RAN OAM CM API's, RAN OAM FM API's, Adding Configuration schema information API, AI/ML Model deployment API, AI/Ml model retrieve API, Updates to AI/ML model registration API, AI/Ml model discovery API and AI/ML model training API.
2024.11.21	V07.00	Published with migration of all API's from Release 17 to Release 18 of 3GPP, Updated the A1 Policy management API and Model discovery API and moved the same to release version, Updated the Data registration API, Data access API and Data offer API by removing the consumerId from the uri structure.
2024.07.18	V06.00	Published with addition of Bootstrap API in SME, AI/ML model registration, AI/ML model discovery, AI/ML model training API in AI/ML workflow service, Updated A1 policy management API, Uplifted all the DME, AI/ML,A1 policy management APIs in compliance with 29.501 URI Structure, and generalized the DME API to support further enhancements. Updated the naming of data type to DME type.
2024.03.18	V05.00	Published with addition of A1 policy management API, RAN OAM CM and FM API, updates to data registration API and removing alpha indicator for data registration API.

History

Version	Date	Status
V8.0.0	March 2026	Publication