CUCULUS

Affordable energy and water for everyone

ZONOS Streaming API

Cuculus GmbH

Version 53.0.1

www.cuculus.com

Table of Contents

1. Preface . . . 1
2. Overview . . . 2
   2.1. What is Apache Kafka? . . . 2
   2.2. Data organization in Kafka . . . 2
      2.2.1. Topics . . . 2
      2.2.2. Partitions . . . 2
      2.2.3. Streams . . . 3
      2.2.4. Key/value storage . . . 3
      2.2.5. Data replication . . . 3
      2.2.6. Log compaction . . . 3
   2.3. Changelog . . . 4
      2.3.1. ZONOS 53 . . . 4
      2.3.2. ZONOS 52 . . . 4
      2.3.3. ZONOS 51 . . . 4
3. Technical details . . . 5
   3.1. Data formats . . . 5
      3.1.1. Keys . . . 5
      3.1.2. Values . . . 5
   3.2. List of topics . . . 5
4. Resources . . . 7
   4.1. Incubating . . . 7
      4.1.1. GET /device-derived-measurement . . . 7
      4.1.2. GET /device-event-v2 . . . 7
      4.1.3. GET /device-process . . . 8
      4.1.4. GET /device-profile-facts . . . 9
      4.1.5. GET /group . . . 9
      4.1.6. GET /operational-process . . . 10
      4.1.7. GET /register-statistic . . . 11
   4.2. Stable . . . 11
      4.2.1. GET /device . . . 11
      4.2.2. GET /device-event . . . 12
      4.2.3. GET /device-measurement . . . 12
      4.2.4. GET /device-parameter . . . 13
      4.2.5. GET /device-telemetry . . . 14
      4.2.6. GET /device-topology . . . 15
      4.2.7. GET /metering-point . . . 15
      4.2.8. GET /metering-point-parameter . . . 16
5. Definitions . . . 17

5.1. ActivityCalendarConfiguration . . . 17
5.2. ActivityCalendarRead . . . 17
5.3. ActivityCalendarWrite . . . 17
5.4. Address . . . 17
5.5. AlarmCode . . . 18
5.6. AlarmRegisterRead . . . 18
5.7. AlarmRegisterReset . . . 18
5.8. CertificateExchange . . . 18
5.9. CertificateExchangeCertificateType . . . 19
5.10. ClearAlarm . . . 19
5.11. ClearCredit . . . 19
5.12. ConsumptionDataPoint . . . 19
5.13. ConsumptionDataPointQualityDetails . . . 20
5.14. ConsumptionDataPointQualityFaultyValueReason . . . 21
5.15. CosemAccess . . . 22
5.16. CosemAccessActionSpecification . . . 22
5.17. CosemAccessGetSpecification . . . 22
5.18. CosemAccessSelection . . . 22
5.19. CosemAccessSetSpecification . . . 23
5.20. CosemAccessSpecification . . . 23
5.21. CosemAccessStorageSpecification . . . 23
5.22. CosemAttribute . . . 24
5.23. CosemDate . . . 24
5.24. CosemDateTime . . . 24
5.25. CosemMethod . . . 24
5.26. CosemProfileConfiguration . . . 25
5.27. CosemProfileConfigurationCaptureObject . . . 25
5.28. CosemProfileConfigurationRead . . . 25
5.29. CosemProfileConfigurationWrite . . . 26
5.30. CosemTime . . . 26
5.31. CosemValue . . . 26
5.32. CustomerOptOutRead . . . 28
5.33. DataPoint . . . 28
5.34. DataSource . . . 28
5.35. DataSourceName . . . 28
5.36. DayProfile . . . 29
5.37. DayProfileAction . . . 29
5.38. Device . . . 29
5.39. DeviceAssignment . . . 30
5.40. DeviceCameOnline . . . 30
5.41. DeviceCommunicated . . . 30

5.42. DeviceDerivedMeasurement . . . 30
5.43. DeviceEvent . . . 31
5.44. DeviceEventV2 . . . 32
5.45. DeviceEventV2Alarm . . . 33
5.46. DeviceLocation . . . 33
5.47. DeviceMeasurement . . . 33
5.48. DeviceName . . . 35
5.49. DeviceParameter . . . 35
5.50. DeviceProcess . . . 35
5.51. DeviceProcessExecutionType . . . 37
5.52. DeviceProcessParameters . . . 37
5.53. DeviceProcessState . . . 39
5.54. DeviceProfileFacts . . . 40
5.55. DeviceRegister . . . 40
5.56. DeviceState . . . 41
5.57. DeviceStatusFlag . . . 42
5.58. DeviceTag . . . 42
5.59. DeviceTelemetry . . . 42
5.60. DeviceTopology . . . 43
5.61. DeviceWentOffline . . . 43
5.62. DisconnectorState . . . 43
5.63. DisplayConfiguration . . . 43
5.64. DisplayConfigurationRead . . . 44
5.65. DisplayConfigurationWrite . . . 44
5.66. EditReason . . . 44
5.67. Editor . . . 44
5.68. EngineeringTokenTransfer . . . 44
5.69. EventCode . . . 45
5.70. EventState . . . 45
5.71. FirmwareUpdate . . . 45
5.72. FirmwareUpdateOverTheAir . . . 45
5.73. GenericAction . . . 46
5.74. GeoLocation . . . 46
5.75. GlobalKeyExchange . . . 46
5.76. Group . . . 46
5.77. GroupId . . . 47
5.78. GroupState . . . 47
5.79. InventoryState . . . 47
5.80. KeyExchange . . . 47
5.81. KeyExchangeKeyType . . . 48
5.82. KeyExchangeMechanism . . . 48

5.83. KeyPairGeneration . . . 48
5.84. Location . . . 48
5.85. LogRead . . . 48
5.86. ManagementState . . . 48
5.87. MbusKeyExchange . . . 48
5.88. MeasureTime . . . 49
5.89. MeteringPoint . . . 49
5.90. MeteringPointIdentifier . . . 49
5.91. MeteringPointParameter . . . 49
5.92. MeteringPointState . . . 50
5.93. MeteringPointStateEnum . . . 51
5.94. MeteringPointStateHistoryEntry . . . 51
5.95. OnDemandRead . . . 51
5.96. OperationalProcess . . . 51
5.97. OperationalProcessState . . . 52
5.98. P0PortRead . . . 52
5.99. P0PortWrite . . . 52
5.100. P1PortRead . . . 52
5.101. P1PortWrite . . . 53
5.102. ParameterName . . . 53
5.103. PaymentMode . . . 53
5.104. PersistTime . . . 53
5.105. ProcessStatus . . . 53
5.106. ProfileName . . . 53
5.107. ProfileRead . . . 53
5.108. ProfileRecordingControlRead . . . 54
5.109. ProfileRecordingControlWrite . . . 54
5.110. PushSetupConfigurationRead . . . 54
5.111. PushSetupConfigurationWrite . . . 54
5.112. Quality . . . 54
5.113. ReceiveTime . . . 54
5.114. Register . . . 54
5.115. RegisterMetadataRead . . . 55
5.116. RegisterName . . . 55
5.117. RegisterStatistic . . . 55
5.118. RegisterValue . . . 56
5.119. SeasonProfile . . . 57
5.120. SecretExchange . . . 57
5.121. ServerType . . . 57
5.122. SetDisconnectorState . . . 57
5.123. SetLoadLimitation . . . 57

5.124. SetPaymentMode . . . 58
5.125. SupplierAssignment . . . 58
5.126. TokenTransfer . . . 58
5.127. TopologyChild . . . 58
5.128. UpdateAction . . . 58
5.129. WeekProfile . . . 59
6. Appendix . . . 60
6.1. Incubating examples . . . 60
6.1.1. GET /register-statistic . . . 60
6.1.2. GET /device-derived-measurement . . . 60
6.1.3. GET /device-process . . . 61
6.2. Stable examples . . . 61
6.2.1. GET /device . . . 62
6.2.2. GET /device-event . . . 63
6.2.3. GET /device-measurement . . . 64
6.2.4. GET /device-parameter . . . 64
6.2.5. GET /device-telemetry . . . 64
6.2.6. GET /device-topology . . . 65
6.2.7. GET /metering-point . . . 65
6.2.8. GET /metering-point-parameter . . . 67
6.2.9. GET /group . . . 67
6.3. Previous changelogs . . . 68
6.3.1. ZONOS 50 . . . 68
6.3.2. ZONOS 49 . . . 68
6.3.3. ZONOS 48 . . . 68
6.3.4. ZONOS 46 . . . 68
6.3.5. ZONOS 42 . . . 68
6.3.6. ZONOS 33 . . . 69
6.3.7. ZONOS 32 . . . 69
6.3.8. ZONOS 31 . . . 69
6.3.9. ZONOS 28 . . . 69
6.3.10. ZONOS 27 . . . 69
6.3.11. ZONOS 24 . . . 70
6.3.12. ZONOS 17 . . . 70

Chapter 1. Preface

WARNING

The information compiled in this document is preliminary and may change without prior notice. If you develop an application consuming this API please be prepared to change it as the development of ZONOS continues. With the next major ZONOS release we will provide a stable version of this API.

