Arrowhead Historian Service IDD

**Abstract**

This document defines interface design of the Historian Service based on REST within Arrowhead generation 4.0.

1. Overview

This document describes the Historian service of G4.0 with the REST interface.

This interface uses HTTP, HTTPS, CoAP or CoAPS, therefore the related CP is valid.

1. Interfaces

As per the SD of this Service, there are three methods implemented. Table 1 describe these.

Table 1 Function description

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function** | **URL subpath** | **Method** | **Input** | **Output** |
| Store | “/historian/{systemName}” | PUT | **SensorData GenericFile** | HTTP CODE: CREATED, NO\_CONTENT  BAD\_REQUEST |
| Retrieve | “/historian/{systemName}” | GET | **SensorData GenericFile** | HTTP CODE: OK, NO\_CONTENT  BAD\_REQUEST |

1. Information Model

The SensorData request payload contains generic sensor data. SensorData normally contains information about the unit, source system, timestamp and metadata. The default payload type is JSON-encoded SenML encapsulated in the SigML format. The response to a Store/ Retrieve request is a simple HTTP/CoAP status code (Created/Ok – request was success, No Content – request had no effect).

SigML moves the *bn*, *bt*, *bu,* and  *bver* tags from SenML into the the SigML header. The SenML array of JSON objects are stored in the *sml* tag. The only allowed SenML tags allowed in the sml tag are “n” tags. This approach simplifies passing messages since SigML is encoded as a JSON object instead of a JSON array.

# 

# 3.1. SensorData Store- SigML request

{“p”: 0,  
 "bn":" Temp-sensor-3624-2342",  
 "bt":1.276020076001e+09,

“bu":" degC ","bver":5,

“sml”:[

{"n":"temp", “v":22.3},

{"n":"temp", "v":22.3 “t”: -5}

]

}

# 3.2. SensorData Store- SigML response

{“p”: 0,  
 "x":" 0

}

The x tag (eXeption) value of 0 indicates “No error”. A value different than 0 must also be accompanied by an xs tag that in plain text gives a reason for the error.

If the uploaded JSON is incorrectly constructed, then the following example response could be received.

{“p”: 0,  
 "x":" 2,

“xs”: “JSON Parse error”

}

# 3.2. GenericFile

The GenericFile can be any type of file. The extension and mime type define the content.

# Revision history

# Amendments

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Date | Version | Subject of Amendments | Author |
| 1 | 2018-09-19 | G4.0 | Initial | Jens Eliasson |
| 2 | 2018-10-30 | G4.0 | Text update | Jens Eliasson |
|  |  |  |  |  |

# Quality Assurance

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Date | Version | Approved by |
| 1 |  |  |  |
| 2 |  |  |  |