Thermostat

Software Engineering for Autonomous Systems

Academic Year 2018/2019

Stefano Di Francesco

Fabio Di Silvestro

Stefano Di Francesco
Fabio Di Silvestro

Table of contents

Stefano Di Francesco
Fabio Di Silvestro

GOAL

The system must keep the internal temperature of the home above a user-defined threshold but at the same time reduce the energy consumption.

Monitor

It subscribes to an MQTT broker and intercepts values coming from specific topics. Thanks to the topic, the monitor component knows which sensor is publishing data. It can also listen for new sensors added at runtime.

- *Listening for temperature values coming from sensors inside every room.*

- *Listening for values coming from presence sensors inside every room*: it assumes there are people in the room if there is at least 1 person.

- *Filtering faulty values:* it checks if the data sensed lay in a predefined threshold.

- *Stores to the knowledge.*

Analyzer

- *Estimation of temperature trend for every room*
  It lets the planner know if the temperature is falling or increasing by calculating the trend slope using the most recent values of temperature. Using the temperature trend, we take into account any external factor that could affect the internal temperature, as an open window or a sudden fall of external temperature.

- *Prediction of people presence in a room*
  For each day of week, the prediction algorithm looks back to the presences of the same days of the last two weeks. These data are aggregate in order to create a list of time intervals reflecting when the people are expected to be in the room.
  For example, if today is *Monday* it will selects all the values of presences of the two previous Mondays, those values are binary digits, so it will identify every state change and write an interval using the timestamps of these states.

Planning

Following the basic principle of feedback loop, this algorithm will choose at every cycle the best action to take. It is executed for every room independently. The analyzed data are extracted from the knowledge or directly from the Analyzer instance.

Stefano Di Francesco
Fabio Di Silvestro

Follows the flow diagram of the planner behavior:

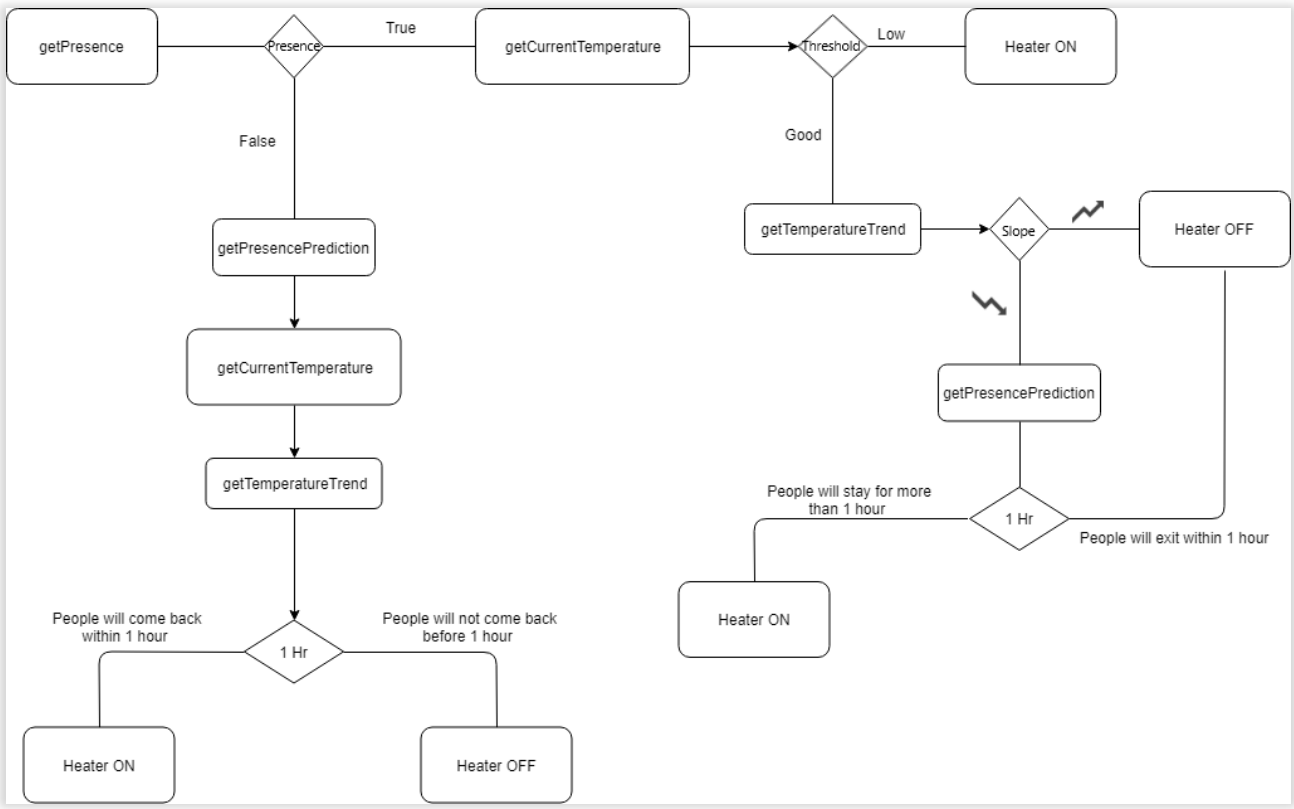

Figure 1 - Planning algorithm

Constraints

- If there are people in the room, the temperature must always be over the preferred threshold of that room.