We will document all breaking changes in a change log section. Breaking changes are considered to be:

- Removing/renaming a topic
- Changing the key of messages on a topic
- Removing/renaming a property in a data model
- Changing the value semantics of a data model property

Chapter 2. Overview

The ZONOS Streaming API provides a flexible and reliable way to retrieve, transform and analyse data stored and processed by ZONOS. This allows developers to easily integrate ZONOS with other applications. The API uses Apache Kafka to stream data out of ZONOS, making it into a data source.

2.1. What is Apache Kafka?

Kafka is an open-source clustered stream processing software platform. It is highly scalable, fault tolerant, provides transactional read/write semantics and different interfaces to read and write or process the data in the cluster. Interactions with Kafka usually follow the producer/consumer pattern and this terms are used throughout its documentation. Data producers (i.e. ZONOS) and consumers (i.e. third party client applications) are decoupled by Kafka and thus do not interact directly. This solves the problem where data can otherwise not be emitted from a system if the respective data consumer might not be available. Further, by acting as a buffer between readers and writers data may be buffered to prevent swamping a consumer if its processing can not keep up with the data writers speed. Data committed to the cluster is stored persistently and retained for a configurable amount of time before the oldest messages are dropped.

2.2. Data organization in Kafka

Apache Kafka organizes data in several ways for achieving its ease of use, reliability and performance.

2.2.1. Topics

Topics group related data in the sense that producers/consumers always know what the conversation is about. Usually a topic carries a stream of a single data type. This makes it easy for consumers to interpret the data. All topics used by an application are created in the cluster beforehand and are identified by name, somewhat similar to how RDBMS tables are explicitly created before they can be used. Several creation time parameters govern how Kafka handles the data on a particular topic with regard to partitioning and replication throughout the cluster.

2.2.2. Partitions

Partitions divide the data to provide for scalability of the system. Partitioning occurs per topic. The key of each message produced onto a topic will determine its partition. Messages with the same key will always end up in the same partition.

Partitioning is especially interesting for consumer applications. Each consumer joins a `consumer group`. Kafka assigns partitions for data consumption to each consumer in the group based on the number of group members. Each consumer group is guaranteed to receive the total of the messages on a topic. Thus, if a consuming application's processing performance is insufficient to handle the total number of messages on a topic, a second instance within the same consumer group can be started to mitigate the performance bottleneck.

2.2.3. Streams

Kafka provides a view of all data as a stream of messages. New data is appended to the end of a topic and usually consumed from the front of the stream. Together with the transactional consumption semantics an application can always be sure to process all data in the stream — even if drops out for a short time. Producing applications share the same benefit as they can be sure the data is safely stored in the cluster when a transaction commits.

Kafka's data partitioning complicates the streaming view, in that it only applies to each individual partition within a topic. Only within a partition is the order of messages as they were produced a guarantee provided by the system. The reason behind this becomes clear when imagining the scenario of a failing consumer. For example, if the topic of natural numbers N is divided into two partitions, with each even number going to one partition and each odd number going to the other, and numbers are produced in sequence, Kafka will guarantee that each message on partition one will carry a higher number than the one before it. The same will be true for partition two. However, there will be no guarantee that number '42' is consumed before number '43' as this crosses partition boundaries. Therefore, each topic partition acts as a log of messages.

2.2.4. Key/value storage

Each message in Kafka consists of a key and a value (i. e. the message contents) of arbitrary data types. A Kafka provided and user extensible serialization framework ensures that message keys and values are transformed into sequences of bytes. Applications may agree to share keys between topics for cross-topic correlation purposes. Message keys are provided by the producing application and in their serialized form used by Kafka to partition the data within the cluster and present it to consumers. They may or may not be related to the message value and no semantics should be ascribed to them except as a means for correlation. However, applications usually expect that keys will not be random, that is, repeating messages about "the same" data entity should carry the same key so that consumers can correlate messages within each stream. Therefore, usually some identifying properties of the message are used to compute its key.

2.2.5. Data replication

Data streamed into the cluster can be replicated onto different cluster nodes to provide resilience against failure of individual cluster nodes. When each topic is created, its replication factor is provided as parameter.

Parameters should be chosen based on the cluster size and expected data volume. Of course, storage requirements will increase on each cluster with higher replication factors. Kafka can ensure that producer transactions will only commit after the data has safely been replicated to a specified minimum number of nodes; thus potentially increasing latency when producing.

2.2.6. Log compaction

Log (or topic) compaction ensures that Kafka will always retain at least the last known value for each message key within the log of data for a single topic partition. Usually data is dropped after the configured retention period if it is not consumed. Compaction provides the means to at least guarantee the most recent message for a particular key is always available for future consumption.

It addresses use cases and scenarios such as restoring state after application crashes or system failures, or reloading caches after application restarts during operational maintenance.

2.3. Changelog

2.3.1. ZONOS 53

- The property `changeTime` has been added to the topic `ext_metering-point-parameter_{instance_name}`. It contains the point in time, when the change became effective.

2.3.2. ZONOS 52

- The property `identifiers` has been added to the topic `ext_device_{instance_name}`. It contains a map of additional device identifiers.

2.3.3. ZONOS 51

- The topic `ext_metering-point_{instance_name}` has been extended to include information about the supplier history and registers, including their data source history.

Previous changelogs

Chapter 3. Technical details

The ZONOS Streaming API can be consumed with any available Kafka client software. Please contact the responsible system administrator for connection and client configuration details.

3.1. Data formats

3.1.1. Keys

Keys on any topic published by ZONOS are opaque identifiers. They have no meaning by themselves and can only be used for correlating messages.

3.1.2. Values

All values produced by the API use JSON as a data representation. This choice makes it easy to consume the API from nearly any language and environment due to the popularity of the format. Further, JSON data models can easily be extended without breaking expectations of older client software.

3.2. List of topics

The following list contains the names of all currently available topics. Please note that due to the fact that several ZONOS installations may share the same deployment environment and therefore the same Kafka cluster, each topic name is suffixed by an 'instance_name'. Further, all published topics carry a prefix of 'ext' to distinguish them from potentially present topics that are internal to ZONOS.

Whenever a topic is referenced throughout this document by its short name, please be aware of the prefix and suffix to construct the complete topic name.

- ext_device-event_{instance_name}
- ext_device-event-v2_{instance_name}
- ext_device-measurement_{instance_name}
- ext_device-parameter_{instance_name}
- ext_device-telemetry_{instance_name} (compacted)
- ext_device-topology_{instance_name} (compacted)
- ext_device_{instance_name} (compacted)
- ext_register-statistic_{instance_name}
- ext_metering-point_{instance_name} (compacted)
- ext_device-process_{instance_name}
- ext_operational-process_{instance_name}
- ext_device-derived-measurement_{instance_name}

- ext_device-profile-facts_{instance_name}
- ext_group_{instance_name} (compacted)

The following chapters describe the topics and data available on them in greater detail.

IMPORTANT	Please be aware that Kafka topics in this document are described in terms of HTTP resources. This is due to the fact that we use OpenAPI a.k.a. Swagger tools to specify the API and automatically generate most of this document from that specification. The OpenAPI tools only support HTTP as a transport.

Chapter 4. Resources

4.1. Incubating

Still under development - subject to change.

4.1.1. GET /device-derived-measurement

Responses

HTTP Code	Description	Schema
200	Stream of device derived measurements	DeviceDerivedMeasurement

Produces

- `application/json`

Example HTTP request

Request path

```
/device-derived-measurement
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.1.2. GET /device-event-v2

Responses

HTTP Code	Description	Schema
200	Stream of device event context messages	DeviceEventV2

Produces

- `application/json`

Example HTTP request

Request path

```
/device-event-v2
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.1.3. GET /device-process

Description

Stream of device process entities.

Responses

HTTP Code	Description	Schema
200	Stream of device process entities	DeviceProcess

Produces

- `application/json`

Example HTTP request

Request path

```
/device-process
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.1.4. GET /device-profile-facts

Description

Stream of device profile facts.

Responses

HTTP Code	Description	Schema
200	Stream of device profile facts	DeviceProfileFacts

Produces

- `application/json`

Example HTTP request

Request path

```
/device-profile-facts
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.1.5. GET /group

Responses

HTTP Code	Description	Schema
200	Stream of group entities	Group

Produces

- `application/json`

Example HTTP request

Request path

```
/group
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.1.6. GET /operational-process

Description

Stream of operational process entities.

Responses

HTTP Code	Description	Schema
200	Stream of operational process entities	OperationalProcess

Produces

- `application/json`

Example HTTP request

Request path

```
/operational-process
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.1.7. GET /register-statistic

Responses

HTTP Code	Description	Schema
200	Stream of register statistic messages	RegisterStatistic

Produces

- `application/json`

Example HTTP request

Request path

```
/register-statistic
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.2. Stable

Fields might be renamed but the general structure is fixed.

4.2.1. GET /device

Responses

HTTP Code	Description	Schema
200	Stream of device entities	Device

Produces

- `application/json`

Example HTTP request

Request path

```
/device
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.2.2. GET /device-event

Responses

HTTP Code	Description	Schema
200	Stream of device event messages	DeviceEvent

Produces

- `application/json`

Example HTTP request

Request path

```
/device-event
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.2.3. GET /device-measurement

Responses

HTTP Code	Description	Schema
200	Stream of device measurement messages	DeviceMeasurement

Produces

- application/json

Example HTTP request

Request path

```
/device-measurement
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.2.4. GET /device-parameter

Description

Stream of device parameter entities. For parameter changes on ZONOS Control Panel or ZONOS API, obfuscated parameters like passwords and keys will not be exposed.

