DASS SRS v2.0

Team 51

Abhijit, 2019113032
Gaurav Singh, 2020111014
Kavya, 2019101127
Pratyay Suvarnapathaki, 2020111016

Overview

Project number	50
Project Title	Framework to access and Validate sensor data
Document	DASS Software Requirements Specification Document v2.0
Creation date	15th February, 2021
Created By	Pratyay Suvarnapathaki, Abhijit Srivastava
Client	Prof. Anuradha Vattem, Smart City Living Lab, IIITH

Problem Statement

The Smart City Living Lab has recently deployed more than 100 data collection nodes across the campus. These nodes track various metrics for air quality, water quality, energy, weather, etc. Thus, a large amount of data is being generated.

The natural extension of this hardware framework is to build a robust software setup to ensure accuracy and consistency in sensor readings. Factors such as hardware malfunctions, network failures, or software glitches affect data quality and result in inaccurate data generation.

Therefore, our role is to develop a framework to integrate with the existing oneM2M-based setup, and implement features such as uniform data quality validation, outlier detection, and hardware malfunction indication.

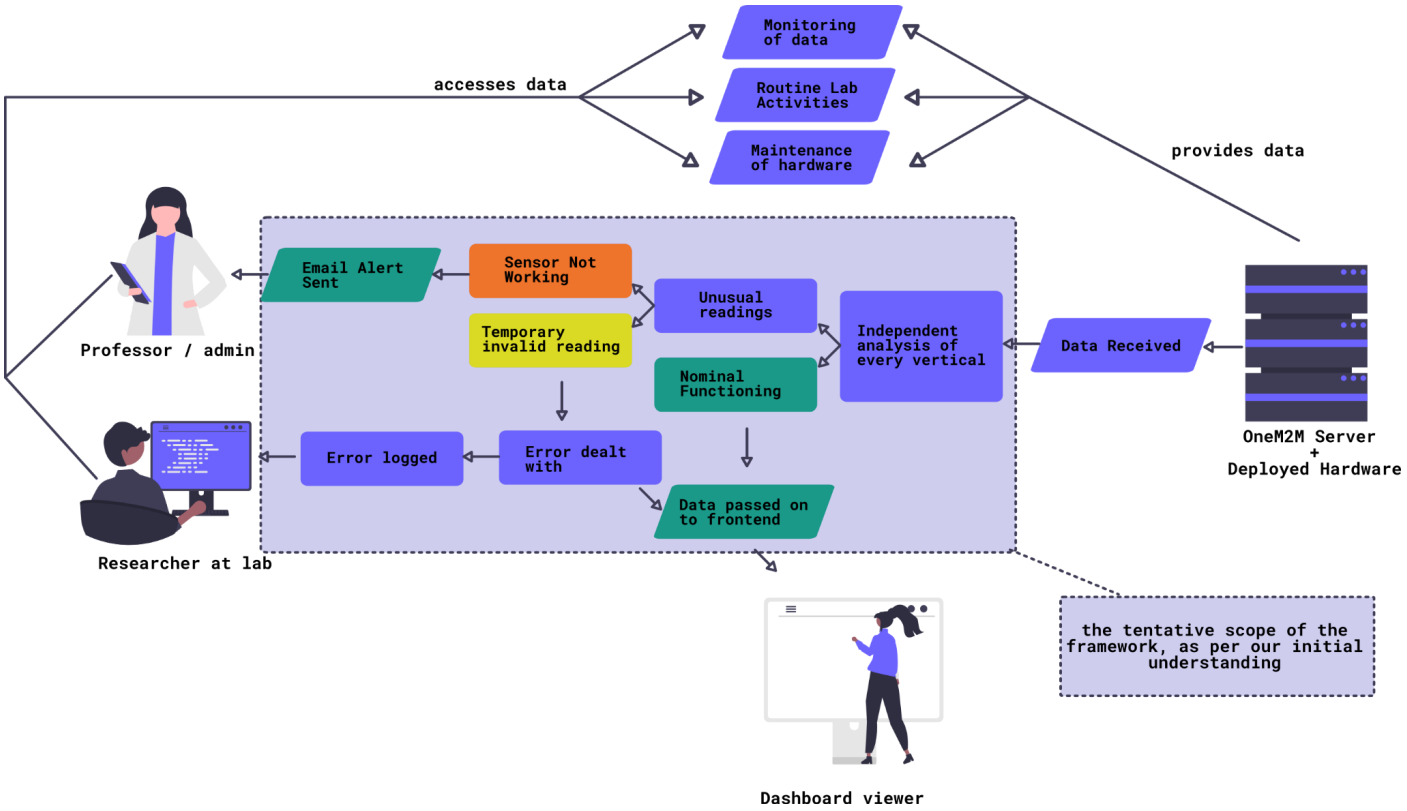
System Requirements

The existing backend framework at the smart city lab utilises a myriad of technologies such as Django, POSTgre, oneM2M, Apachev2.
However, our framework must be a relatively independent system, implemented using python, that runs on the servers of the smart city lab (Ubuntu 18.04, utilising cron jobs), that efficiently performs the requirements. We shall retrieve data from the lab's onem2m setup directly using HTTP, and perform operations on the retrieved data.

User Profiles

The main end users of the framework are the researchers at the Smart City Living Lab.
In essence, they require minimal abstraction and maximum control. They should be able to granularly edit the existing setup of nodes, and be able to conveniently monitor the (predicted) data collection status of each deployed node.
Mentioning the developers of the front-end dashboard as users also makes sense, but since their project is being developed concurrently to ours, it is unclear how much cooperation/integration would be possible/recommended.

Usage Model Diagram



Feature Requirements

For **each of the oneM2M verticals (there are a total of 7)**, the following requirements need to be fulfilled.

No.	Use Case Name	Description	Release
1.	UpAndRunning	Checking whether data is posted at regular intervals or varying time exists	R1
2.	Data Validation	Examine whether node is sending NaN / out-of bounds values (outlier detection)	R1/R2
3.	Alert Sending	Upon detecting unintended/invalid data, an appropriate alert should be sent via email (or via a mobile notification)	R2
4.	Summary health report	Summary for health reports to be sent at a specified frequency	R2

Use Case Description

Use Case Number	UC-01
Use Case Name	UpAndRunning
Overview	Checking whether data is posted at regular intervals or varying time exists
