Autonomous Systems: The Smart Home

Department of Information Engineering, Computer Science and Mathematics

University Of Study Of L'Aquila, Italy

Submitted to Professor **Davide Di Ruscio**

Course Titled **Software Engineering for Autonomous Systems**

Date	01/02/2022
Team Name	MGM

Team Member	M. Number	Email Address
Martina Nolletti	286160	martina.nolletti@student.univaq.it
Michele Intrevado	284789	michele.intrevado@student.univaq.it
Giacomo Sfratato	282330	giacomo.sfratato@student.univaq.it

Summary

1. Introduction

An autonomous system combines sensors and control systems to enable complex sequences of operations that can be performed on different types of systems. The need to develop systems of this type arises from the need to be able to manage increasingly complex and constantly evolving systems. The advantage of this type of application lies mainly in the possibility of operating in environments without the aid of man or which require constant monitoring over time.

Despite the numerous advantages and benefits of such systems, various 'uncertainties' exist during their development. For example, one of these could be the danger related to their possible failures such as damaging an element managed by the system or the loss of human lives.

In general, any system is defined as autonomous if it respects the MAPE-K Loop, which describes the flow of data between its various components. Our MAPE-K loop is defined later.

The developed application represents an autonomous management system of a house.

In particular, the trend of the following factors was simulated: internal temperature (from the insulation of the internal environment), internal humidity, internal light intensity, and the presence of people in the house.

Furthermore, the behavior following the activation of the various actuators which will modify the environmental situation inside the house, in relation to the objectives to be achieved, was simulated. In particular, the following have been simulated: an air conditioner (capable of increasing and decreasing the temperature), a dehumidifier (capable of increasing and decreasing the percentage of humidity), a sound and light alarm (capable of activating in certain situations), and intelligent lighting (able to adjust the brightness inside the house).

The advantages of building such a system lie in the possibility of automating human behavior, to avoid errors resulting from distractions or negligence.

Chapter 2 describes the objectives to be achieved using this application. In Chapters 3 and 4, on the other hand, the technologies used and the functions of the system are respectively described. Chapter 5 describes the modalities that the system can assume. In Chapters 6 and 7 the implementation of the MAPE-K loop and of the various components of our system were respectively described. Chapter 8 describes the implementation of the obtained graphs and tables in Grafana. Finally, the conclusions and instructions are presented in chapters 9 and 10.

2. Goals

We set out to develop an autonomous system for automated homes. Sensors have been identified within our system, such as:

- **Internal air humidity sensors.**

Humidity is one of the fundamental factors inside a house, and consequently, it too must be constantly monitored. It is affected by the dehumidifier. Using the latter, the internal humidity is decreased/increased when necessary and correctly.

- **Internal temperature sensors.**

Temperature is another of the most important factors inside a home. If this were to increase or decrease too much, there would be many problems, therefore it is an element to be kept in constant monitoring.

One type of actuator influences this quantity: air conditioners. If the air conditioners are active, the temperature tends to increase or decrease according to the set power.

- **Internal brightness sensors.**

The internal light intensity is a factor to take into consideration because, if it is daytime, it is not necessary to have the lights on and on the contrary, if it is night-time, it is necessary to turn on the lights. For this purpose, smart lighting is used, which allows you to dim/increase the light.

- **Motion/presence sensors in the house.**

Presence sensors make it possible to check if someone is in the house.

In addition, several actuators have been identified, such as:

- **Conditioners.**

Physical system (simulated) able to increase/decrease the internal temperature of the house.

- **Dehumidifiers.**

Physical system (simulated) able to increase/decrease the humidity inside the house.

- **Sound and light alarm.**

Physical system (simulated) able to detect dangerous situations inside the house.

- **Smart lighting.**

Physical system (simulated) able to increase/decrease the lights of the house.

3. Used technologies

The software has been completely implemented through the Python programming language and the use of the following technologies:

a) MQTT

The protocol used by the Mosquitto broker is MQTT. We used this messaging protocol to get data from sensors. Then we loaded the data via python to send it to the other components where the data can be fetched and processed from.

Our published topics include:

- indoor/nameRoom/light
- indoor/ nameRoom/temperature
- indoor/ nameRoom/humidity
- indoor/ nameRoom/movement

Where nameRoom can be: livingRoom, bathRoom, kitchen, bedRoom.

b) InfluxDB

InfluxDB is used to store the continuous flow of data coming and going through python files. The main benefit of using InfluxDB is the ease with which data can be sorted and found.

