Arrowhead DataManager Core Service 4.1.3 Release Notes

In Arrowhead version 4.1.3, the whole codebase of the Arrowhead framework has been rewritten using Spring Boot technology. We believe the new version is better both in code quality, performance and robustness. It is easier to use, maintain and improve.
However, this means that Arrowhead version 4.1.3 is NOT compatible with providers and consumers written for version 4.1.2 because the changes affect not just the backend but the public interface of the core services, too.
This document describes the new public API of the Arrowhead DataManager Core Service 4.1.3 and shows the key differences between the new interface and the old one.

Client Services - Proxy

These services can be used by consumers in an Arrowhead Cloud.

Uri: /datamanager/echo
Type: GET

Returns a “Got it” message with the purpose of testing the core service availability.

Uri: /datamanager/proxy
Type: GET

Gets a list of all systems that have created a proxy endpoint.

Output JSON structure for empty set:

```
{
  "systems":[]
}
```

or

```
{
  "systems":["sys1"]
}
```

if one system with the name ‘sys1’ has created a service endpoint.

Uri: /datamanager/proxy/
Type: PUT

Depending on input message, this operation can list, create or delete an endpoint for a specific service. The endpoint can then be used by a service data producer to push SenML data to the Proxy service.

Input JSON structure for List operation:

```
{
  "op":"list"
}
```

Input JSON structure for Create operation:

```
{
  "op":"create",
    "srvName": "_temperature._http._sys1._arrowhead.eu",
    "srvType": "_temperature"
}
```

Input JSON structure for Delete operation:

```
{
  "op":"delete",
  "srvName": "_temperature._http._sys1._arrowhead.eu"
}
```

Uri: /datamanager/proxy/<systemName>
Type: GET

Gets a list of all all services that system <systemName> have created.

Output JSON structure for empty set:

```
{
  "services":[]
}
```

or

```
{
  "services":[
    "_temperature._http._sys1._arrowhead.eu"
  ]
}
```

when one system named *sys1*, has registered one service. The example request URI is: /datamanager/proxy/sys1

Uri: /datamanager/proxy/<systemName>/<serviceName>
Type: PUT
Content-Type: application/json

Pushes a SenML message to a service endpoint named <serviceName> that system <systemName> have created. Building on the previously used examples with sys1 as system, the entire URI will be: /datamanager/proxy/sys1/_temperature._http._sys1._arrowhead.eu

Input JSON:

```
[
  {"bn":"_temperature._http._sys1._arrowhead.eu", "bt": 14625223},
  {"n": "temperature", "v": 19.2}
]
```

4.1.3.

- Note that the *bn* must occur once, and must be placed in the first JSON object in the array.
- Note that the *bt* must occur once, and must be placed in the first JSON object in the array. The bt tag is the number of seconds since 1/1 1970, i.e. UNIX time.
- Note that the *bu* can only occur once, and if exists, must be placed in the first JSON object in the array.
- The *bn* field must match the serviceName parameter.
- For more details, see the SenML RFC - https://tools.ietf.org/html/rfc8428

Uri: /datamanager/proxy/<systemName>/<serviceName>
Type: GET
Content-Type: application/json

Fetches the lastest SenML message from a service endpoint named <serviceName> belonging to system <systemName>.

Output JSON:

```
[
  {"bn":"_temperature._http._sys1._arrowhead.eu", "bt": 14625223},
  {"n": "temperature", "v": 19.2}
]
```

Note that the Proxy service only stores one message (the newest one), so if a data producer uploads messages faster than a consumer is downloading them, data will be used. Use the Historian service if data guarantees are needed.

Client Services - Historian

These services can be used by consumers in a local Arrowhead Cloud.

Uri: /datamanager/historian
Type: GET
Content-Type: application/json

Gets a list of all systems that have created a Historian service endpoint.

Output JSON structure for empty set:

```
{
  "systems":[]
}
```

or

```
{
  "systems":["sys1"]
}
```

if one system with the name ‘sys1’ has created at least one service endpoint.

Uri: /datamanager/historian/<systemName>
Type: GET
Content-Type: application/json

Gets a list of all all service endpoints that system <systemName> have created in the Historian.

Output JSON structure:

```
{
  "services":[
    "_sensor-342._temperature._http._tw.net"
  ]
}
```

Uri: /datamanager/historian/<systemName>
Type: PUT
Content-Type: application/json

Depending on input message, this operation can list, create or delete an endpoint for a specific service. The endpoint can then be used by a service data producer to push SenML data to the Historian service.

Input JSON structure for List operation:

```
{
  "op":"list"
}
```

Input JSON structure for Create operation:

```
{
  "op":"create",
    "srvName": "_temperature._http._sys1._arrowhead.eu",
    "srvType": "_temperature"
}
```

Output JSON structure for successful Create operation:

```
{
  "x": 0
}
```

Output JSON structure for unsuccessful Create operation:

```
{
  "x": -1,
  "xs": "Error string here"
}
```

Input JSON structure for Delete operation:

```
{
  "op":"delete",
  "srvName": "_temperature._http._sys1._arrowhead.eu"
}
```

Uri: /datamanager/historian/<systemName>/<serviceName>
Type: GET
Content-Type: application/json
Query params:

- *count* – number of measurements to fetch (optional),
- *sigX* – signal name, multiple sig parameters can be used, where X goes from 0 to the number of requested signals. For example: ?sig0=temperature&sig1=humidity
- *ts* – the timestamp to use
- *tsop* – the operation to use on the timestamp. Usage: ?ts= 1576473543&tsop=ge returns all measurements taken on or after the UNIX timestamp 1576473543. Can be;
  - ge – greater than or equal (default value if tsop is empty)
  - gt – greater than
  - le – less than or equal
  - ls – less than
  - eq – equal

This operation fetches sensor data from an endpoint.

Output JSON:

```
[
  {"bn":"_temperature._http._sys1._arrowhead.eu", "bt": 14625223},
  {"n": "temperature", "v": 19.2}
]
```

Uri: /datamanager/historian/<systemName>/<serviceName>
Type: PUT
Content-Type: application/json

This operation pushes sensor data into an endpoint.

Input JSON:

```
[
  {"bn":"_temperature._http._sys1._arrowhead.eu", "bt": 14625223},
  {"n": "temperature", "v": 19.2}
]
```

Private Services

Not available.