|  |
| --- |
| O-RAN.WG6.O-Cloud Notification API-v04.00 |
| Technical Specification |
| O-RAN Working Group 6  O-Cloud Notification API Specification for Event Consumers |
|  |
|  |

Copyright © 2024 by the O-RAN ALLIANCE e.V.

The copying or incorporation into any other work of part or all of the material available in this specification in any form without the prior written permission of O-RAN ALLIANCE e.V. is prohibited, save that you may print or download extracts of the material of this specification for your personal use, or copy the material of this specification for the purpose of sending to individual third parties for their information provided that you acknowledge O-RAN ALLIANCE as the source of the material and that you inform the third party that these conditions apply to them and that they must comply with them.

O-RAN ALLIANCE e.V., Buschkauler Weg 27, 53347 Alfter, Germany

Register of Associations, Bonn VR 11238, VAT ID DE321720189

Table of Contents

[Chapter 1 Introductory Material 3](#_Toc163046984)

[1.1 Scope 3](#_Toc163046985)

[1.2 References 3](#_Toc163046986)

[1.3 Definitions and Abbreviations 4](#_Toc163046987)

[1.3.1 Definitions 4](#_Toc163046988)

[1.3.2 Abbreviations 4](#_Toc163046989)

[Chapter 2 Introduction 5](#_Toc163046990)

[Chapter 3 Usage of HTTP 6](#_Toc163046991)

[3.1 General 6](#_Toc163046992)

[3.1.1 HTTP/2 shall be transported over Transmission Control Protocol (TCP), as required by HTTP/2 (see IETF RFC 7540 [8]) HTTP standard headers 6](#_Toc163046993)

[3.1.2 Content type 7](#_Toc163046994)

[3.1.3 Void 7](#_Toc163046995)

[3.1.4 Resource addressing 8](#_Toc163046996)

[Chapter 4 Subscription API Definition 10](#_Toc163046997)

[4.1 Resource Structure 10](#_Toc163046998)

[4.1.1 Resources and HTTP Methods 11](#_Toc163046999)

[4.1.2 Subscription resource definition 12](#_Toc163047000)

[4.1.3 Individual subscription resource definition 14](#_Toc163047001)

[Chapter 5 Status Notifications API Definition 17](#_Toc163047002)

[5.1 Description 17](#_Toc163047003)

[5.1.1 Event Consumer Notification Resource Definition 18](#_Toc163047004)

[Chapter 6 Event Pull Status Notifications API Definition 22](#_Toc163047005)

[6.1 Description 22](#_Toc163047006)

[6.1.1 Resources Pull Status Notification Definition 23](#_Toc163047007)

[Chapter 7 Event Data Model 25](#_Toc163047008)

[7.1 Subscription Data Model 25](#_Toc163047009)

[7.1.1 Structured data types 25](#_Toc163047010)

[7.2 Status Notifications Data Model 25](#_Toc163047011)

[7.2.1 Structured data types 25](#_Toc163047012)

[7.2.2 Event Data Model 26](#_Toc163047013)

[7.2.3 Synchronization Event Specifications 28](#_Toc163047014)

[7.3 Appendix A 33](#_Toc163047015)

[7.3.1 Helper/Sidecar containers 33](#_Toc163047016)

[Helper/Sidecar value: 33](#_Toc163047017)

# Introductory Material

## Scope

This Technical Specification has been produced by the O-RAN Alliance.

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

Release x.y.z

where:

x the first digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc. (the initial approved document will have x=01).

y the second digit is incremented when editorial only changes have been incorporated in the document.

z the third digit included only in working versions of the document indicating incremental changes during the editing process.

The present document describes a REST API that allows Event Consumers (EC) such as a O-RAN NFs to subscribe to events/status from the O-Cloud. The O-Cloud shall provide Event Producers (EP) to enable workloads to receive events/status that might be known only to the Cloud Infrastructure (CInf).

## References

The following documents contain provisions which, through reference in this text, constitute provisions of this specification (see also <https://www.o-ran.org/specifications>).

1. 3GPP TR 21.905, Vocabulary for 3GPP Specifications.
2. 3GPP TS 28.622, Telecommunication management; Generic Network Resource Model (NRM) Integration Reference Point (IRP); Information Service (IS).
3. O-RAN WG1, O-RAN Architecture Description – v02.00, Technical Specification.
4. O-RAN WG1, Operations and Maintenance Architecture – v03.00, Technical Specification.
5. O-RAN WG4, Control, User and Synchronization Plane Specification – v06.00, Technical Specification.
6. O-RAN WG6, Cloud Architecture and Deployment Scenarios for O-RAN Virtualized RAN – v02.01, Technical Report.
7. O-RAN Infrastructure Project, <https://wiki.o-ran-sc.org/display/IN/Infrastructure+Home>
8. IETF RFC 7540: "Hypertext Transfer Protocol Version 2 (HTTP/2)".
9. IETF RFC 8259: "The JavaScript Object Notation (JSON) Data Interchange Format".
10. IETF RFC 7231: "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content".
11. IETF RFC 7230: "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing".
12. IETF RFC 7807: "Problem Details for HTTP APIs".
13. IETF RFC 7235 for authentication mechanisms over HTTP/1.1,
14. 3GPP TS 29.501, 5G System; Principles and Guidelines for Services Definition
15. CloudEvents.io specification, https://github.com/cloudevents/

## Definitions and Abbreviations

### Definitions

For the purposes of the present document, the terms given in O-RAN.WG6.CADS [6] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in [6].

### Abbreviations

For the purposes of the present document, the abbreviations given in O-RAN.WG6.CADS [6] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in [6].