Responses

HTTP Code	Description	Schema
200	Stream of device parameter entities	DeviceParameter

Produces

- application/json

Example HTTP request

Request path

```
/device-parameter
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.2.5. GET /device-telemetry

Description

Stream of device telemetry events. Provides information about device communications, f.e. when did the last communication via an online data source happen?

Responses

HTTP Code	Description	Schema
200	Stream of device telemetry events	DeviceTelemetry

Produces

- `application/json`

Example HTTP request

Request path

```
/device-telemetry
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.2.6. GET /device-topology

Responses

HTTP Code	Description	Schema
200	Stream of device topology entities	DeviceTopology

Produces

- `application/json`

Example HTTP request

Request path

```
/device-topology
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.2.7. GET /metering-point

Responses

HTTP Code	Description	Schema
200	Stream of metering point entities	MeteringPoint

Produces

- `application/json`

Example HTTP request

Request path

```
/metering-point
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

4.2.8. GET /metering-point-parameter

Description

Stream of metering point parameter entities. For parameter changes on ZONOS Control Panel or ZONOS API, obfuscated parameters like passwords and keys will not be exposed.

Responses

HTTP Code	Description	Schema
200	Stream of metering point parameter entities	MeteringPointParameter

Produces

- `application/json`

Example HTTP request

Request path

```
/metering-point-parameter
```

Example HTTP response

Response 200

```
{
  "Key" : "object",
  "Value" : "object"
}
```

Chapter 5. Definitions

5.1. ActivityCalendarConfiguration

Name	Description	Schema
dayProfiles *required*	**Example** : `[ "DayProfile" ]`	< DayProfile > array
name *required*	**Example** : `"string"`	string
seasonProfiles *required*	**Example** : `[ "SeasonProfile" ]`	< SeasonProfile > array
weekProfiles *required*	**Example** : `[ "WeekProfile" ]`	< WeekProfile > array

5.2. ActivityCalendarRead

Name	Description	Schema
activityCalendarId *optional*	**Example** : `"string"`	string

5.3. ActivityCalendarWrite

Name	Description	Schema
activateAt *optional*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
activityCalendarConfiguration *optional*	**Example** : ActivityCalendarConfiguration	ActivityCalendarConfiguration
activityCalendarId *optional*	**Example** : `"string"`	string

5.4. Address

An address as usable for mailing

Name	Description	Schema
city *optional*	**Example** : `"Erfurt"`	string
company *optional*	**Example** : `"ACME"`	string
country *optional*	**Example** : `"DE"`	string
district *optional*	**Example** : `"Loebervorstadt"`	string
floor *optional*	**Example** : `"2"`	string
houseNumber *optional*	**Example** : `"1b"`	string
postalCode *optional*	**Example** : `"99096"`	string
reference *optional*	**Example** : `"X24M+QJ Erfurt"`	string
region *optional*	**Example** : `"Thuringia"`	string
street *optional*	**Example** : `"Arndtstraße"`	string
timeZone *optional*	**Example** : `"Europe/Berlin"`	string

5.5. AlarmCode

Unique identifier for an event in ZONOS

Type : string

5.6. AlarmRegisterRead

Type : object

5.7. AlarmRegisterReset

Type : object

5.8. CertificateExchange

Name	Description	Schema
certificate *required*	**Example** : `"string"`	string
certificateType *required*	**Example** : CertificateExchangeCertificateType	CertificateExchangeCertificateType
match *required*	**Example** : `"object"`	match

match

Name	Description	Schema
issuer *required*	**Example** : `"string"`	string
serialNumber *required*	**Example** : `"string"`	string

5.9. CertificateExchangeCertificateType

Type : enum (sub_ca, digital_signature)

5.10. ClearAlarm

Type : object

5.11. ClearCredit

Type : object

5.12. ConsumptionDataPoint

Name	Description	Schema
quality *required*	**Example** : Quality	Quality
qualityDetails *optional*	**Example** : ConsumptionDataPointQualityDetails	ConsumptionDataPointQualityDetails
time *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
value *required*	**Example** : `42.1337`	number

5.13. ConsumptionDataPointQualityDetails

Details on the quality of the consumption data point. Only one property may be set.

Name	Description	Schema
faultyValue *optional*	**Example** : `"object"`	faultyValue
forecastValue *optional*	**Example** : `"object"`	forecastValue
manualValue *optional*	**Example** : `"object"`	object
replacementValue *optional*	**Example** : `"object"`	replacementValue
trueValue *optional*	**Example** : `"object"`	object

faultyValue

Name	Description	Schema
reasons *optional*	**Example** : `[ "ConsumptionDataPointQualityFaultyValueReason" ]`	< ConsumptionDataPointQualityFaultyValueReason > array

forecastValue

Name	Description	Schema
forecastRule *required*	Forecast rule that produced this value **Example** : `"object"`	forecastRule

forecastRule

Name	Description	Schema
id *required*	ID of the forecast rule **Example** : `0`	integer
name *required*	Name of the forecast rule **Example** : `"string"`	string

replacementValue

Name	Description	Schema
estimationRule *required*	Estimation rule that produced this value **Example** : `"object"`	estimationRule

estimationRule

Name	Description	Schema
id *required*	ID of the estimation rule **Example** : `0`	integer
name *required*	Name of the estimation rule **Example** : `"string"`	string

5.14. ConsumptionDataPointQualityFaultyValueReason

Name	Description	Schema
invalidStatusWord *optional*	**Example** : `"object"`	invalidStatusWord
validationFailed *optional*	**Example** : `"object"`	validationFailed

invalidStatusWord

Name	Description	Schema
statusWord *required*	The value of the status word **Example** : `0`	integer

validationFailed

Name	Description	Schema
note *optional*	Note containing human-readable details on the validation failure **Example** : `"string"`	string
validationRule *required*	Validation rule that failed **Example** : `"object"`	validationRule

validationRule

Name	Description	Schema
id *required*	ID of the failing validation rule **Example** : `0`	integer
name *required*	Name of the failing validation rule **Example** : `"string"`	string

5.15. CosemAccess

Name	Description	Schema
specifications *optional*	**Example** : `[ "CosemAccessSpecification" ]`	< CosemAccessSpecification > array

5.16. CosemAccessActionSpecification

Specification for invoking a method

Name	Description	Schema
method *required*	**Example** : CosemMethod	CosemMethod
parameters *required*	**Example** : CosemValue	CosemValue
storage *required*	**Example** : CosemAccessStorageSpecification	CosemAccessStorageSpecification

5.17. CosemAccessGetSpecification

Specification for getting an attribute

Name	Description	Schema
accessSelection *optional*	**Example** : CosemAccessSelection	CosemAccessSelection
attribute *required*	**Example** : CosemAttribute	CosemAttribute
storage *optional*	**Example** : CosemAccessStorageSpecification	CosemAccessStorageSpecification

5.18. CosemAccessSelection

Speicification of the part of the attribute to get or set

Name	Description	Schema
dataIndex *optional*	Index of the element of a structured attribute **Example** : `0`	integer

5.19. CosemAccessSetSpecification

Specification for setting an attribute

Name	Description	Schema
accessSelection *required*	**Example** : CosemAccessSelection	CosemAccessSelection
attribute *required*	**Example** : CosemAttribute	CosemAttribute
storage *required*	**Example** : CosemAccessStorageSpecification	CosemAccessStorageSpecification
value *required*	**Example** : CosemValue	CosemValue

5.20. CosemAccessSpecification

Specification of the COSEM access. Either get, set or action is set.

Name	Description	Schema
action *optional*	**Example** : CosemAccessActionSpecification	CosemAccessActionSpecification
get *optional*	**Example** : CosemAccessGetSpecification	CosemAccessGetSpecification
set *optional*	**Example** : CosemAccessSetSpecification	CosemAccessSetSpecification

5.21. CosemAccessStorageSpecification

Storage specification for the result. The specification may include a device where to store the result and a parameter to store the result in.

