Embedded Systems and Internet-of-Things
-
Third Assignment

Kimi Osti

February 18, 2025

Contents

Chapter 1

System Requirements

The system is a smart IoT-based temperature monitor. In particular, it measures a closed environment's temperature at any given time, and controls a window connected to a motor to properly ventilate the room in case of critical temperatures. It also implements a manual mode, which can be activated via a button on the Window Controller, or via a web-based Operator Dashboard, that allows an operator to physically control the window opening angle thanks to a potentiometer attached to the Window Controller.

1.1 Temperature Monitor

It's the main system component. It periodically measures the room's temperature, and communicates it to the Control Unit, which is responsible of controlling the other component's behavior according to the valued registered by this subsystem. The frequency of the measurements depends on the state of the system, which is stored in the Control Unit and is communicated to this component in real-time. It's connected to the Control Unit via MQTT, and must include a LED signaling whether the connection is properly established.

1.2 Window Controller

It's the in-place operator interface. It's responsible of physically triggering the window movement, according to the values communicated by the Control Unit if the system is in automatic mode, or according to the value controlled by the potentiometer if the system is in manual mode. It has a button to switch between these two modes, and it also has a screen that tells the

operator the state of the system at any given time. All necessary info is communicated by the Control Unit via Serial communication.

1.3 Operator Dashboard

It's a web-based user interface that allows operators to work remotely on the system. It shows a graph representing the current state of the system and a brief history of the last measurements, sided by a statistic showing the average, minimum and maximum values of the last period of time. In addition to this info, it exposes a simple operator interface which allows the user to switch between manual and automatic mode, as well as to restore the system status in case an alarm was triggered. It communicates with the Control Unit via HTTP in order to reflect user actions on the actual in-place subsystems.

1.4 Control Unit

It's the core of the entire system, and it serves as a mediator between subsystem interactions. Its main function is to track the system state and all info related to the measurements and to the current operating mode (manual or automatic). It also determines the sampling frequency for the Temperature Monitor according to the current measure, and the window opening percentage according to the system state (if in automatic mode). It's also responsible of storing all system data, including the last measurements that the Dashboard shows in its graph, and the average, minimum and maximum values that are shown to the operator.

Chapter 2

System Architecture

This system can be realized following the MVC architectural pattern. In particular, the Temperature Monitor and the Window Controller are the Model (since they sense real-world values and actuate the system behavior on the real application domain), while the Control Unit serves as the Controller (tracking the system state and storing user-relevant data) and the Operator Dashboard serves as the View, presenting data to the user and offering an interface to reflect user actions on the Model.

2.1 Temperature Monitor

This component can be modeled as a Finite State Machine. Its main duty is to periodically sample the room's temperature, and to send the collected data to the Control Unit. The sampling frequency varies according to its state, which depends on the measured temperature itself. It also implements a default behavior when the system is in manual mode, and eventually shows network errors when they occur. When the network goes down, it attempts reconnecting in order to restore its previous state.

