

**System Description (SysD) Windmill cloud**

**Abstract**

This document provides a black box design overview of the Windmill local cloud system. The system consists of one provider and one consumer.

 1

**System Description (SysD) Windmill cloud** 1

1. System Description Overview 3

2. Use-cases 3

2.1. Behaviour Diagrams 4

When a request to get data from the wm-data service is sent. The service first checks if there are any data points left to return. If not, nothing happens and an error is returned. If there is data available, the service first puts the data into the DataManager and after that is done the service returns the requested data. 5

3. System services 5

3.1. Produced Services 5

3.2. Consumed Services 6

3.3. Sequence Diagrams 6

4. Security 7

4.1. Security Objectives 7

4.2. Assets 8

4.3. Non-technical Security Requirements 8

5. References 8

6. Revision history 8

6.1. Amendments 8

6.2. Quality Assurance 8

# System Description Overview

The system provides one service wm-data, that provides sensory data from a windmill. The data consists of the windmills speed, time since start of measurements, and a time series of the number of Gs measured in the windmills main bearing. The time series consist of 16384 measurements.

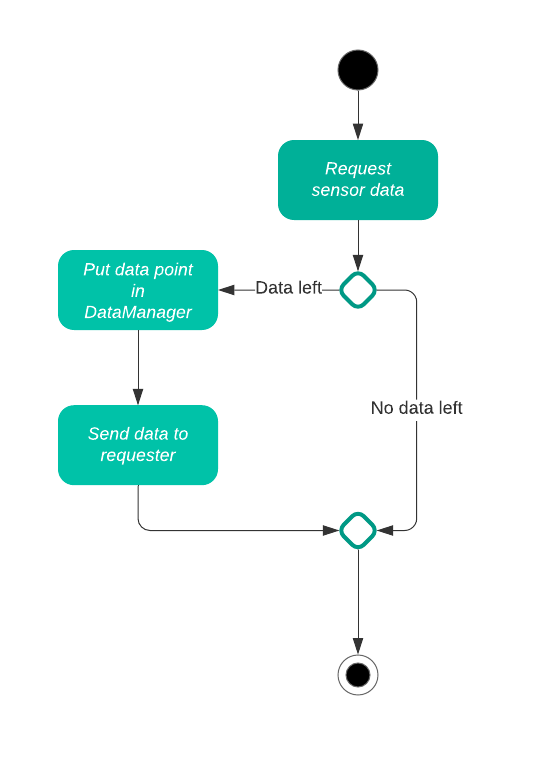
# Use-cases

These are description of typical use cases for the system.

Table 1 Use-case description

|  |
| --- |
| Name of the Use-case: Case 1 |
| **Brief description**:  Consumer request data from the service |
| **Primary actors**:  Monitoring department of windmill farm |
| **Secondary actors**:  Windmill workshop, spare part manufacturing plant. |
| **Preconditions**:  None |
| **Main flow**:  -Provider receives request from a consumer  -Provider performs a request to DataManager’s historian services  -Sensor data gets stored in the DataManager’s database  -Sensor data will be sent to the consumer |
| **Postconditions**:  None |
| **Alternative flows**:  Eliminating the consumer from the flow. E.g. make the process driven by the provider instead when it receives new data from the windmill. (Since I got the data as a whole csv file, I choose to have a consumer drive the process instead.) |

# Behavior Diagrams



When the vm-data service gets a request then the service first checks if there are any data points left to return. If no datapoints exist then nothing happens, and an error is returned. If there is data available, the service first puts the data into the DataManager and after that is done the service returns the requested data.

# System services

This section describes the produced and consumed services by providing pointers to the services Service Description (SD) and the Interface Design Description (IDD)

# Produced Services

Table 2 Pointers to IDD documents

|  |  |  |
| --- | --- | --- |
| Service | SD Document Reference | IDD Document Reference |
| wm-data | Windmill SD path: \documentation\SD\_windmill v2.0.docx | Windmill IDD path: \documentation\IDD\_windmill v2.0.docx |

The services provided reads and provides sensor data from an windmill.