- If the room is empty, the priority is to save energy consumption.

The algorithm takes as input the target temperature (from MQTT broker), the current temperature, the temperature trend and the list of intervals of time that the Analyzer predicted for the current day.

- The presence prediction is used both for calculate how much time is left before someone will enter a specific room and the time remaining before the room will be empty.
- This behavior allows us to improve the energy consumption since the heater is turned on only when it is strictly needed.
- On the other side we can power off the heater earlier basing the decision on the knowledge generated by the Analyzer.

It can be improved using the exact value of the trend slope in order to establish the time remaining before the temperature goes below the threshold in combination with the time remaining before the presence of people change, then it will know when to turn on the heater, if necessary, so that the environment will be compliant to the roomers when they are back or until they are inside.

If there are not yet presence prediction data, the algorithm assumes people could suddenly come back when executing the left branch, while it will prioritize the energy saving for the right branch.

Executor

- It provides the API to control the actuators through the MQTT broker. Every actuator takes either ON and OFF (case sensitive) as value.

Additional components

MQTT broker

All the used topics have *home*/ as root

- *Simulator*
  - publishes to "home/*sensor_type*/*room*/values"
- *Sensors*
  - publish to "home/sensors/*room*/*sensor_type*"
  - subscribe to "home/*sensor_type*/*room*/values"
- *Monitor*
  - subscribe to "home/sensors/#"
- Planner
  - subscribe to "home/thresholds/#"
- Executor
  - publishes to "home/*room*/heater"

Sensors

In order to sense and send a datatype, we need to start a dedicated sensor. The configuration of the component is done by a config file. Since the sensors publish data on a specific topic of the MQTT broker, we have the possibility to add new sensors directly at runtime without restarting the overall system.

Stefano Di Francesco
Fabio Di Silvestro

openHAB panel

The openHAB panel shows the current temperature and the state of the actuators.

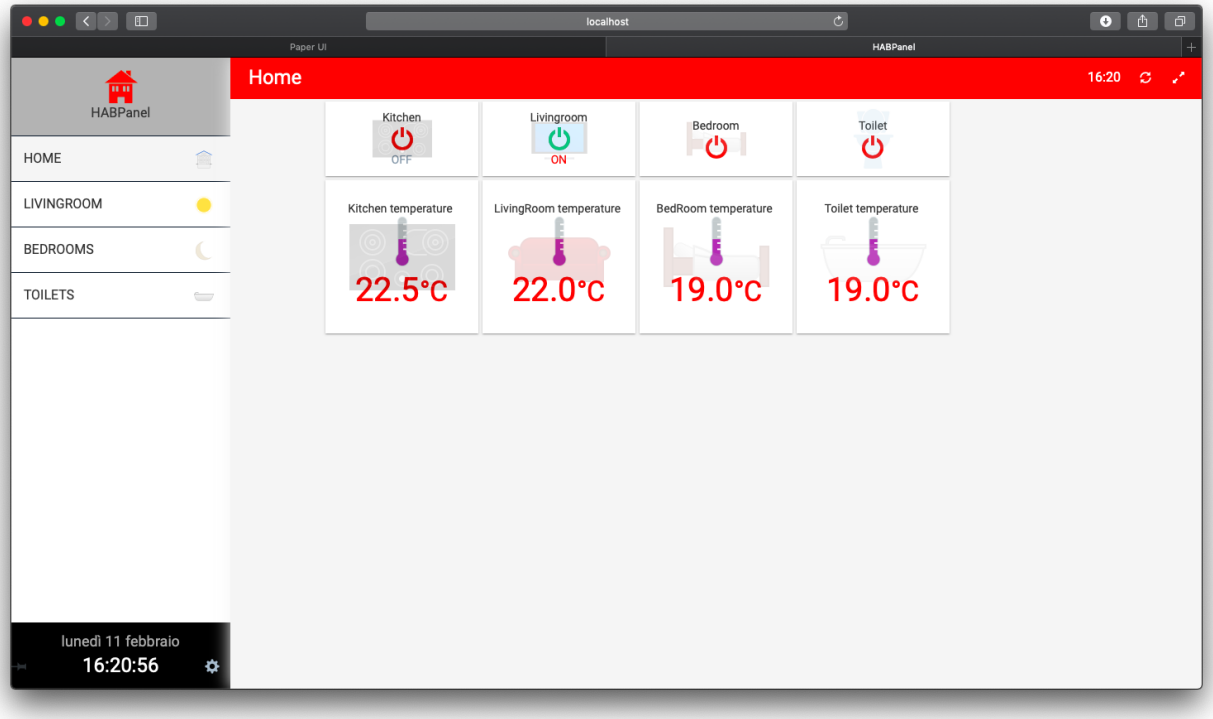

Figure 2 – openHAB

The user can define the preferred thresholds through the detailed view of the zones of the home: Living room, Bedrooms, and Toilets.