Name	Description	Schema
device *optional*	**Example** : DeviceName	DeviceName
deviceParameter *optional*	Name of the parameter to store the result in **Example** : `"tcp.inactivity_timeout"`	string

5.22. CosemAttribute

Attribute of a COSEM object

Name	Description	Schema
attributeId *required*	ID of the attribute **Example** : `2`	integer
classId *required*	Interface class ID of the object **Example** : `3`	integer
instanceId *required*	Instance ID of the object **Example** : `"1-0:1.8.0*255"`	string

5.23. CosemDate

Name	Description	Schema
dayOfMonth *required*	**Example** : `0`	integer
dayOfWeek *required*	**Example** : `0`	integer
month *required*	**Example** : `0`	integer
year *required*	**Example** : `0`	integer

5.24. CosemDateTime

Name	Description	Schema
date *required*	**Example** : CosemDate	CosemDate
deviation *required*	**Example** : `0`	integer
status *required*	**Example** : `0`	integer
time *required*	**Example** : CosemTime	CosemTime

5.25. CosemMethod

Method of a COSEM object

Name	Description	Schema
classId *required*	Interface class ID of the object **Example** : `3`	integer
instanceId *required*	Instance ID of the object **Example** : `"1-0:1.8.0*255"`	string
methodId *required*	ID of the method **Example** : `1`	integer

5.26. CosemProfileConfiguration

Name	Description	Schema
capturePeriod *optional*	Capture period of the profile in seconds **Example** : `900.0`	number
profileConfiguration *optional*	**Example** : CosemProfileConfiguration	CosemProfileConfiguration
profileEntries *optional*	Number of available entries in the profile **Example** : `1000.0`	number
registers *optional*	Registers captured in the profile **Example** : `[ "CosemProfileConfigurationCaptureObject" ]`	< CosemProfileConfigurationCaptureObject > array

5.27. CosemProfileConfigurationCaptureObject

Name	Description	Schema
attributeIndex *required*	Index of the attribute to be captured **Example** : `2.0`	number
classId *required*	Interface class ID of the capture object **Example** : `3.0`	number
dataIndex *required*	Data index **Example** : `0.0`	number
logicalName *required*	Logical name of the capture object **Example** : `"1-0:1.8.0*255"`	string

5.28. CosemProfileConfigurationRead

Name	Description	Schema
profileLogicalName *required*	The logical name of the profile for which the configuration shall be read **Example** : `"1-0:99.1.0*255"`	string

5.29. CosemProfileConfigurationWrite

Name	Description	Schema
profileConfiguration *required*	**Example** : CosemProfileConfiguration	CosemProfileConfiguration
profileLogicalName *required*	The logical name of the profile for which the configuration shall be written **Example** : `"1-0:99.1.0*255"`	string

5.30. CosemTime

Name	Description	Schema
hour *required*	**Example** : `0`	integer
hundredths *required*	**Example** : `0`	integer
minute *required*	**Example** : `0`	integer
second *required*	**Example** : `0`	integer

5.31. CosemValue

COSEM value. Only one property may be set. If no property is set, the value represents `null-data`.

Name	Description	Schema
array *optional*	**Example** : `[ "CosemValue" ]`	< CosemValue > array
bcd *optional*	**Example** : `"string"`	string
bitString *optional*	**Example** : `"string"`	string
boolean *optional*	**Example** : `true`	boolean

Name	Description	Schema
compactArray *optional*	**Example** : `[ "CosemValue" ]`	< CosemValue > array
date *optional*	**Example** : CosemDate	CosemDate
dateTime *optional*	**Example** : CosemDateTime	CosemDateTime
doubleLong *optional*	**Example** : `0`	integer
doubleLongUnsigned *optional*	**Example** : `0`	integer (int64)
enum *optional*	**Example** : `0`	integer
float32 *optional*	**Example** : `0.0`	number (float)
float64 *optional*	**Example** : `0.0`	number (double)
integer *optional*	**Example** : `0`	integer
long *optional*	**Example** : `0`	integer
long64 *optional*	**Example** : `0`	integer (int64)
long64Unsigned *optional*	**Example** : `"string"`	string
longUnsigned *optional*	**Example** : `0`	integer
octetString *optional*	**Pattern** : `"^(?:[A-Za-z0-9+/]{4})*(?:[A-Za-z0-9+/]{2}==|[A-Za-z0-9+/]{3}=)?$"` **Example** : `"Ynl0ZQ=="`	string (byte)
structure *optional*	**Example** : `[ "CosemValue" ]`	< CosemValue > array
time *optional*	**Example** : CosemTime	CosemTime
unsigned *optional*	**Example** : `0`	integer
utf8String *optional*	**Example** : `"string"`	string

Name	Description	Schema
visibleString *optional*	**Example** : `"string"`	string

5.32. CustomerOptOutRead

Type : object

5.33. DataPoint

Name	Description	Schema
measureTime *optional*	**Example** : MeasureTime	MeasureTime
register *required*	**Example** : RegisterName	RegisterName
unit *required*	**Example** : `"kWh"`	string
value *required*	**Example** : `42.1337`	number

5.34. DataSource

Data source information

Name	Description	Schema
activeSince *optional*	Point in time since when the state was active **Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
activeUntil *optional*	Point in time until when the state was active **Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
deviceRegister *optional*	**Example** : DeviceRegister	DeviceRegister
id *optional*	**Example** : `"bbca72a6-ef6a-4ecd-87f9-6a05647b6e67"`	string (uuid)

5.35. DataSourceName

Unique identifier for a data source

Type : string

5.36. DayProfile

Name	Description	Schema
actions *required*	**Example** : `[ "DayProfileAction" ]`	< DayProfileAction > array
dayId *required*	**Example** : `0`	integer

5.37. DayProfileAction

Name	Description	Schema
scriptLogicalName *required*	**Example** : `"string"`	string
scriptSelector *required*	**Example** : `0`	integer
startAt *required*	**Example** : `"string"`	string

5.38. Device

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *optional*	**Example** : `"object"`	Value

Key

Name	Description	Schema
device *required*	**Example** : DeviceName	DeviceName

Value

Name	Description	Schema
current *optional*	**Example** : DeviceState	DeviceState
persistTime *required*	**Example** : PersistTime	PersistTime
previous *optional*	**Example** : DeviceState	DeviceState

5.39. DeviceAssignment

Name	Description	Schema
assignTime *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
device *required*	**Example** : DeviceName	DeviceName
removeTime *optional*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)

5.40. DeviceCameOnline

Name	Description	Schema
cameOnlineAt *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)

5.41. DeviceCommunicated

Name	Description	Schema
communicatedAt *optional*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)

5.42. DeviceDerivedMeasurement

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *required*	**Example** : `"object"`	Value

Key

Name	Description	Schema
device *required*	**Example** : DeviceName	DeviceName

Value

Name	Description	Schema
consumptionDataPoints *optional*	**Example** : `[ "ConsumptionDataPoint" ]`	< ConsumptionDataPoint > array
device *required*	**Example** : DeviceName	DeviceName
persistTime *optional*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
register *required*	**Example** : RegisterName	RegisterName
unit *optional*	**Example** : `"kWh"`	string

5.43. DeviceEvent

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *required*	**Example** : `"object"`	Value

Key

Name	Description	Schema
device *required*	**Example** : DeviceName	DeviceName

Value

Name	Description	Schema
context *required*	**Example** : `{ "1" : "HXEE81010001", "2" : "2018-12-31T23:59:59.123Z" }`	< string, string > map
device *required*	**Example** : DeviceName	DeviceName
event *required*	**Example** : EventCode	EventCode
firstOccurrenceTime *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
lastOccurrenceTime *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)

Name	Description	Schema
occurrenceCount *required*	**Minimum value** : 1 **Example** : 4	integer
persistTime *required*	**Example** : PersistTime	PersistTime
receiveTime *required*	**Example** : ReceiveTime	ReceiveTime
state *required*	**Example** : EventState	EventState

5.44. DeviceEventV2

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *required*	**Example** : `"object"`	Value

Key

Name	Description	Schema
device *required*	**Example** : DeviceName	DeviceName

Value

Name	Description	Schema
alarms *optional*	List of ZONOS alarms created for this event **Example** : `[ "DeviceEventV2Alarm" ]`	< DeviceEventV2Alarm > array
device *required*	**Example** : DeviceName	DeviceName
eventCode *required*	Event code **Example** : `"string"`	string
logId *required*	External identifier of the log **Example** : `"0-0:99.98.0*255"`	string
occurredAt *required*	The time the event occurred **Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
payload *optional*	**Example** : `"object"`	payload

payload

Name	Description	Schema
cosemValues *optional*	**Example** : `[ "CosemValue" ]`	< CosemValue > array
registerValues *optional*	**Example** : `[ "RegisterValue" ]`	< RegisterValue > array

5.45. DeviceEventV2Alarm

Name	Description	Schema
alarmCode *optional*	**Example** : AlarmCode	AlarmCode
firstOccurredAt *optional*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)

5.46. DeviceLocation

Name	Description	Schema
address *optional*	**Example** : Address	Address
geo *optional*	**Example** : GeoLocation	GeoLocation
logicalInstallationPoint *optional*	**Example** : `"FB4RL2"`	string

5.47. DeviceMeasurement

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *optional*	**Example** : `"object"`	Value

Key

Name	Description	Schema
device *required*	**Example** : DeviceName	DeviceName

Value

Name	Description	Schema
correlationId *optional*	Optional correlation ID **Example** : `"bbca72a6-ef6a-4ecd-87f9-6a05647b6e67"`	string
dataPoints *required*	**Example** : `[ "DataPoint" ]`	< DataPoint > array
dataSource *optional*	**Example** : DataSourceName	DataSourceName
device *required*	**Example** : DeviceName	DeviceName
edit *optional*	Edit information for this measurement **Example** : `"object"`	edit
measureTime *required*	Point in time since this value was measured by the meter **Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
persistTime *required*	**Example** : PersistTime	PersistTime
profile *optional*	**Example** : ProfileName	ProfileName
readingReason *required*	Reading reason code **Example** : `"1"`	string
receiveTime *required*	**Example** : ReceiveTime	ReceiveTime
revision *optional*	Revision of this measurement **Example** : `0`	integer
status *required*	Status of the meter at the time of measurement. The status consists of several status flags. **Example** : `[ "DeviceStatusFlag" ]`	< DeviceStatusFlag > array
tags *required*	Metered data tags for this measurement **Example** : `{ "string" : "string" }`	< string, string > map

edit

Name	Description	Schema
editedBy *optional*	**Example** : Editor	Editor
note *optional*	**Example** : EditReason	EditReason