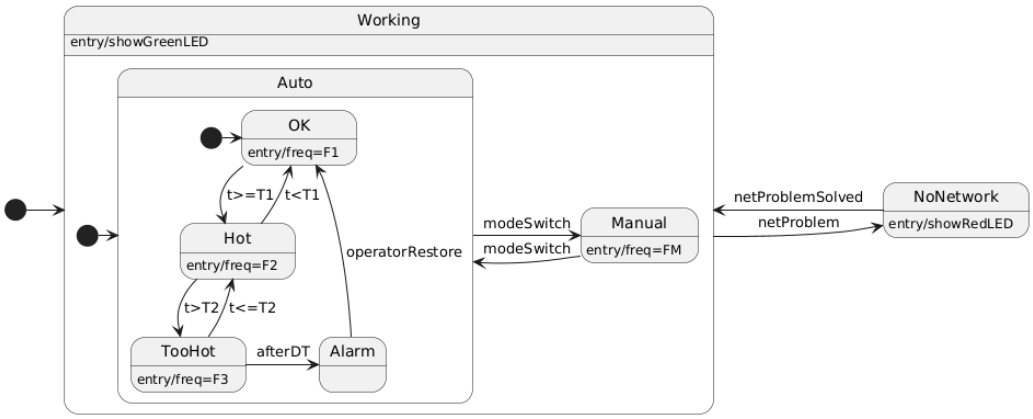

Figure 2.1: FSM modeling the Temperature Monitor behavior

2.2 Window Controller

Also the Window Controller can be modeled as a Finite State Machine. Its behavior depends on whether the system is in automatic or manual mode, independently from the Temperature Monitor Measurements. Since it exposes a control panel, as a Model component, it's not properly adhering to the MVC architectural pattern. On the other hand, though, it can be considered Model since all the actions performed by the control panel can be handled internally, communicating to the Controller only state transitions. So, it can be thought as a Model component to simplify and clarify the architectural software structure.

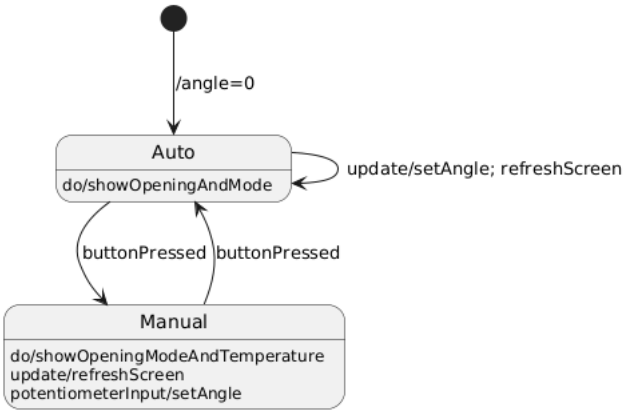

Figure 2.2: FSM modeling the Window Controller behavior

2.3 Operator Dashboard

The Operator Dashboard serves here as a remote user interface to control the whole system. It offers a graphical interface in order to present to the user relevant data with a certain clarity and processing, in order to simplify their operations. It's based on a web interface, and shows various components, which can represent plain output or be associated to input commands. In particular, it offers the equivalent of the Window Controller in-place panel, with a button to switch between manual and automatic mode, and it adds to that a button to register that an alarm situation has been restored. It communicates via HTTP to the Control Unit, which is then responsible of reflecting user actions on the Model.

2.4 Control Unit

This subsystem serves as the application Controller. It exchanges data with all subsystems, storing internally what's relevant for the user in order to represent it properly, and ensures the user input is effective on the real-world domain. It's the core of the application, and therefore tracks the state of the other subsystems, and bridges among the various communication protocols (in this case Serial, HTTP and MQTT) to allow effective data exchange between subsystems. It will be made of various components itself, and its main goal is to decouple the other subsystems, trying to minimize inner coupling between its own components.

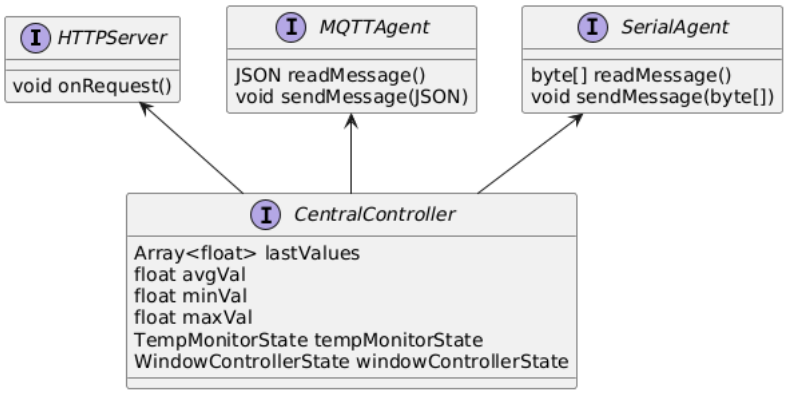

Figure 2.3: UML lass diagram showing the architecture scheme for the Control Unit sub-components

Chapter 3

Implementing Choices

3.1 Temperature Monitor

3.2 Window Controller

3.3 Operator Dashboard

3.4 Control Unit