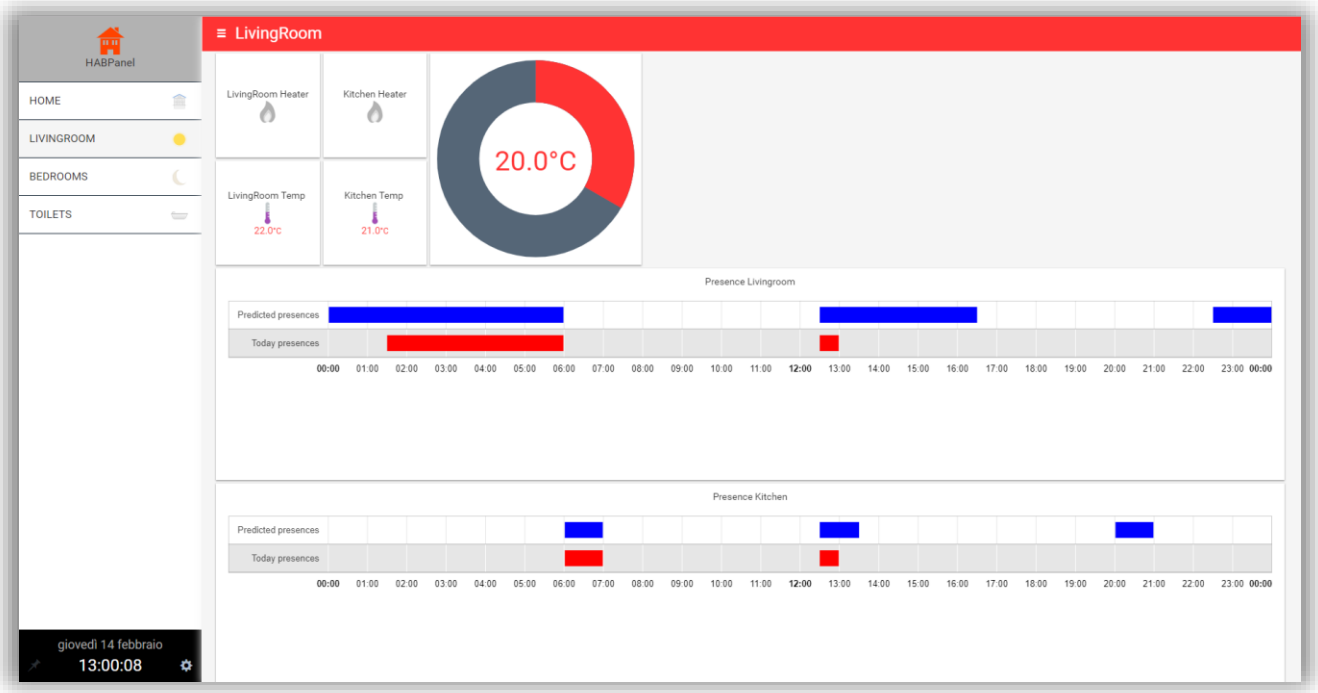

Figure 3 - Specific zone

Simulator

Simulates the environment in which the sensors will sense values.
It lets us change the current temperature of every room and the current presence of people.
It updates the values every 5 seconds.