5.48. DeviceName

Unique identifier for a device in ZONOS

Type : string

5.49. DeviceParameter

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *required*	**Example** : `"object"`	Value

Key

Name	Description	Schema
device *required*	**Example** : DeviceName	DeviceName

Value

Name	Description	Schema
changeTime *required*	Point in time at which the change became effective **Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
device *required*	**Example** : DeviceName	DeviceName
newValue *required*	**Example** : `"ready to connect"`	string
oldValue *optional*	**Example** : `"disconnected"`	string
parameter *required*	**Example** : ParameterName	ParameterName
persistTime *required*	**Example** : PersistTime	PersistTime
receiveTime *required*	**Example** : ReceiveTime	ReceiveTime

5.50. DeviceProcess

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *optional*	**Example** : `"object"`	Value

Key

Name	Description	Schema
device *required*	**Example** : DeviceName	DeviceName

Value

Name	Description	Schema
completionTime *optional*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
current *optional*	**Example** : DeviceProcessState	DeviceProcessState
device *required*	**Example** : DeviceName	DeviceName
executeUntilTime *optional*	Deadline - latest point in time at which this process will be executed **Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
executionType *optional*	**Example** : DeviceProcessExecutionType	DeviceProcessExecutionType
externalId *optional*	**Example** : `"6fcb153f-e47e-4e53-94d0-ba98009f0070"`	string
id *required*	**Example** : `0`	integer (int64)
initTime *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
parameters *optional*	**Example** : DeviceProcessParameters	DeviceProcessParameters
persistTime *required*	**Example** : PersistTime	PersistTime
previous *optional*	**Example** : DeviceProcessState	DeviceProcessState
startTime *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)

Name	Description	Schema
stopTime *optional*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
type *required*	**Example** : `"DCT_OnDemandRead"`	string
username *required*	**Example** : `"string"`	string

5.51. DeviceProcessExecutionType

Type : enum (single, scheduled, recurring)

5.52. DeviceProcessParameters

Parameters of a device process. Only a single property will be set. Which property is set, depends on the process type.

Name	Description	Schema
activityCalendarRead *optional*	**Example** : ActivityCalendarRead	ActivityCalendarRead
activityCalendarWrite *optional*	**Example** : ActivityCalendarWrite	ActivityCalendarWrite
alarmRegisterRead *optional*	**Example** : AlarmRegisterRead	AlarmRegisterRead
alarmRegisterReset *optional*	**Example** : AlarmRegisterReset	AlarmRegisterReset
certificateExchange *optional*	**Example** : CertificateExchange	CertificateExchange
cosemAccess *optional*	**Example** : CosemAccess	CosemAccess
cosemProfileConfigurationRead *optional*	**Example** : CosemProfileConfigurationRead	CosemProfileConfigurationRead

Name	Description	Schema
cosemProfileConfigurationWrite *optional*	**Example** : CosemProfileConfigurationWrite	CosemProfileConfigurationWrite
customerOptOutRead *optional*	**Example** : CustomerOptOutRead	CustomerOptOutRead
displayConfigurationRead *optional*	**Example** : DisplayConfigurationRead	DisplayConfigurationRead
displayConfigurationWrite *optional*	**Example** : DisplayConfigurationWrite	DisplayConfigurationWrite
engineeringTokenTransfer *optional*	**Example** : EngineeringTokenTransfer	EngineeringTokenTransfer
firmwareUpdate *optional*	**Example** : FirmwareUpdate	FirmwareUpdate
firmwareUpdateOverTheAir *optional*	**Example** : FirmwareUpdateOverTheAir	FirmwareUpdateOverTheAir
genericAction *optional*	**Example** : GenericAction	GenericAction
globalkeyExchange *optional*	**Example** : GlobalKeyExchange	GlobalKeyExchange
keyExchange *optional*	**Example** : KeyExchange	KeyExchange
keyPairGeneration *optional*	**Example** : KeyPairGeneration	KeyPairGeneration
logRead *optional*	**Example** : LogRead	LogRead
mbusKeyExchange *optional*	**Example** : MbusKeyExchange	MbusKeyExchange
onDemandRead *optional*	**Example** : OnDemandRead	OnDemandRead

Name	Description	Schema
p0PortRead *optional*	**Example** : P0PortRead	P0PortRead
p0PortWrite *optional*	**Example** : P0PortWrite	P0PortWrite
p1PortRead *optional*	**Example** : P1PortRead	P1PortRead
p1PortWrite *optional*	**Example** : P1PortWrite	P1PortWrite
profileRead *optional*	**Example** : ProfileRead	ProfileRead
profileRecordingControlRead *optional*	**Example** : ProfileRecordingControlRead	ProfileRecordingControlRead
profileRecordingControlWrite *optional*	**Example** : ProfileRecordingControlWrite	ProfileRecordingControlWrite
pushSetupConfigurationRead *optional*	**Example** : PushSetupConfigurationRead	PushSetupConfigurationRead
pushSetupConfigurationWrite *optional*	**Example** : PushSetupConfigurationWrite	PushSetupConfigurationWrite
registerMetadataRead *optional*	**Example** : RegisterMetadataRead	RegisterMetadataRead
secretExchange *optional*	**Example** : SecretExchange	SecretExchange
tokenTransfer *optional*	**Example** : TokenTransfer	TokenTransfer

5.53. DeviceProcessState

Name	Description	Schema
error *optional*	**Example** : `"string"`	string

Name	Description	Schema
status *required*	**Example** : ProcessStatus	ProcessStatus

5.54. DeviceProfileFacts

Name	Description	Schema
Key *optional*	**Example** : `"object"`	Key
Value *optional*	**Example** : `"object"`	Value

Key

Name	Description	Schema
device *required*	**Example** : DeviceName	DeviceName

Value

Name	Description	Schema
actual *required*	**Example** : `0`	integer (int64)
completeness *required*	**Example** : `0.0`	number (double)
device *required*	**Example** : DeviceName	DeviceName
expected *required*	**Example** : `0`	integer (int64)
from *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
gatheredAt *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
profile *required*	**Example** : ProfileName	ProfileName
to *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)

5.55. DeviceRegister

Name	Description	Schema
device *optional*	**Example** : `"device-name"`	string
obisCode *optional*	Unique identifier for a profile in ZONOS (OBIS code) **Example** : `"1-0:99.1.0*255"`	string

5.56. DeviceState

Name	Description	Schema
communicationId *optional*	**Example** : `"HXEE81010001"`	string
description *optional*	**Example** : `"Only accessible from the left side"`	string
device *required*	**Example** : DeviceName	DeviceName
dispatchGroup *optional*	**Example** : `"Cell_ID_1"`	string
group *required*	**Example** : `"ROOT"`	string
groupUuid *optional*	**Example** : `"bbca72a6-ef6a-4ecd-87f9-6a05647b6e67"`	string (uuid)
identifiers *optional*	Additional device identifiers **Example** : `{ "string" : "string" }`	< string, string > map
inventoryState *required*	**Example** : InventoryState	InventoryState
location *optional*	**Example** : DeviceLocation	DeviceLocation
managementState *required*	**Example** : ManagementState	ManagementState
manufacturer *optional*	**Example** : `"Hexing"`	string
model *optional*	**Example** : `"HXE310 MKII"`	string
tags *optional*	**Example** : `[ "DeviceTag" ]`	< DeviceTag > array

Name	Description	Schema
type *required*	**Example** : `"Hexing HXE310"`	string

5.57. DeviceStatusFlag

Type : enum (critical_error, clock_adjusted, power_failure, clock_invalid, data_invalid)

5.58. DeviceTag

Type : string

5.59. DeviceTelemetry

Name	Description	Schema
Key *optional*	**Example** : `"object"`	Key
Value *optional*	**Example** : `"object"`	Value

Key

Name	Description	Schema
device *required*	**Example** : DeviceName	DeviceName

Value

Name	Description	Schema
device *required*	**Example** : DeviceName	DeviceName
deviceCameOnline *optional*	**Example** : DeviceCameOnline	DeviceCameOnline
deviceCommunicated *optional*	**Example** : DeviceCommunicated	DeviceCommunicated
deviceWentOffline *optional*	**Example** : DeviceWentOffline	DeviceWentOffline
lastReceiveTime *required*	Point in time since the last event for this device from an online data source was received **Example** : `"1970-01-01T00:00:00Z"`	string (date-time)

5.60. DeviceTopology

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *required*	**Example** : `"object"`	Value

Key

Name	Description	Schema
device *required*	**Example** : DeviceName	DeviceName

Value

Name	Description	Schema
children *required*	**Example** : `[ "TopologyChild" ]`	< TopologyChild > array
device *required*	**Example** : DeviceName	DeviceName
persistTime *required*	**Example** : PersistTime	PersistTime
receiveTime *required*	**Example** : ReceiveTime	ReceiveTime