Actors	All the OneM2M data nodes belonging to the Application Entry of the vertical
Pre Condition	Invocation of our framework (this is the fundamental/basic use case)
Main Flow	1. Check if the data nodes are being generated at proper time intervals 2. If they are, proceed to performing further functionality
Alternate Flow	1. If this is not happening, proceed to the alert sending use case
Post Condition	Further processing steps / use cases

Use Case Number	UC-02
Use Case Name	Data Validation
Overview	Examine whether a node is sending NaN / out-of bounds values (outlier detection)

Actors	All the OneM2M data nodes belonging to the Application Entry of the vertical
Pre Condition	The node is up and running
Main Flow	1. Inspect the data being posted to ensure it is not NaN / null 2. Probe the posted data value to make sure it is with some sane boundaries using an efficient detection algorithm. 3. If both these conditions are satisfied, the data is considered to be validated.
Alternate Flow	1. If the data is out of bounds / NaN, proceed to the alert sending use case
Post Condition	Further processing steps / use cases

Use Case Number	UC-03
Use Case Name	Alert Sending
Overview	Upon detecting unintended/invalid data, an appropriate alert should be sent via email (or via a mobile notification)
Actors	The inbox/phone that receives the alert
Pre Condition	Some invalid data / 'broken' node is detected in previous use cases
Main Flow	1. Send an email/push notification, containing appropriately formatted data about the malfunction.
Alternate Flow	1. If there is some failure in sending the email/notification, the error should be properly logged.
Post Condition	Continue monitoring for incoming data

Use Case Number	UC-04
Use Case Name	Summary Health Report
Overview	Summary for health reports to be sent at a specified frequency
Actors	The data collected within the specified long-term time period
Pre Condition	Specified time period has passed since the previous report
Main Flow	1. Generate and store an appropriately formatted summary report containing analytics and statistics about the performance of the nodes over the recent time period.
Post Condition	Start logging data for the next report

An Optional Use Case

In the most recent client meet, prof. Vattem brought up the idea about a possible collaboration with Team 9 (whose project is to redesign and improve the web based dashboard of the lab), wherein provisions could be made to stop the frontend from displaying outlier data.
The details on this are fuzzy, but the description could be:

Use Case Number	UC-05
Use Case Name	FrontEndIntegration
Overview	Notifying the frontend of erroneous nodes so their data is not displayed on the public dashboard
Actors	Dashboard developers, viewers
Pre Condition	UC-03 is invoked
Main Flow	1. Taking the alert sending functionality further, a list of currently-erroneous-nodes shall be maintained in a file that the frontend can access. 2. The frontend shall not display data corresponding to the nodes included in the file
Post Condition	Continue monitoring for incoming data