EC Event Consumer

EP Event Producer

REST Representational State Transfer

# Introduction

This document describes a REST API that allows Event Consumers (EC) such as a vO-DU or CNF to subscribe to events/status from the O-Cloud. The cloud infrastructure will provide Event Producers (EP) to enable cloud workloads to receive events/status that might be known only to the infrastructure.

An EC will use the REST API to subscribe to specific event types or categories of events by specifying the event / status producer address. The addressing scheme is covered in [Resource Addressing](#51w7kj7rf0x8). An EC will be able to unsubscribe from receiving events and status by deleting the subscription through the REST API. The REST API is an integration point to an event and status framework that is running in the underlying O-Cloud (IMS and/or DMS).

The REST API and associated event framework implementation is intended to be used in situations where the path from event detection to event consumption must have the lowest possible latency. Intra-node delivery of events is a primary focus with inter-node delivery also supported.

The event framework described here is not intended to be an island of communication and should interact with north-bound interfaces such as O2 through the IMS. Hence, this Event Consumers API is not intended to replace O2ims notifications (including PTP loss of sync), but rather to complement it. Please see the CAD [6] for more information.

Interfacing with external entities is necessary for communication with orchestrating entities and for permanent storage of event information for root-cause analysis. Communication with external entities is intended to be in one direction with events flowing from this framework outward. The flow of events from this framework to external entities must not affect the latency performance of the framework for intra-node or inter-node delivery.

Please note that while this API document describes an interface to general events and status provided by the cloud infrastructure, the discussions and examples in this document will focus on events and status related to PTP / Synchronization as it this is the first defined use case that affects the vO-DU per the CUSP [5] requirements.

*“If an O-DU transits to the FREERUN state, the O-DU shall disable RF transmission on all connected O-RUs, and keep it turned off until synchronization is reacquired.”*

*“Whether in ‘synchronized’ or ‘Holdover’ state, it is expected that O-DU monitors the ‘SYNCED/HOLDOVER’ status of the O-RUs under its management.”*

Please note that the timing requirements for notification regarding FREERUN should follow WG4 guidelines when available in the CUSP document. These guidelines may influence the future evolution and design of this API. Please see the CUSP [5] for more information.

Subscription/Publication use case:

* Subscription by the Event Consumer (e.g. vO-DU or other CNF) triggers the readiness of the Event Consumer to receive the notifications.
* The REST API handler implementation, provided by the Cloud infrastructure, resides in the application (workload) and is an application appropriate implementation of a REST API handler.
* Upon subscription, the EC will receive an initial notification of the EP resource status. For example, the current synchronization status of the PTP system will be sent to the EC when subscribing to the sync-status address. Or as another example, the current interface carrier status will be sent to the EC when subscribing to the interface-status address. This initial notification allows the joining application to synchronize to the current status of the system being observed.
* Event Consumers will be able to subscribe to resource status notifications offered by the cloud.
* Multiple Event Consumers in the same container, Pod, or VM can subscribe to events and status as the REST API allows multiple receive endpoint URI.
* If the eventing framework cannot provide the requested subscription the eventing framework will deny the subscription request and Event Consumer (vO-DU, vO-CU etc) will be able to make a decision if to proceed with its operation

# Usage of HTTP

## General

HTTP/2, IETF RFC 7540, shall be used.

### HTTP/2 shall be transported over Transmission Control Protocol (TCP), as required by HTTP/2 (see IETF RFC 7540 [8]) HTTP standard headers

#### Request header fields

This clause describes the usage of selected HTTP header fields of the request messages in the O-Cloud APIs.

**Table 3.1.3.2-1: Header fields supported in the request message**

| **Header field name** | **Reference** | **Descriptions** |
| --- | --- | --- |
| Accept | IETF RFC 7231 [10] | This field is used to specify response media types that are acceptable by the client sending the request.  Content-Types that are acceptable for the response.  This header field shall be present in the HTTP request message sent by the client if the response is expected to have a non-empty message body. |
| Content-Type | IETF RFC 7231 [10] | This field is used to indicate the media type of the associated representation.  This header field shall be present if the request has a non-empty message body. |
| Authorization | IETF RFC 7235 [13] | The authorization token for the request and is optional. In a local scenario (i.e. within the POD/VM) this is not mandated. If the consumer is external to the POD/VM then Authorization is mandated. |
| Accept-Encoding | IETF RFC 7231 [10] | This field may be used to indicate what response content-encodings (e.g gzip) are acceptable in the response. |
| Content-Length | IETF RFC 7230 [11] | This field is used to provide the anticipated size, as a decimal number of octets, for a potential payload body. |
| Content-Encoding | IETF RFC 7231[10] | This field may be used in some requests to indicate the content encodings (e.g gzip) applied to the resource representation beyond those inherent in the media type. |

#### Response header fields

This clause describes the usage of selected HTTP header fields of the response messages in the O-Cloud APIs.