5.61. DeviceWentOffline

Name	Description	Schema
lastCommunicatedAt *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
wentOfflineAt *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)

5.62. DisconnectorState

Type : enum (disconnected, connected)

5.63. DisplayConfiguration

Name	Description	Schema
loadProfileEnabled *required*	**Example** : `true`	boolean

5.64. DisplayConfigurationRead

Type : object

5.65. DisplayConfigurationWrite

Name	Description	Schema
configuration *required*	**Example** : DisplayConfiguration	DisplayConfiguration

5.66. EditReason

Reason given by the user for the edition of this measurement

Type : string

5.67. Editor

User who edited the value of this measurement

Name	Description	Schema
UserId *optional*	**Example** : `"user-id"`	string
Username *optional*	**Example** : `"username"`	string

5.68. EngineeringTokenTransfer

Name	Description	Schema
deviceName *required*	**Example** : `"string"`	string
method *optional*	Methods transferred tokens may be used for. Only one property may be set. **Example** : `"object"`	method

method

Name	Description	Schema
clearAlarm *optional*	**Example** : ClearAlarm	ClearAlarm
clearCredit *optional*	**Example** : ClearCredit	ClearCredit
setDisconnectorState *optional*	**Example** : SetDisconnectorState	SetDisconnectorState
setLoadLimitation *optional*	**Example** : SetLoadLimitation	SetLoadLimitation
setPaymentMode *optional*	**Example** : SetPaymentMode	SetPaymentMode

5.69. EventCode

Unique identifier for an event in ZONOS

Type : string

5.70. EventState

Type : enum (unknown, open, closed)

5.71. FirmwareUpdate

Name	Description	Schema
firmwareId *required*	**Example** : `0`	integer (int64)

5.72. FirmwareUpdateOverTheAir

Name	Description	Schema
fallbackImageName *required*	**Example** : `"string"`	string
firmwareId *required*	**Example** : `0`	integer (int64)
imageName *required*	**Example** : `"string"`	string

Name	Description	Schema
ipAddress *required*	**Example** : `"string"`	string
password *required*	**Example** : `"string"`	string
port *required*	**Example** : `0`	integer
serverType *required*	**Example** : ServerType	ServerType
updateAction *required*	**Example** : UpdateAction	UpdateAction
username *required*	**Example** : `"string"`	string

5.73. GenericAction

Name	Description	Schema
actionName *optional*	**Example** : `"string"`	string

5.74. GeoLocation

A geolocation expressed in latitude and longitude

Name	Description	Schema
latitude *required*	**Example** : `50.9569977`	number
longitude *required*	**Example** : `11.0318898`	number

5.75. GlobalKeyExchange

Type : object

5.76. Group

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *optional*	**Example** : `"object"`	Value

Key

Name	Description	Schema
group *required*	**Example** : GroupId	GroupId

Value

Name	Description	Schema
current *optional*	**Example** : GroupState	GroupState
persistTime *required*	**Example** : PersistTime	PersistTime
previous *optional*	**Example** : GroupState	GroupState

5.77. GroupId

Unique identifier for a group

Type : string (uuid)

5.78. GroupState

Name	Description	Schema
description *optional*	**Example** : `"Root Group of the project"`	string
id *required*	**Example** : `"bbca72a6-ef6a-4ecd-87f9-6a05647b6e67"`	string (uuid)
name *required*	**Example** : `"ROOT"`	string
parent *optional*	**Example** : `"c22cf463-03ef-4663-812e-8ef7be895314"`	string (uuid)

5.79. InventoryState

Type : enum (unknown, in_storage, quality_control, installed, uninstalled, disposed, quarantined)

5.80. KeyExchange

Name	Description	Schema
client *optional*	The device client for which to exchange the keys **Example** : `"string"`	string
keyTypes *required*	The types of keys to exchange **Example** : `[ "KeyExchangeKeyType" ]`	< KeyExchangeKeyType > array
mechanism *required*	**Example** : KeyExchangeMechanism	KeyExchangeMechanism

5.81. KeyExchangeKeyType

Type : enum (authentication_key, encryption_key, master_key)

5.82. KeyExchangeMechanism

Type : enum (key_agreement, key_transfer)

5.83. KeyPairGeneration

Type : object

5.84. Location

Name	Description	Schema
address *optional*	**Example** : Address	Address
geo *optional*	**Example** : GeoLocation	GeoLocation

5.85. LogRead

Name	Description	Schema
logId *required*	The name of the log which is read **Example** : `"string"`	string

5.86. ManagementState

Type : enum (unknown, registered, connected, billing, was_billing)

5.87. MbusKeyExchange

Type : object

5.88. MeasureTime

Point in time when the value was measured on the device. For extended registers.

Type : string (date-time)

5.89. MeteringPoint

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *optional*	**Example** : `"object"`	Value

Key

Name	Description	Schema
meteringPoint *required*	**Example** : MeteringPointIdentifier	MeteringPointIdentifier

Value

Name	Description	Schema
current *optional*	**Example** : MeteringPointState	MeteringPointState
persistTime *required*	**Example** : PersistTime	PersistTime
previous *optional*	**Example** : MeteringPointState	MeteringPointState

5.90. MeteringPointIdentifier

Unique identifier for a metering point in ZONOS

Type : string

5.91. MeteringPointParameter

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *required*	**Example** : `"object"`	Value

Key

Name	Description	Schema
meteringPoint *required*	**Example** : MeteringPointIdentifier	MeteringPointIdentifier

Value

Name	Description	Schema
changeTime *optional*	Point in time at which the change became effective **Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
meteringPoint *required*	**Example** : MeteringPointIdentifier	MeteringPointIdentifier
newValue *optional*	**Example** : `"ready to connect"`	string
oldValue *optional*	**Example** : `"disconnected"`	string
parameter *required*	**Example** : ParameterName	ParameterName
persistTime *required*	**Example** : PersistTime	PersistTime

5.92. MeteringPointState

Name	Description	Schema
devices *required*	**Example** : `[ "DeviceAssignment" ]`	< DeviceAssignment > array
group *required*	**Example** : `"ROOT"`	string
groupUuid *optional*	**Example** : `"bbca72a6-ef6a-4ecd-87f9-6a05647b6e67"`	string (uuid)
location *optional*	**Example** : Location	Location
meteringPoint *required*	**Example** : MeteringPointIdentifier	MeteringPointIdentifier
registers *optional*	**Example** : `[ "Register" ]`	< Register > array
serviceLevel *required*	**Example** : `"active"`	string

Name	Description	Schema
stateHistory *required*	**Example** : `[ "MeteringPointStateHistoryEntry" ]`	< MeteringPointStateHistoryEntry > array
supplierHistory *optional*	**Example** : `[ "SupplierAssignment" ]`	< SupplierAssignment > array

5.93. MeteringPointStateEnum

Type : enum (unknown, under_establishment, connected, disconnected, terminated)

5.94. MeteringPointStateHistoryEntry

Name	Description	Schema
activeSince *optional*	Point in time since when the state was active **Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
state *optional*	**Example** : MeteringPointStateEnum	MeteringPointStateEnum

5.95. OnDemandRead

Name	Description	Schema
readingReasonCode *required*	The reading reason code of this on demand read **Example** : `1.0`	number
registerGroups *required*	**Example** : `[ "string" ]`	< string > array
registers *required*	**Example** : `[ "string" ]`	< string > array

5.96. OperationalProcess

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *optional*	**Example** : `"object"`	Value

Key

Name	Description	Schema
operationalProcessId *required*	**Example** : `0`	integer (int64)

Value

Name	Description	Schema
current *optional*	**Example** : OperationalProcessState	OperationalProcessState
externalId *optional*	**Example** : `"6fcb153f-e47e-4e53-94d0-ba98009f0070"`	string
id *required*	**Example** : `0`	integer (int64)
persistTime *required*	**Example** : PersistTime	PersistTime
previous *optional*	**Example** : OperationalProcessState	OperationalProcessState
type *required*	**Example** : `"BATCH_DCT"`	string

5.97. OperationalProcessState

Name	Description	Schema
status *optional*	**Example** : ProcessStatus	ProcessStatus

5.98. P0PortRead

Type : object

5.99. P0PortWrite

Name	Description	Schema
enable *required*	**Example** : `true`	boolean

5.100. P1PortRead

Type : object

5.101. P1PortWrite

Name	Description	Schema
outputInterval *required*	**Example** : `0`	integer

5.102. ParameterName

Unique identifier for a parameter in ZONOS

Type : string

5.103. PaymentMode

Type : enum (prepayment, postpayment)

5.104. PersistTime

Point in time since this change was persisted by the ZONOS Core

Type : string (date-time)

5.105. ProcessStatus

Type : enum (inactive, ready, executing, finished, failed, deleted)

5.106. ProfileName

Unique identifier for a profile in ZONOS (OBIS code)

Type : string

5.107. ProfileRead

Name	Description	Schema
from *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
profileId *required*	The name of the profile which is read **Example** : `"string"`	string
readingReasonCode *required*	The reading reason code of this profile read **Example** : `1.0`	number

Name	Description	Schema
to *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)

