telemetrySensorTurbine HTTP/TLS/JSON

Interface Design Description

Abstract
This document defines the Interface Design Description (IDD) for the *telemetrySensorTurbine* service. The service provides real-time telemetry data acquisition from turbine sensors, supporting secure HTTP/TLS communication and JSON-encoded payloads.

Contents

1 Overview

The *telemetrySensorTurbine* service interface provides operations for registering sensors, retrieving telemetry data, and checking system status.

Profile Type	Type	Version
Transfer protocol	HTTP	1.1
Data encryption	TLS	1.3
Encoding	JSON	RFC 8259
Compression	N/A	-
Semantics	SenML	RFC 8428
Ontology	N/A	-

Table 1: Communication and semantics profile for telemetrySensorTurbine service interface

SysML overview description: The SysML diagram for this service shows the interactions between the *Telemetry Consumer*, *Telemetry Provider*, and the *Service Registry*. The telemetry provider exposes three operations: registerTelemetry, getSensorData, and echo. Data flows are modeled as SenML JSON objects over HTTPS/TLS.

2 Service Operations

SysML operation overview description: The operation overview diagram illustrates the telemetrySensorTurbine interface, where: - The *registerTelemetry* operation allows a sensor system to register its data stream. - The *getSensorData* operation allows clients to request telemetry readings. - The *echo* operation is used for service health checks.

2.1 POST /telemetry/register

Operation: registerTelemetry
Input: TelemetryRegisterRequest
Output: TelemetryRegisterResponse

Registers a turbine telemetry sensor, including metadata such as sensor ID, measurement unit, and update interval.

```
POST /telemetry/register HTTP/1.1
Content-Type: application/json

{
  "sensorId": "TURB-AX12",
  "unit": "rpm",
  "updateInterval": 5,
  "location": "Turbine A - Axis 1"
}
```

Listing 1: A registerTelemetry invocation.

```
{
  "status": "registered",
  "timestamp": "2025-10-16 12:00:00"
}
```

Listing 2: A registerTelemetry response.

2.2 GET /telemetry/data?*sensorId={id}*

Operation: getSensorData
Input: SensorDataRequest
Output: SensorDataResponse

Retrieves the latest telemetry measurement from a registered turbine sensor.

```
GET /telemetry/data?sensorId=TURB-AX12 HTTP/1.1
Accept: application/json
```

Listing 3: A getSensorData invocation.

```
{
  "sensorId": "TURB-AX12",
  "timestamp": "2025-10-16T12:00:00Z",
  "value": 1530.4,
  "unit": "rpm"
}
```

Listing 4: A getSensorData response.

2.3 GET /telemetry/echo

Operation: echo
Output: StatusCodeKind

Checks service availability.

```
GET /telemetry/echo HTTP/1.1

200 OK
```

Listing 5: An echo invocation.

3 Data Models

3.1 struct TelemetryRegisterRequest

Field	Type	Description
”sensorId”	Name	Unique identifier of the turbine sensor.
”unit”	String	Unit of measurement (e.g., rpm, ℃).
”updateInterval”	Number	Frequency of data updates in seconds.
”location”	String	Physical location of the sensor.

3.2 struct TelemetryRegisterResponse

Field	Type	Description
”status”	String	Registration result (e.g., registered, failed).
”timestamp”	DateTime	UTC timestamp of the registration.

3.3 struct SensorDataResponse

Field	Type	Description
”sensorId”	Name	Identifier of the sensor.
”timestamp”	DateTime	UTC time when the data was captured.
”value”	Number	Measured value.
”unit”	String	Measurement unit.

3.4 Primitives

JSON Type	Description
DateTime	ISO 8601 UTC timestamp, e.g., ”2025-10-16T12:00:00Z”.
Name	Short alphanumeric string used as identifier.
String	UTF-8 encoded string.
Number	Floating-point numeric value.

4 Revision History

4.1 Amendments

No.	Date	Version	Subject	Author
1	2025-10-16	1.0.0	Initial release of telemetrySensorTurbine IDD	Your Name

4.2 Quality Assurance

No.	Date	Version	Approved by
1	2025-10-16	1.0.0	QA Manager