**Table 3.1.3.3-1: Header fields supported in the response message**

| **Header field name** | **Reference** | **Descriptions** |
| --- | --- | --- |
| Content-Type | IETF RFC 7231 [10] | This header field shall be used to indicate the media type of the associated representation. |
| Content-Length | IETF RFC 7231 [10] | This header field may be used to provide the anticipated size, as a decimal number of octets, for a potential payload body.  This header field shall be present if the response has a non-empty message body. |
| Location | IETF RFC 7231 [10] | This field may be used in some responses to refer to a specific resource in relation to the response.  Used in redirection, or when a new resource has been created.  This header field shall be present if the response status code is 201 or 3xx. |
| Content-Encoding | IETF RFC 7231 [10] | This header may be used in some responses to indicate to the HTTP/2 client the content encodings (e.g gzip) applied to the resource representation beyond those inherent in the media type. |
| WWW-Authenticate | IETF RFC 7235 [13] | Challenge if the corresponding HTTP request has not provided authorization, or error details if the corresponding HTTP request has provided an invalid authorization token. This is optional. When the notification producer and consumer are locally present in the same compute, API authorization is not mandatory. |
| Retry-After | IETF RFC 7231 [10] | Used to indicate how long the user agent ought to wait before making a follow-up request.  It can be used with 503 responses.  The value of this field can be an HTTP-date or a number of seconds to delay after the response is received. |

### Content type

JSON, IETF RFC 8259 shall be used as content type of the HTTP bodies specified in the present specification.The use of the JSON format shall be signaled by the content type "application/json".

"Problem Details" JSON object shall be used to indicate additional details of the error in a HTTP response body and shall be signalled by the content type "application/problem+json", as defined in IETF RFC 7807.

### Void

.

### Resource addressing

The format of the resource address is shown in [Table 1](#table1). The resource address specifies the Event Producer with a hierarchical path. The path format provides the ability for management and monitoring to extend beyond a single cluster and node.

**Table 1: Resource address format**

|  |
| --- |
| /{clusterName}/{siteName}(/optional/hierarchy/..)/{nodeName}/{(/optional/hierarchy)/resource} |
|  |

An example hierarchy could include an IMS and DMS designator i.e., **/ims-1/dms-2/node1/*sync/sync-status/sync-state***. The event framework is minimally required to support nodeName addressing. The event framework addressing nomenclature for nodeName shall match the O-Cloud technology naming scheme.

This hierarchy path is part of the environment variables provided to the CNF by the Downward API (see <https://kubernetes.io/docs/tasks/inject-data-application/environment-variable-expose-pod-information/#the-downward-api>)

Field definitions are shown in [Table 2](#table2).

**Table 2: Resource address fields**

|  |  |  |
| --- | --- | --- |
| **Address Component** | **Description** | **Example** |
| /optional/hierarchy/nodeName/... or /./nodeName/... | The hierarchical name that uniquely specifies the DMS where the nodeName node resides. name of the cloud where the producer exists. A ‘.’ is used to indicate the current DMS where the Event Consumer nodeName node is located. The additional hierarchy is optional. If addressing begins with **/./** a nodeName or nodeName wildcard is required. | /dms1/nodeName1/... to specify a specif DMS and node, or  /./nodeName/...1 to specify the current DMS and specific node  /././…... to specify the current DMS and current node. |
| nodeName | Name of the Worker node or Compute node where the producer exists. The name must map to the nomenclature in use for the underlying cloud infrastructure. A regular expression with \* or . may be specified to subscribe to multiple nodes. | node27  node\* -> all nodes  . -> current node |
| resource | The hierarchical path for the subsystem that will produce the notifications. This path may also include an optional hierarchy to describe different Event Producers in the same Node.  The hierarchical path is inclusive such that all notifications for subsystems below the specified path will be delivered as part of the subscription.  The full path can be used to explicitly specify a single type of notification.  Multiple subscriptions can be used to select a subset of notification types for event delivery specified level. | A subscription to /***sync*** would deliver notifications for all types of synchronization events implemented by the synchronization subsystem. Since this cover all notification, individual subscriptions (as described below) will be ignored.  A subscription to /***sync/sync-status/sync-state*** would deliver notifications for the event.sync.sync-status.synchronization-state-change event only.  Individual subscriptions to /***sync/sync-status/sync-state*** *and /****sync/gnss-status/gnss-sync-status*** would deliver notifications for both the overall synchronization health (event.sync.sync-status.synchronization-state-change) and GNSS specific status (event.sync.gnss-status.gnss-state-change).  Examples for a ‘resource’ with an optional hierarchy:  *../Node1/NIC1/sync*  *../Node1/NIC2/sync/sync-status/sync-state/*  Note: In the future, Resource can be expanded to other infrastructure subsystems such as thermal notifications and network interface link status. |

# Subscription API Definition

## Resource Structure

[Figure 1](#figure1) shows the overall resource URI structure defined for the subscription’s API. [Table 3](#table3) lists the individual resources defined, and the applicable HTTP methods with the message flow diagram, [Figure 2](#figure2).

**Figure 1: Resource URI structure of the subscription’s API**

Diagram

Description automatically generated

**Figure 2: Message flow diagram**

|  |
| --- |
| [Diagram  Description automatically generated](https://lucid.app/documents/edit/c6911e15-e3c4-43e4-bcb0-579a8820c6e5/0?callback=close&name=docs&callback_type=back&v=2273&s=612)  **Helper\***  **Workload** |
| \*Helper provided by cloud vendors  **Table 3: Resources and methods overview**   |  |  |  |  | | --- | --- | --- | --- | | **Resource name** | **Resource URI** | **HTTP method or custom operation** | **Description** | | Subscriptions | {apiRoot}/ocloudNotifications/{apiMajorVersion}/subscriptions | POST | To create a new individual subscription resource. | | GET | Get a list of subscription resources. | | Individual subscription | {apiRoot}/ocloudNotifications/{apiMajorVersion} /subscriptions/{subscriptionId} | GET | Get Detail of individual subscription resources. | | DELETE | Delete individual subscription resources. | |

### Resources and HTTP Methods

An Event Consumer (e.g. vDU or other CNF) will use a POST request to subscribe to receive notifications per its desirable resource. This resource is mapped to a data type/payload (see data model).