5.108. ProfileRecordingControlRead

Type : object

5.109. ProfileRecordingControlWrite

Name	Description	Schema
enable *required*	**Example** : `true`	boolean

5.110. PushSetupConfigurationRead

Name	Description	Schema
obisCode *required*	**Example** : `"string"`	string

5.111. PushSetupConfigurationWrite

Name	Description	Schema
id *required*	**Example** : `"string"`	string
obisCode *required*	**Example** : `"string"`	string

5.112. Quality

Type : enum (true_value, manual_value, replacement_value, forecast_value, faulty_value)

5.113. ReceiveTime

Point in time since this change was received by the ZONOS Core

Type : string (date-time)

5.114. Register

Register information

Name	Description	Schema
dataSourceHistory *optional*	**Example** : `[ "DataSource" ]`	< DataSource > array
obisCode *optional*	Unique identifier for a profile in ZONOS (OBIS code) **Example** : `"1-0:99.1.0*255"`	string
unit *optional*	Unit of measurement **Example** : `"kWh"`	string

5.115. RegisterMetadataRead

Name	Description	Schema
id *optional*	**Example** : `"string"`	string
registers *required*	**Example** : `[ "string" ]`	< string > array

5.116. RegisterName

Unique identifier for a register in ZONOS (OBIS code)

Type : string

5.117. RegisterStatistic

Name	Description	Schema
Key *required*	**Example** : `"object"`	Key
Value *required*	**Example** : `"object"`	Value

Key

Name	Description	Schema
device *required*	**Example** : DeviceName	DeviceName

Value

Name	Description	Schema
date *required*	**Example** : `"1970-01-01"`	string (date)

Name	Description	Schema
deliveryDelay *required*	**Example** : `"object"`	deliveryDelay
device *required*	**Example** : DeviceName	DeviceName
meterReads *required*	**Example** : `"object"`	meterReads
register *required*	**Example** : RegisterName	RegisterName

deliveryDelay

Name	Description	Schema
average *required*	**Minimum value** : `0` **Example** : `1337.42`	number (double)
maximum *required*	**Minimum value** : `0` **Example** : `4711`	integer
minimum *required*	**Minimum value** : `0` **Example** : `42`	integer

meterReads

Name	Description	Schema
edited *required*	**Minimum value** : `0` **Example** : `2`	integer
estimated *required*	**Minimum value** : `0` **Example** : `6`	integer
expected *required*	**Minimum value** : `0` **Example** : `96`	integer
invalidated *required*	**Minimum value** : `0` **Example** : `3`	integer
received *required*	**Minimum value** : `0` **Example** : `93`	integer

5.118. RegisterValue

Name	Description	Schema
captureTime *optional*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
registerId *optional*	**Example** : `"string"`	string

Name	Description	Schema
scaler *optional*	**Example** : `0`	integer
unitSymbol *optional*	**Example** : `"string"`	string
value *optional*	**Example** : `0`	integer (int64)

5.119. SeasonProfile

Name	Description	Schema
name *required*	**Example** : `"string"`	string
startAt *required*	**Example** : `"string"`	string
weekProfileName *required*	**Example** : `"string"`	string

5.120. SecretExchange

Name	Description	Schema
client *required*	**Example** : `"string"`	string

5.121. ServerType

Type : enum (ftp, ftps_with_auth_ssl, ftps_with_auth_tls, ftps_implicit, unknown)

5.122. SetDisconnectorState

Name	Description	Schema
disconnectorState *optional*	**Example** : DisconnectorState	DisconnectorState

5.123. SetLoadLimitation

Name	Description	Schema
limit *optional*	**Example** : `0`	integer (int64)

5.124. SetPaymentMode

Name	Description	Schema
paymentMode *optional*	**Example** : PaymentMode	PaymentMode

5.125. SupplierAssignment

Supplier information

Name	Description	Schema
activeSince *optional*	Point in time since when the state was active **Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
activeUntil *optional*	Point in time until when the state was active **Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
externalId *optional*	**Example** : `"Supplier-external-id"`	string
name *optional*	**Example** : `"Supplier-name"`	string

5.126. TokenTransfer

Name	Description	Schema
token *optional*	**Example** : `"string"`	string

5.127. TopologyChild

Name	Description	Schema
addTime *required*	**Example** : `"1970-01-01T00:00:00Z"`	string (date-time)
childDevice *required*	**Example** : `"81010001-1"`	string
dataSource *required*	**Example** : DataSourceName	DataSourceName

5.128. UpdateAction

Type : enum (update, fallback, unknown)

5.129. WeekProfile

Name	Description	Schema
friday *required*	**Example** : `0`	integer
monday *required*	**Example** : `0`	integer
name *required*	**Example** : `"string"`	string
saturday *required*	**Example** : `0`	integer
sunday *required*	**Example** : `0`	integer
thursday *required*	**Example** : `0`	integer
tuesday *required*	**Example** : `0`	integer
wednesday *required*	**Example** : `0`	integer

Chapter 6. Appendix

6.1. Incubating examples

6.1.1. GET /register-statistic

Example HTTP response


```
{
  "Key": {
    "device": "81010001",
    "register": "1-0:1.8.0*255",
    "date": "2022-08-29"
  },
  "Value": {
    "device": "81010001",
    "register": "1-0:1.8.0*255",
    "date": "2022-08-29",
    "meterReads": {
      "expected": 96,
      "received": 93,
      "edited": 2,
      "invalidated": 3,
      "estimated": 6
    },
    "deliveryDelay": {
      "minimum": 42,
      "maximum": 4711,
      "average": 1337.42
    }
  }
}
```


6.1.2. GET /device-derived-measurement

Example HTTP response


```
{
  "device" : "meter1",
  "register" : "1-0:1.8.0*255",
  "persistTime" : "2020-08-14T13:27:12.947Z",
  "unit" : "kWh",
  "consumptionDataPoints" : [ {
    "time" : "2020-08-13T14:00:00.000Z",
    "value" : "123.12",
    "quality" : "true_value"
  }, {
```

```
    "time" : "2020-08-13T14:15:00.000Z",
    "value" : "123.12",
    "quality" : "true_value"
  }, {
    "time" : "2020-08-13T14:30:00.000Z",
    "value" : "000.00",
    "quality" : "faulty_value"
  }, {
    "time" : "2020-08-13T14:45:00.000Z",
    "value" : "123.12",
    "quality" : "true_value"
  }, {
    "time" : "2020-08-13T15:00:00.000Z",
    "value" : "123.12",
    "quality" : "true_value"
  }, {
    "time" : "2020-08-13T16:30:00.000Z",
    "value" : "999.12",
    "quality" : "true_value"
  } ]
}
```


6.1.3. GET /device-process

Example HTTP response


```
{
  "id" : 1187,
  "type" : "DCT_FirmwareUpdate",
  "device" : "fw2",
  "persistTime" : "2020-08-18T12:58:22.330Z",
  "previous" : {
    "status" : "ready"
  },
  "current" : {
    "status" : "failed"
  },
  "parameters" : {
    "firmwareUpdate" : {
      "firmwareId" : 10
    }
  }
}
```


6.2. Stable examples

6.2.1. GET /device

Example HTTP response


```
{
  "Key": {
    "device": "81010001"
  },
  "Value": {
    "previous": {
      "device": "81010001",
      "type": "Hexing HXE310",
      "group": "ROOT",
      "communicationId": "HXEE81010001",
      "inventoryState": "installed",
      "managementState": "connected",
      "manufacturer": "Hexing",
      "description": "Only accessible from the left side",
      "model": "HXE310 MKII",
      "location": {
        "geo": {
          "latitude": 50.9569977,
          "longitude": 11.0318898
        },
        "address": {
          "city": "Erfurt",
          "postalCode": 99096,
          "street": "Arndtstraße",
          "houseNumber": "1b"
        },
        "logicalInstallationPoint": "FB4RL2"
      },
      "tags": ["foo", "bar"]
      "groupUuid": "bbca72a6-ef6a-4ecd-87f9-6a05647b6e67",
      "dispatchGroup": "LP7773"
      "identifiers": {
        "sap_equipment_id": "81010001",
        "dlms.logical_device_name": "81010001HXE19512"
      }
    },
    "current": {
      "device": "81010001",
      "type": "Hexing HXE310",
      "group": "ROOT",
      "communicationId": "HXEE81010001",
      "inventoryState": "installed",
      "managementState": "connected",
      "manufacturer": "Hexing",
      "description": "Only accessible from the left slide",
      "model": "HXE310 MKII",
```

```
      "location": {
        "geo": {
          "latitude": 50.9569977,
          "longitude": 11.0318898
        },
        "address": {
          "city": "Erfurt",
          "postalCode": 99096,
          "street": "Arndtstraße",
          "houseNumber": "1b"
        },
        "logicalInstallationPoint": "FB4RL2"
      }
      "tags": ["foo", "bar"]
      "groupUuid": "bbca72a6-ef6a-4ecd-87f9-6a05647b6e67",
      "dispatchGroup": "LP77734"
      "identifiers": {
        "sap_equipment_id": "81010001",
        "dlms.logical_device_name": "81010001HXE19512"
      }
    },
    "persistTime": "2019-08-29T14:13:44.535Z"
  }
}
```