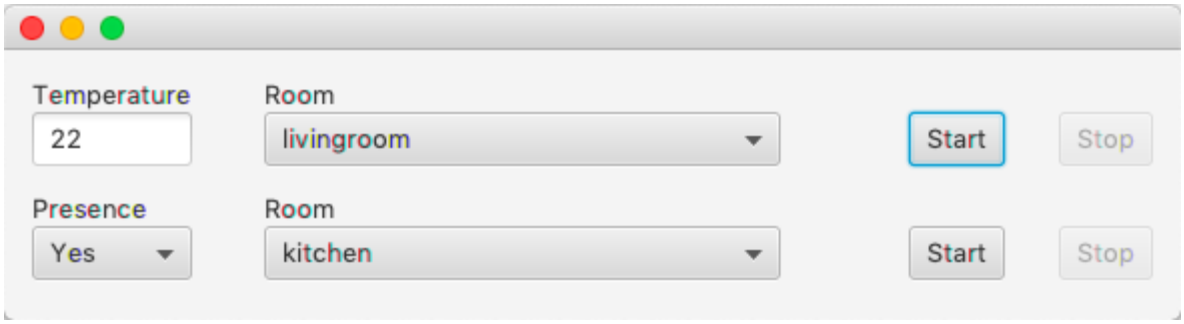

Figure 4 - Simulator

Technologies

- Java Development Kit 8
- OSGI Framework
- JavaFX
- Mosquitto
- OpenHab 2 (copy 'jsondb' folder from repository to your OpenHab installation, start OpenHab, then install 'MQTT Binding' addon)
- PHP interpreter (for custom widget of OpenHab showing presence history on a timeline. **PORT 8888**)
- MySQL (create a database called '*se4as*' , then import the dump. User and password settings can be changed in SQLManager project)

Stefano Di Francesco
Fabio Di Silvestro

Component Diagram

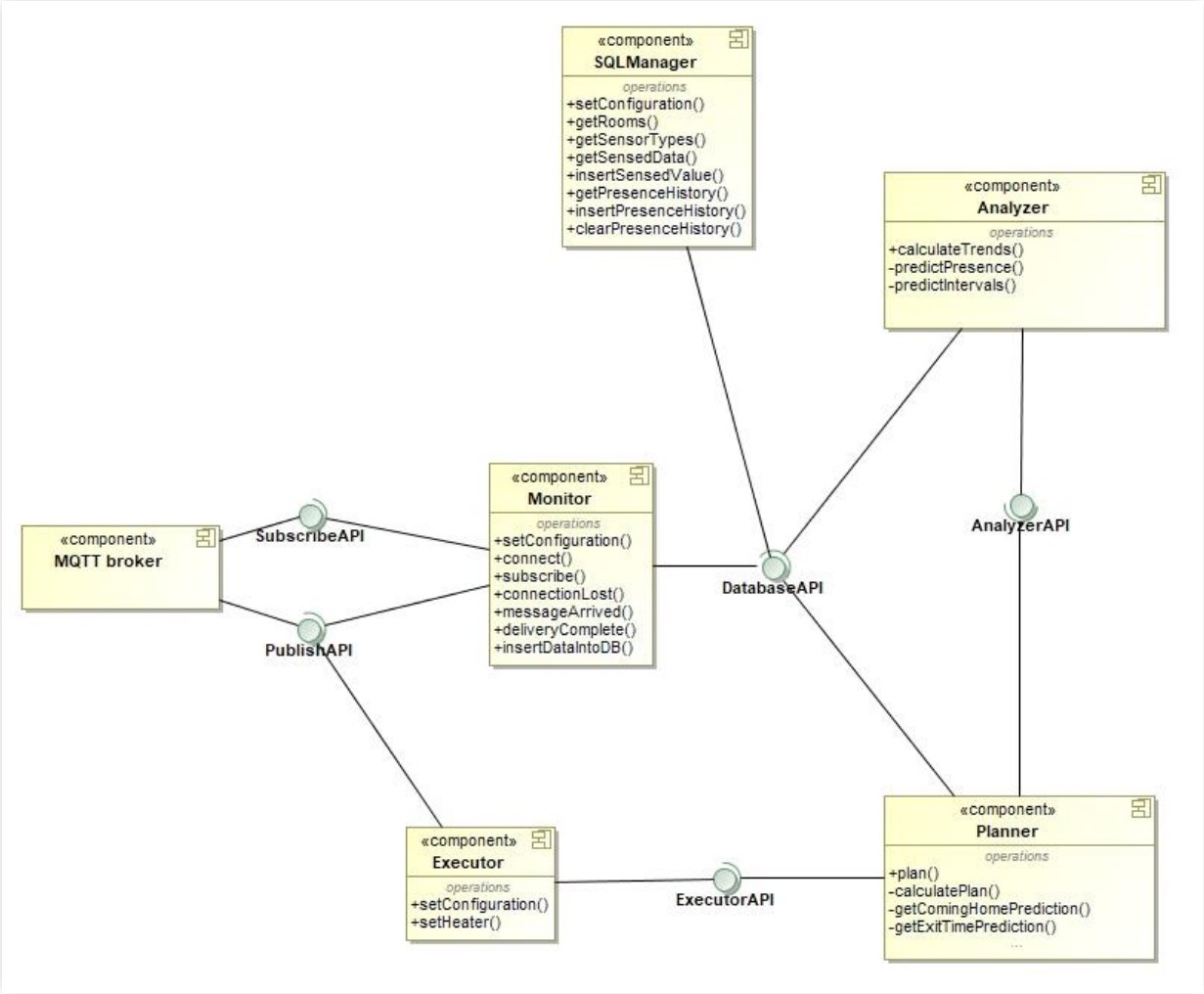

Figure 5 - Component Diagram