The POST’s payload will also include the notification endpoint (callback URI) for the API Producer to send the notifications back to the EC.

The API Producer, in this case the Helper (see appendix A), will validate that the resource requested is offered by the cluster and available at the particular address. If the resource does not exist an error code will be sent to the client’s EndpointURI. This will be followed by a sanity check of the requested notification endpoint and creating the resource if communication to the notification endpoint is successful. To reduce security concerns and lifecycle management burden the notification endpoint URI must be part of the same localhost, this is the localhost shared by the Event Consumer and Helper, with the assumption that they are located in the same POD or VM.

### Subscription resource definition

The resource URI is:

**{apiRoot}/ocloudNotifications/{apiMajorVersion}/subscriptions**

The resource URI variables supported by the resource shall be defined as [Table 4](#table4) illustrates.

**Table 4: Resource URI variables for this resource**

|  |  |
| --- | --- |
| **Name** | **Definition** |
| apiRoot | described in clause 4.4.1 of 3GPP TS 29.501 |
| apiMajorVersion | v2 |

#### Subscription POST Method

The POST method creates a subscription resource for the Event Consumer. As the result of successfully executing this method, a new subscription resource shall exist as defined in clause 1.2, and a variable value (*subscriptionId*) will be used in the representation of that resource. An initial status notification for the type of event (for example, PTP synchronization status) shall be triggered. The status describes the initial status of the producer resource when successfully executing this method as defined in clause 1.1.4, followed by any PTP status notifications (triggered if there is a change in PTP status).

URI query parameters supported by the method shall be defined as [Table 5](#table5) illustrates.

**Table 5: URI query parameters supported by a method on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Data type** | **P** | **Cardinality** | **Description** | **Applicability** |
| n/a |  |  |  |  |  |

Data structures supported by the request body of the POST method shall be specified as [Table 6](#table6) illustrates.

**Table 6: Data structures supported by the request body on the resource**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data type** | **P** | **Cardinality** | **Description** |
| Subscriptioninfo | M | 1 | The payload will include an event notification request, endpointUri and ResourceAddress. See note below. |

**Note**: The *Subscriptioninfo* is defined in the subscription data model section

Data structures supported by the response body of the method shall be specified as [Table 7](#table7) illustrates.

**Table 7: Data structures supported by the response body on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Response body | **Data type** | **P** | **Cardinality** | **Response**  **codes** | **Description** |
| SubscriptionInfo | M | 1 | 201 | Shall be returned when the subscription resource is created successfully.  See note below. |
| n/a |  |  | 400 | Bad request by the EC. For example, the endpoint URI does not include ‘localhost’. |
| n/a |  |  | 404 | Subscription resource is not available. For example, PTP is not supported by the node. |
| n/a |  |  | 409 | The subscription resource already exists. |

**Note**: The *SubscriptionInfo* is defined in the subscription data model section, see [Table 30](#table30)

The following example shows a subscription request/response for /sync-state which would deliver notifications for the event.sync.sync-status.synchronization-state-change event only.

**Example Create Subscription Resource: JSON request**

|  |
| --- |
| {  {  "ResourceAddress": "/east-edge-10/Node3/sync/sync-status/sync-state/",  "EndpointUri "http://localhost:{port}/{path}  }  } |

**Example Create Subscription Resource: JSON response**

|  |
| --- |
| {  “SubscriptionId”: “789be75d-7ac3-472e-bbbc-6d62878aad4a”,  "ResourceAddress": "/east-edge-10/Node3/sync/sync-status/sync-state/",  “UriLocation”: “http://localhost:8080/ocloudNotifications/v2/subsciptions/789be75d-7ac3-472e-bbbc-6d62878aad4a”  "EndpointUri ": "[http://localhost:9090/publishers/{publisherid](http://localhost:9090/publishers/%7Bpublisherid)}"  } |

#### Subscription GET Method

The GET method queries the subscription object and its associated properties. As a result of a successful execution of this method a list of subscription object(s) and their associated properties will return by the API Producer.

URI query parameters supported by the method shall be defined as [Table 8](#table8) illustrates.

**Table 8: URI query parameters supported by a method on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Data type** | **P** | **Cardinality** | **Description** | **Applicability** |
| n/a |  |  |  |  |  |

Data structures supported by the response body of the method shall be specified as [Table 9](#table9) illustrates.

**Table 9: Data structures supported by the response body on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Response body | **Data type** | **P** | **Cardinality** | **Response**  **codes** | **Description** |
| SubscriptionInfo | M | 0..N | 200 | Returns the subscription resources and their associated properties that already exist.  See note below. |
| n/a | O | 0..1 | 400 | Bad request by the EC. For example, the endpoint URI does not include ‘localhost’. |

**Note**: The *SubscriptionInfo* is defined in the subscription data model section, see [Table 30](#table30)

### Individual subscription resource definition

The resource URI is:

**{apiRoot}/ocloudNotifications/{apiMajorVersion}/subscriptions/{subscriptionId}**

The resource URI variables supported by the resource shall be defined as [Table 10](#table10) illustrates.

**Table 10: Resource URI variables for this resource**

|  |  |
| --- | --- |
| **Name** | **Definition** |
| apiRoot | described in clause 4.4.1 of 3GPP TS 29.501 |
| apiMajorVersion | v2 |
| subscriptionId | Identifier for subscription resource, created after a successful subscription. See table Data Model’s [table 30](#table30) |

#### Individual Subscription DELTE Method

The DELETE method deletes an individual subscription resource object and its associated properties. As the result of a successful execution of this method a subscription resource object (the one associated with the *subscriptionId*) and its associated properties will be deleted by the API Producer.