6.2.2. GET /device-event

Example HTTP response


```
{
  "Key": {
    "device": "81010001"
  },
  "Value": {
    "device": "81010001",
    "event": "308000",
    "firstOccurrenceTime": "2022-08-29T14:27:29.371Z",
    "lastOccurrenceTime": "2022-08-29T14:27:29.371Z",
    "occurrenceCount": 4,
    "receiveTime": "2022-08-29T14:27:29.371Z",
    "persistTime": "2022-08-29T14:27:29.371Z",
    "state": "open",
    "context": {
      "1": "HXEE81010001",
      "2": "2021-12-31T23:59:59.123Z"
    }
  }
}
```

6.2.3. GET /device-measurement

Example HTTP response


```
{
  "Key": {
    "device": "81010001"
  },
  "Value": {
    "device": "81010001",
    "profile": "1-0:99.1.0*255",
    "measureTime": "2019-08-29T14:27:46.907Z",
    "receiveTime": "2019-08-29T14:27:46.907Z",
    "persistTime": "2019-08-29T14:27:46.907Z",
    "readingReason": "Periodic",
    "dataPoints": [
      {
        "register": "1-0:1.8.0*255",
        "value": 42.1337,
        "unit": "kWh"
      }
    ]
  }
}
```


6.2.4. GET /device-parameter

Example HTTP response


```
{
  "Key": {
    "device": "81010001",
  },
  "Value": {
    "device": "81010001",
    "parameter": "ext.foo",
    "persistTime": "2019-08-29T14:27:59.558Z",
    "newValue": "ready to connect",
    "oldValue": "disconnected"
  }
}
```


6.2.5. GET /device-telemetry

Example HTTP response


```
{
```

```
  "Key": {
    "device": "81010001"
  },
  "Value": {
    "device": "81010001",
    "lastReceiveTime": "2019-08-29T14:28:13.343Z"
  }
}
```



```
{
  "Key": {
    "device": "81010001"
  },
  "Value": {
    "device" : "1KFM100000018",
    "lastReceiveTime" : "2021-11-03T09:56:20Z",
    "deviceCommunicated" : {
      "communicatedAt" : "2021-11-03T09:56:20Z"
    }
  }
}
```


6.2.6. GET /device-topology

Example HTTP response


```
{
  "Key": {
    "device": "81010001"
  },
  "Value": {
    "device": "81010001",
    "receiveTime": "2019-08-29T14:28:25.077Z",
    "persistTime": "2019-08-29T14:28:25.077Z",
    "children": [
      {
        "childDevice": "81010001-1",
        "addTime": "2019-08-29T14:28:25.077Z",
        "dataSource": "DLMS Connector"
      }
    ]
  }
}
```


6.2.7. GET /metering-point

Example HTTP response


```
{
  "Key": {
    "meteringPoint": "DE00056266802AO6G56M11SN51G21M24S"
  },
  "Value": {
    "previous": {
      "meteringPoint": "DE00056266802AO6G56M11SN51G21M24S",
      "group": "ROOT",
      "serviceLevel": "active",
      "location": {
        "geo": {
          "latitude": 50.9569977,
          "longitude": 11.0318898
        },
        "address": {
          "city": "Erfurt",
          "postalCode": 99096,
          "street": "Arndtstraße",
          "houseNumber": "1b"
        }
      },
      "devices": [
        {
          "device": "81010001",
          "assignTime": "2022-07-29T14:28:34.775Z",
          "removeTime": "2022-08-29T12:28:34.775Z"
        }
      ]
    },
    "current": {
      "meteringPoint": "DE00056266802AO6G56M11SN51G21M24S",
      "group": "ROOT",
      "serviceLevel": "active",
      "location": {
        "geo": {
          "latitude": 50.9569977,
          "longitude": 11.0318898
        },
        "address": {
          "city": "Erfurt",
          "postalCode": 99096,
          "street": "Arndtstraße",
          "houseNumber": "1b"
        }
      },
      "devices": [
        {
          "device": "81010001",
          "assignTime": "2022-07-29T14:28:34.775Z",
```

```
          "removeTime": "2022-08-29T12:28:34.775Z"
        },
        {
          "device": "81010042",
          "assignTime": "2022-08-30T14:28:34.775Z"
        }
      ]
    },
    "persistTime": "2019-08-29T14:28:34.775Z"
  }
}
```


6.2.8. GET /metering-point-parameter

Example HTTP response


```
{
  "Key": {
    "meteringPoint": "DE00056266802AO6G56M11SN51G21M24S",
  },
  "Value": {
    "meteringPoint": "DE00056266802AO6G56M11SN51G21M24S",
    "parameter" : "ext.foo",
    "changeTime" : "2023-05-24T10:12:07.835Z",
    "receiveTime" : "2023-05-24T10:12:07.835Z",
    "persistTime" : "2023-05-24T10:12:07.835Z",
    "newValue" : "43",
    "oldValue" : "42"
  }
}
```


6.2.9. GET /group

Example HTTP response


```
{
  "Key": {
    "id": "bbca72a6-ef6a-4ecd-87f9-6a05647b6e67"
  },
  "Value": {
    "previous": {
      "id": "bbca72a6-ef6a-4ecd-87f9-6a05647b6e67",
      "name": "Group name",
      "description": "This is a test group",
      "parent": "c22cf463-03ef-4663-812e-8ef7be895314"
    },
    "current": {
      "id": "bbca72a6-ef6a-4ecd-87f9-6a05647b6e67",
```

```
      "name": "New group name",
      "description": "This is a test group",
      "parent": "c22cf463-03ef-4663-812e-8ef7be895314"
    },
    "persistTime": "2022-04-14T09:28:34.775Z"
  }
}
```

6.3. Previous changelogs

Latest changelogs

6.3.1. ZONOS 50

- The topic `ext_metering-point-parameter_{instance_name}` was added.
  - It contains the parameters of metering points.
  - A message will be emitted when a parameter was added, changed or deleted.

6.3.2. ZONOS 49

- The property `dispatchGroup` has been added to the topic `ext_device_{instance_name}`.

6.3.3. ZONOS 48

- The property `groupUuid` has been added to the topic `ext_metering-point_{instance_name}`.
- Messages on external Kafka topics `ext_device-event_{instance_name}`, `ext_device-parameter_{instance_name}`, `ext_device-process_{instance_name}`, `ext_register-statistic_{instance_name}` are now *guaranteed* to be partitioned by device, i.e. all messages related to the same device will always end up in the same partition. Applications relying on the previous implementation of message keys may need to be adapted.

6.3.4. ZONOS 46

- Topic `ext_group_{instance_name}` was added. It contains the groups currently available in ZONOS.
- The property `groupUuid` has been added to the topic `ext_device_{instance_name}`.
- Messages on all topics available on this API are compressed with the gzip codec from now on
  - This behaviour is configurable in the ZONOS central configuration file
  - Compression is transparent to consumers

6.3.5. ZONOS 42

- Topic `ext_device-profile-facts_{instance_name}` was added. It contains statistics about the data completeness of a profile.
- The model for device telemetry was extended. The events contain now new fields which show if

a device communicates, came online or went offline. Except these cases, the events look like before and provides the information when the device receives last data.

6.3.6. ZONOS 33

- Topic `ext_device-event-v2_{instance_name}` was added. It contains device events and their context as read from the device event logs. Also included are alarm code and first occurrence time to identify ZONOS alarms that were created for these events. ZONOS alarms are exposed on topic ext_device-event_{instance_name} as before.
- New device process types are available representing new device tasks:
  - `secretExchange`
  - `displayConfigurationRead`
  - `displayConfigurationWrite`
  - `profileRecordingControlRead`
  - `profileRecordingControlWrite`
  - `firmwareUpdateOverTheAir`
- The model for device measurements was extended. Their data points now also contain the timestamp at which they were measured. This is usually the same as the device measurement's measure time. Exceptions are values from extended registers, as these provide their own measurement time. This affects topic `ext_device-measurement_{instance_name}`, `dataPoints` now have a property `measureTime`.

6.3.7. ZONOS 32

- A new device inventory state `quarantined` has been introduced in ZONOS. This impacts the `inventoryState` property of `DeviceState` objects on topic `ext_device_{instance_name}`.

6.3.8. ZONOS 31

- Renamed device process parameter model properties for reading and resetting alarm registers from `alarmRead` and `alarmReset` to `alarmRegisterRead` and `alarmRegisterReset` to avoid internal naming conflict.

6.3.9. ZONOS 28

- The data model for metering points on topic `ext_metering-point_{instance_name}` has been extended with the metering point's state history.

6.3.10. ZONOS 27

The following topic was added:

- ext_device-derived-measurement_{instance_name}
  - This topic will contain VEE data for devices. A message will be emitted when the VEE service computes data for a device. See `DeviceDerivedMeasurement` for details on the structure of the

messages.

6.3.11. ZONOS 24

The following topics were added:

- ext_device-process_{instance_name}
  - This topic will contain messages for device processes. A message will be emitted when a device process is changed, e.g. when its state transitions from *running* to *successful*. See `DeviceProcess` for details on the structure of the messages.
- ext_operational-process_{instance_name}
  - This topic will contain messages for operational processes. A message will be emitted when an operational process is changed, e.g. when its state transitions from *running* to *successful*. See `OperationalProcess` for details on the structure of the messages.

6.3.12. ZONOS 17

The following properties were added:

- `DeviceMeasurement.Value` now has a property `status` which is an array of device status flags