# Consumed Services

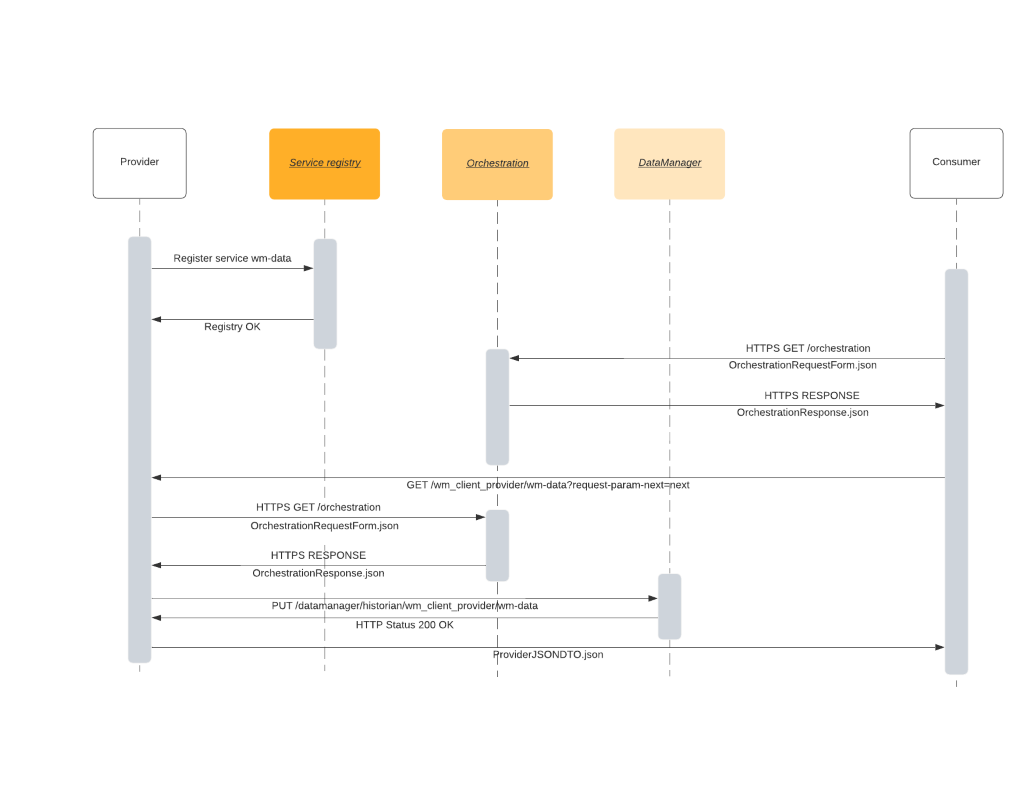
Table 3 Pointers to IDD documents

|  |  |  |
| --- | --- | --- |
| Service | SD Document Reference | IDD Document Reference |
| wm-data | Windmill SD path: \documentation\SD\_windmill v2.0.docx | Windmill IDD path: \documentation\IDD\_windmill v2.0.docx |
| historian | [SD Historian](https://github.com/eclipse-arrowhead/core-java-spring/blob/datamanager/datamanager/documentation/Arrowhead%20Historian%20Service%20G4.1.3%20SD.docx) | [IDD Historian](https://github.com/eclipse-arrowhead/core-java-spring/blob/datamanager/datamanager/documentation/Arrowhead%20Historian%20Service%20G4.1.3%20IDD.docx) |

When the service is consumed, the data gets stored in the DataManager and is then returned to the consumer

# Sequence Diagrams

This diagram shows the sequence of a transaction of sensor data between the consumer and provider, including the DataManager historian service.



# Security

This section defines the security objects of the Windmill local cloud system. It defines the steps take to ensure Availability, Integrity and Confidentiality of the system.

# Security Objectives

The system uses HTTP over TLS to encrypt the requests and transmission. No other payload encryption is implemented currently.

In terms of availability, the system stores the data in the DataManager which always ensures the availability of data. As long as the Arrowhead core services aren’t compromised.

Integrity, data cannot be altered as the system is designed right now. There is a possibility of data loss if an authorized system deletes data from the DataManager, in that case though the system deleting data needs to have the proper security credentials.

Confidentiality, the system relies on TLS for communication between the different services/systems which ensures that data can only be read by authorized subjects.

# Assets

None

# Non-technical Security Requirements

# References

# Revision history

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Date | Version | Subject of Amendments | Author |
| 1 | 2021-01-25 | 1.0 | First draft | Robin Jonsson |
|  | 2021-02-04 | 2.0 | Major update of content | Robin Jonsson |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# Quality Assurance

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