URI query parameters supported by the method shall be defined as [Table 11](#table11) illustrates.

**Table 11: URI query parameters supported by a method on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Data type** | **P** | **Cardinality** | **Description** | **Applicability** |
| n/a |  |  |  |  |  |

Data structures supported by the request body of the DELETE method shall be specified as [Table 12](#table12) illustrates.

**Table 12: Data structures supported by the request body on the resource**

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

Data structures supported by the response body of the method shall be specified as [Table 13](#table13) illustrates.

**Table 13: Data structures supported by the response body on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Response body | **Data type** | **P** | **Cardinality** | **Response**  **codes** | **Description** |
| n/a |  |  | 204 | *DELETE ../subscriptions/*{subscriptionId} deletes an individual subscription resource. |

#### Individual Subscription GET Method

The GET method combined with the *subscriptionId* variable queries an individual subscription object and its associated properties. As a result of successful execution of this method an individual subscription resource object (the one associated with the *subscriptionId*) and its associated properties will return by the API Producer.

URI query parameters supported by the method shall be defined as [Table 14](#table14) illustrates.

**Table 14: URI query parameters supported by a method on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Data type** | **P** | **Cardinality** | **Description** | **Applicability** |
| n/a |  |  |  |  |  |

Data structures supported by the request body of the GET method shall be specified as [Table 15](#table15) illustrates.

**Table 15: Data structures supported by the request body on the resource**

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

Data structures supported by the response body of the method shall be specified as [Table 16](#table16) illustrates.

**Table 16: Data structures supported by the response body on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Response body | **Data type** | **P** | **Cardinality** | **Response**  **codes** | **Description** |
| SubscriptionInfo | M | 1 | 200 | Returns the subscription resource object and its associated properties.  See note below. |
| n/a |  |  | 404 | Subscription resources are not available (not created). |

**Note**: The *SubscriptionInfo* is defined in the subscription Data Model section

# Status Notifications API Definition

## Description

After a successful subscription (a subscription resource was created) the Event Consumer (e.g. vO-DU or other CNF) shall be able to receive event notifications from the subscribed resource.

Events are sent by the Event Framework when a change of resource state occurs. The significance of the change of state is dependent upon the Event Producer service. An example for the PTP use case might be that a **synchronization-state-change** has occurred, i.e. FREERUN->LOCKED or LOCKED->HOLDOVER.

The HTTP method for delivering the notification (push) to the EC shall be POST and the notification shall be sent to the endpoint reference provided by the EC client during the creation of the subscription resource (see [Table 17](#table17)  The payload body of the POST request shall contain the event payload (see event data model).

[Figure 3](#figure3) illustrates an intra-node (local notification) event delivery. In this example, the following occurs:

1. The Event Framework determines that an event condition has occurred
2. The Event Consumer (vO-DU etc) has previously subscribed to the event type and the API Producer performs a POST to the EV (vO-DU etc) with the complete JSON event payload

**Figure 3: Local Notification**

**Diagram

Description automatically generated**

**Table 17: API Producer Notification methods overview**

|  |  |  |
| --- | --- | --- |
| **Resource URI** | **HTTP method or custom operation** | **Description** |
| http://localhost:{port}/{path} | POST | **Deliver notification to subscriber.** |
| Sanity check of the endpoint URI. |
|  | | |

### Event Consumer Notification Resource Definition

The EC’s endpoint URI is used by the API Producer (Helper) to deliver events to the Event Consumer (e.g. vO-DU or CNF).

The EC’s Endpoint URI2 is:

**http://localhost:{port}/{path}**

The resource URI variables supported by the resource shall be defined as [Table 18](#table18) illustrates.

**Table 18 Resource URI variables for this resource**

|  |  |
| --- | --- |
| **Name** | **Definition** |
| Port | The port of the endpoint URI provided by the subscriber |
| Path | The path of the endpoint URI provided by the subscriber |

#### Consumer Notification Delivery Method

The HTTP method for the notification that corresponds to an explicit subscription shall be POST and the notification shall be sent to the endpoint reference provided during the creation of the subscription resource. The payload body of the POST request shall contain the event notification payload (see event data model).

URI query parameters supported by the method shall be defined as [Table 19](#table19) illustrates.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2Note: To reduce security concerns and lifecycle management burden the endpoint URI must be part of the same localhost, this is the localhost shared by the EC and API Producer in a POD or VM.

**Table 19: URI query parameters supported by a method on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Data Type** | **P** | **Cardinality** | **Description** | **Applicability** |
| n/a |  |  |  |  |  |

Data structures supported by the request body of the POST method shall be specified as [Table 20](#table20) illustrates.

**Table 20: Data structures supported by the request body on the resource**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data type** | **P** | **Cardinality** | **Description** |
| Event | M | 1 | The payload will include event notification3. |
|  | | | |

Data structures supported by the response body of the method shall be specified as [Table 21](#table21) illustrates.

**Table 21: Data structures supported by the response body on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Response body | **Data type** | **P** | **Cardinality** | **Response**  **codes** | **Description** |
| n/a | M | 1 | 204 | Success (notification was received). |
| n/a |  |  | 400 | Bad request by the API Producer. |
| n/a |  |  | 404 | Not found. |
| n/a |  |  | 408 | Request timeout. |
|  | | | | | |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3Note: The *Notification* is defined in the notification Data Model section

[Figure 4](#figure4) shows an example event notification payload received by an Event Consumer.

**Figure 4: Example Push Event Notification: request body in JSON**

|  |
| --- |
| {  "specversion": "1.0",  "type": ”event.synchronization-state-change",  "source": “/sync/sync-status/sync-state",  "id": "831e1650-001e-001b-66ab-eeb76e069631",  "time": "2021-03-05T20:59:59.998888999Z",  “data”: {  "version": “1.0”,  “values”: [  {  “type”: “notification”  “ResourceAddress”: “/east-edge-10/Node3/sync/sync-status/sync-state”,  “value\_type”: “enumeration”,  "value": ”HOLDOVER"  }  ]  }  } |
|  |

#### Notification Sanity Check Method

The Event Consumer POST request to create a subscription resource will trigger the initial delivery of producer status of the resource that will be sent to the endpoint URI provided by Event Consumer. The purpose is to confirm that the endpoint URI is valid and to send the initial status for the resource. If the validation fails, the subscription for the resource will not be created.

URI query parameters supported by the method shall be defined as [Table 22](#table22) illustrates.

**Table 22: URI query parameters supported by a method on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Data type** | **P** | **Cardinality** | **Description** | **Applicability** |
| n/a |  |  |  |  |  |

Data structures supported by the request body of the POST method shall be specified as [Table 23](#table23) illustrates.

**Table 23: Data structures supported by the request body on the resource**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data type** | **P** | **Cardinality** | **Description** |
| Event | M | 1 | The payload will include event notification. See note below. |

**Note**: The *Notification* is defined in the notification Data Model section

Data structures supported by the response body of the method shall be specified as [Table 24](#table24) illustrates.

**Table 24: Data structures supported by the response body on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Response body | **Data type** | **P** | **Cardinality** | **Response**  **codes** | **Description** |
| n/a | M | 1 | 204 | The API Producer tests the endpoint URI before creating a subscription resource. |
| n/a | O | 0..1 | 404 | URI not found. |

# Event Pull Status Notifications API Definition

## Description

In addition to receiving event status notifications the Event Consumer (e.g. vO-DU or CNF) shall be able to pull event status notifications. This status notifications will be limited only to the node that the vO-DU resides on.

[Figure 5](#figure5) illustrates event pull status notifications and [Table 25](#table25) describes resources and methods.

**Figure 5: Pull Notifications**

|  |
| --- |
| **Workload**  Event Consumer / API Consumer  ddddd  API Producer  1. GET PTP Status  **Helper**  **vDU**  2. 200 OK with event status content  Event Consumer / API Consumer  c  API Producer |
| \*Helper is provided by cloud vendors |

**Table 25: Pull Events Notifications methods overview**

|  |  |  |  |
| --- | --- | --- | --- |
| **Resource name** | **Resource URI** | **HTTP method or custom operation** | **Description** |
| Pull Status Notifications | {apiRoot}/ocloudNotifications/{apiMajorVersion}/{ResourceAddress}/CurrentState | GET | Event Consumer pulls status notifications |
|  | | | |

### Resources Pull Status Notification Definition

The resource URI is:

**{apiRoot}/ocloudNotifications/{apiMajorVersion}/{ResourceAddress}/CurrentState**

The resource URI variables supported by the resource shall be defined as [Table 26](#table26) illustrates.

**Table 26: Resource URI variables for this resource**

|  |  |
| --- | --- |
| **Name** | **Definition** |
| apiRoot | described in clause 4.4.1 of 3GPP TS 29.501 |
| apiMajorVersion | v2 |
| ResourceAddress | see [Table 1](#table1) |

#### Event Pull Status Notification GET Method

The GET method combined with the *ResourceAddress* variable pulls the event status notifications. As a result of successful execution of this method the Event Consumer will receive the current event status notifications of the node that the Event Consumer resides on.

URI query parameters supported by the method shall be defined as [Table 27](#table27) illustrates.

**Table 27: URI query parameters supported by a method on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Data type** | **P** | **Cardinality** | **Description** | **Applicability** |
| n/a |  |  |  |  |  |

Data structures supported by the request body of the GET method shall be specified as [Table 28](#table28) illustrates.

**Table 28: Data structures supported by the request body on the resource**

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

Data structures supported by the response body of the method shall be specified as [Table 29](#table29) illustrates.

**Table 29: Data structures supported by the response body on the resource**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Response body | **Data type** | **P** | **Cardinality** | **Response**  **codes** | **Description** |
| Event | M | 1 | 200 | The payload includes event notification as defined in the Data Model. |
| n/a | O | 0..1 | 404 | Event notification resource is not available on this node. |

**Editor’s note:** Currently the pull status operator returns the PTP Sync State event as defined in [PTP Sync-State](#_9s5i4y3v6j4g). In future versions of this specification, status information can be expanded to other metrics / information pertinent to the operation of the system.

# Event Data Model

## Subscription Data Model

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

### Structured data types

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

#### Type: SubscriptionInfo

**[Table 30](#table30) shows the data types used for subscription.**

**Table 30: Definition of type <SubscriptionInfo>**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute name** | **Data type** | **P** | **Cardinality** | **Description** | **Applicability** |
| SubscriptionId | string | M | 1 | Identifier for the created subscription resource.  The EC can ignore it in the POST body when creating a subscription resource (this will be sent to the client after the resource is created).  **See note 1 below.** |  |
| UriLocation | string | M | 1 | ../subscriptions/{subscriptionId}  The EC can ignore it in the POST body when creating a subscription resource (this will be sent to the client after the resource is created).  **See note 1 below.** |  |
| ResourceAddress | string | M | 1 | see [Resource Addressing](#_Resource_addressing) |  |
| EndpointUri | string | M | 1 | Endpoint URI (a.k.a callback URI), e.g. http://**localhost**:8080/resourcestatus/ptp  **Please note that ‘localhost’ is mandatory and cannot be replaced by an IP or FQDN.** |  |

**Note 1:** The API Producer (Helper) shall ignore *SubscriptionId* and *UriLocation* if sent by the EC for creating subscription.

## Status Notifications Data Model

This clause specifies the event Status Notification data model supported by the API. The current model supports JSON encoding of the CloudEvents.io specification [15] for the event payload.

### Structured data types

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

[Table 31](#table31) shows the data types used in the event data model JSON.

**Table 31: Data Model Types**

|  |  |
| --- | --- |
| **CloudEvents** | **JSON** |
| Boolean | [boolean](https://tools.ietf.org/html/rfc7159#section-3) |
| Integer | [number](https://tools.ietf.org/html/rfc7159#section-6), only the integer component optionally prefixed with a minus sign is permitted |
| String | [string](https://tools.ietf.org/html/rfc7159#section-7) |
| Binary | [string](https://tools.ietf.org/html/rfc7159#section-7), [Base64-encoded](https://tools.ietf.org/html/rfc4648#section-4) binary |
| URI | [string](https://tools.ietf.org/html/rfc7159#section-7) following [RFC 3986](https://tools.ietf.org/html/rfc3986) |
| URI-reference | [string](https://tools.ietf.org/html/rfc7159#section-7) following [RFC 3986](https://tools.ietf.org/html/rfc3986) |
| Timestamp | [string](https://tools.ietf.org/html/rfc7159#section-7) following [RFC 3339](https://www.ietf.org/rfc/rfc3339.txt) (ISO 8601) |
|  | |

### Event Data Model

|  |
| --- |
|  |

**Table 32: Top-Level JSON Schema**

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Constraint** | **Description** |
| id | String | rcv-only | Identifies the event. The Event Producer SHALL ensure that source + id is unique for each distinct event. |
| type | String | req | This attribute contains a value describing the type of event related to the originating occurrence. |
| source | URI-reference | rcv-only | Identifies the context in which an event happened. |
| specversion | String | rcv-only | The version of the CloudEvents specification which the event uses. This enables the interpretation of the context. |
| time | Timestamp | req | Time at which the event occurred. |
| data | String  (JSON array) | req | Array of JSON objects defining the information for the event |
| version | String | req | Version of the Notification API Schema generating the event.  ‘1.0’ until a future revision. |
| values | String  (JSON array) | req | A json array of values defining the event. |
|  | | | |
|  | | | |

**Table 35: Data Array Object Schema**

|  |  |  |
| --- | --- | --- |
| **Property** | **Type** | **Description** |
| data\_type | String | Type of value object. ( **notification** | **metric)** |
| ResourceAddress | String  (path) | See Table 2 |
| value\_type | Enumeration | The type format of the *value* property () |
| value | String | String representation of value in value\_type format |
| Table 34 shows an example event that contains Sync-State information.  **Table 34: Example Event -- Sync-State**   |  | | --- | | {  "id": "A234-1234-1234",  "specversion": "1.0",  "source": "/sync/sync-status/sync-state",  "type": "event.sync.sync-status.synchronization-state-change",  "time": "2021-03-05T20:59:00.999999999Z",  "data": {  "version": "1.0",  "values": [  {  "data\_type": "notification",  "ResourceAddress": "/east-edge-10/Node3/sync/sync-status/sync-state",  "value\_type": "enumeration",  "value": "HOLDOVER"  }  ]  }  } | | | |

### Synchronization Event Specifications

The following sections define the events related to synchronization events.

Editor's Note: synchronization state change events are addressed first due to priority of the RAN use cases, the event distribution infrastructure and associated interfaces are not limited to one specific event category, and events from other subsystems will be added in the future versions of this document.

Editor's Note: the present event set is aligned with / based on the WG4/WG5 YANG models; however, use of some other definitions such as composite clock modes in G.8275 (10/2020), Appendix VIII (or composite of the two approaches) \*may\* be more useful to convey the information in detail required to adequately specify the states in the cloud nodes context.

#### Synchronization State

This notification abstracts the underlying technology that the node is using to synchronize itself. It provides the overall synchronization health of the node. This notification includes the health of the OS System Clock which is consumable by application(s).

**Table 36: Synchronization State Notification**

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Description** |
| type | event.sync.sync-status.synchronization-state-change | Notification used to inform about the overall synchronization state change |
| source | /sync/sync-status/sync-state | Overall synchronization health of the node, including the OS System Clock |
| value\_type | enumeration |  |
| value | LOCKED | Equipment is in the locked mode, as defined in ITU-T G.810 |
| HOLDOVER | Equipment clock is in holdover mode, as defined in ITU-T G.810 |
| FREERUN | Equipment clock isn't locked to an input reference, and is not in the holdover mode, as defined in ITU-T G.810 |

#### PTP Synchronization State

**Table 37: Synchronization State Notification**

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Description** |
| type | event.sync.ptp-status.ptp-state-change | Notification used to inform about ptp synchronization state change |
| source | /sync/ptp-status/lock-state | ptp-state-change notification is signalled from equipment at state change |
| value\_type | enumeration |  |
| value | LOCKED | Equipment is in the locked mode, as defined in ITU-T G.810 |
| HOLDOVER | Equipment clock is in holdover mode, as defined in ITU-T G.810 |
| FREERUN | Equipment clock isn't locked to an input reference, and is not in the holdover mode, as defined in ITU-T G.810 |

#### Void

#### Void

#### GNSS-Sync-State

**Table 40: GNSS-Sync-State Notification**

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Description** |
| type | event.sync.gnss-status.gnss-state-change | Notification used to inform about gnss synchronization state change |
| source | /sync/gnss-status/gnss-sync-status | gnss-state-change notification is signalled from equipment at state change |
| value\_type | enumeration |  |
| value | SYNCHRONIZED | GNSS functionality is synchronized |
| ACQUIRING-SYNC | GNSS functionality is acquiring sync |
| ANTENNA-DISCONNECTED | GNSS functionality has its antenna disconnected |
| BOOTING | GNSS functionality is booting |
| ANTENNA-SHORT-CIRCUIT | GNSS functionality has an antenna short circuit |
| FAILURE-MULTIPATH | GNSS Sync Failure - Multipath condition detected |
| FAILURE-NOFIX | GNSS Sync Failure - Unknown |
| FAILURE-LOW-SNR | GNSS Sync Failure - Low SNR condition detected |
| FAILURE-PLL | GNSS Sync Failure - PLL is not functioning |

#### Void

#### OS Clock Sync-State

**Table 37: OS clock Sync-State Notification**

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Description** |
| type | event.sync.sync-status.os-clock-sync-state-change | The object contains information related to a notification |
| source | /sync/sync-status/os-clock-sync-state | State of node OS clock synchronization is notified at state change |
| value\_type | enumeration |  |
| value | LOCKED | Operating System real-time clock is in the locked mode, node operating system clock is synchronized to traceable & valid time/phase source |
| HOLDOVER | Operating System real-time clock is in holdover mode |
| FREERUN | Operating System real-time clock isn't locked to an input reference, and is not in the holdover mode |

#### SyncE Lock-Status-Extended

This notification is a SyncE Lock-state notification that provides detail about the synce PLL states.

**Table 39: SyncE-Extended Lock-State Notification**

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Description** |
| **type** | event.sync.synce-status.synce-state-change | Notification used to inform about synce synchronization state change, enhanced state information |
| **source** | **/sync/synce-status/lock-state** | synce-state change notification is signalled from equipment at state change, enhanced information |
| **value\_type** | **enumeration** |  |
| **value** | **LOCKED** | **The integrated ordinary clock is synchronizing to the reference, recovered from SyncE signal** |
| **HOLDOVER** | **The integrated ordinary clock is not synchronizing to the reference recovered from the SyncE signal, and is in holdover mode** |
| **FREERUN** | **The integrated ordinary clock is not synchronizing to the reference, recovered from SyncE signal** |
|  | | |

#### PTP Clock Class Change

A PTP Clock Class change notification is generated when the PTP clock change attribute in the Announce message changes.

**Table 36: PTP Clock class change Notification**

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Description** |
| type | event.sync.ptp-status.ptp-clock-class-change | Notification used to inform about ptp clock class changes. |
| source | /sync/ptp-status/clock-class | ptp-clock-class-change notification is generated when the clock-class changes. |
| value\_type | metric |  |
| value | Uint8 | New clock class attribute |

#### SyncE Clock Quality Change

A SyncE Clock Quality change notification is generated when the SyncE clock quality attribute in the ESMC message changes.

**Table 43: SyncE Clock class change Notification**

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Description** |
| type | event.sync.synce-status.synce-clock-quality-change | Notification used to inform about changes in the clock quality of the primary SyncE signal advertised in ESMC packets |
| source | /sync/synce-status/clock-quality | synce-clock-quality-change notification is generated when the clock-quality changes. |
| value\_type | metric |  |
| value | Uint8 | New clock quality attribute |

|  |
| --- |
|  |

## Appendix A

### Helper/Sidecar containers

Reference: <https://kubernetes.io/blog/2015/06/the-distributed-system-toolkit-patterns/>

Helper/Sidecar containers extend and enhance the "main" container, they take existing containers and make them better.

As an example, consider a container that runs the Nginx web server.  Add a different container that syncs the file system with a git repository, share the file system between the containers and one has built built Git push-to-deploy. And it has been done in a modular manner where the git synchronizer can be built by a different team, and can be reused across many different web servers (Apache, Python, Tomcat, etc).  Because of this modularity, the git synchronizer may be written and tested only once and reused across numerous apps.

Diagram

Description automatically generated

### Helper/Sidecar value:

* Interacts with the notification framework on behalf of the vO-DU
* Decouples the app logic from the notification framework, hence removes the burden of implementing a lot of code on the vO-DU and maintaining this code
* Single secure and reliable API endpoint since it is exposed over the localhost
* Eliminating the discovery of an external pod implementation

Annex (informative):   
Change History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision** | **Author** | **Description** |
| 05/10/2021 | 00.00.01 | Aaron Smith (RH)  Udi Schwager (WRS) | Initial skeleton. |
| 07/22/2021 | 01.00.00 | Kaustubh Joshi (AT&T) | Approved for publication. |
| 03/28/2022 | 02.00.00 | Padma Sudarsan (VMWare) | Incorporated 2 approved CRs (VMware, Wind River, RedHat, Altiostar) |
| 03/05/2022 | 02.00.01 | Udi Schwager (Wind River) | Ready for TSC review. |
| 07/25/2022 | 03.00.00 | Udi Schwager (Wind River) | Support for multiple event producers |
| 03/15/2024 | 03.00.01 | Udi Schwager (Wind River) | Incorporated Qualcomm CR |
| 03/21/2024 | 03.00.03 | Udi Schwager (Wind River) | Editorial updates |