influxdb

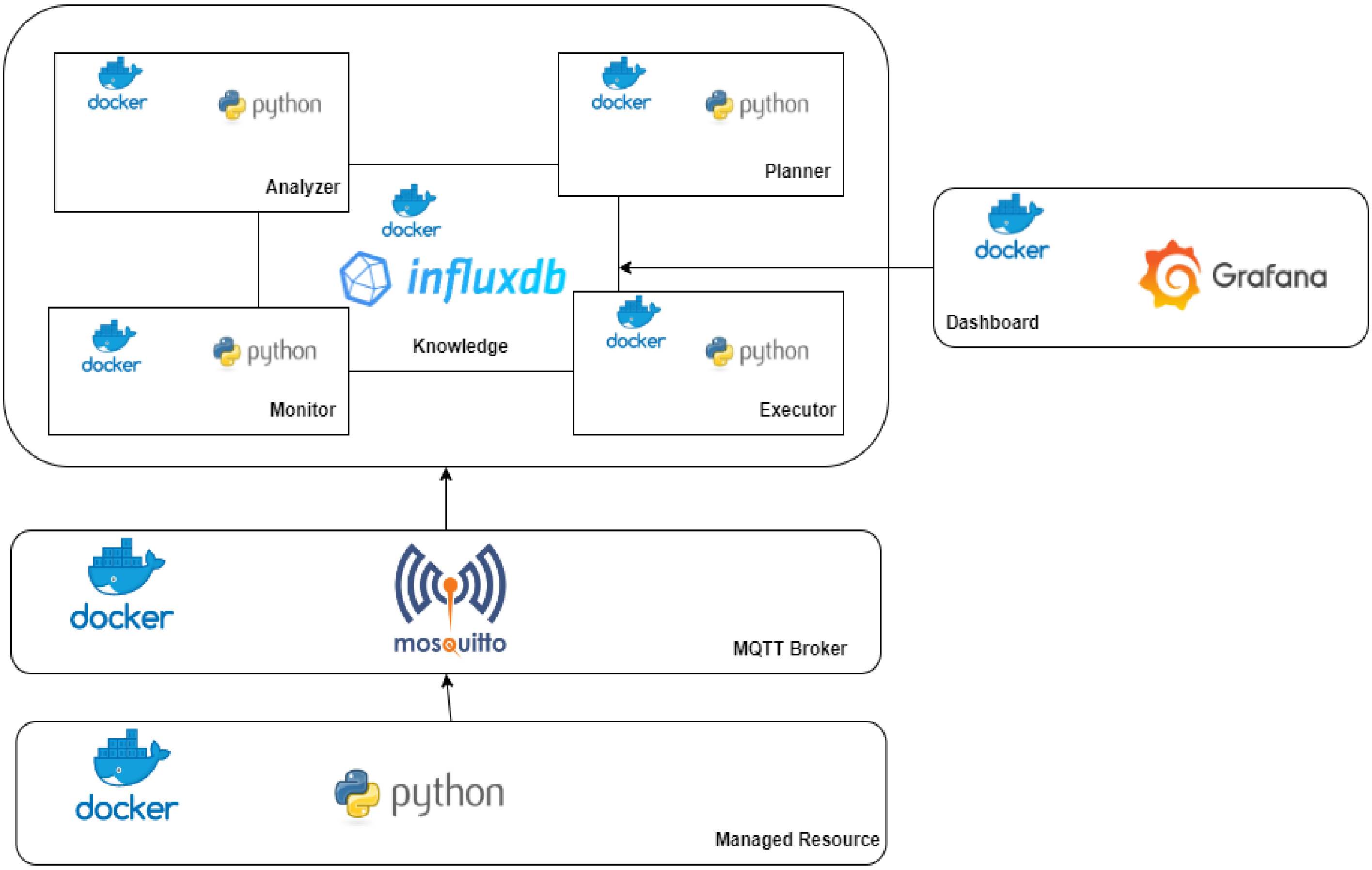
c) Grafana (Dashboard)

We used Grafana to visualize and understand the data. The main benefit of Grafana that we found was that, in addition to providing better visualization, it provides a way to create multiple dashboards at once which allowed us to better manage the information.

Grafana

The MQTT messaging protocols have been used to exchange information between the system and InfluxDB. While Grafana was mainly used as a graphical interface and as a tool for interacting with the system.

4. System Architecture



The diagram above illustrates the system architecture.

5. System functionality

Responsive functionality was implemented during development. It brings together all those operations which allow the system to react to non-optimal situations which may occur inside the house during the day.

5.1 Reactive

- **Activation of Dehumidifier based on Humidity**: to regulate the process of increase/decrease of humidity inside the home, the dehumidifier is activated/deactivated.
  If the internal humidity exceeds the optimal threshold and the dehumidifier is active, it is switched off. If the indoor humidity is below the optimal threshold and the dehumidifier is off, it is switched on.
- **Activation of Air Conditioner by Temperature**: the air conditioner is switched on/off to regulate the heating/cooling process of the house.
  If the internal temperature exceeds the optimal threshold and the air conditioner is on, it will be switched off. If the internal temperature is below the optimal threshold and the air conditioner is off, it will be turned on.
- **Activation of Smart Lighting**: If the light intensity is too strong inside the house, the smart lighting actuators are deactivated, in order to dim the light. Otherwise, if the light inside the house is too low, the smart lighting is activated.
- **Activation of Sound and Light Alarm**: If there is a dangerous situation inside the house, the alarm will detect this particular condition and activate it.

6. System mode

System modes consist of the energy settings we have available; they are set automatically by the system. All the applicable modes and their characteristics are described below.

- **ECO.**

It consists of the energy saving plan, as well as the default mode of the application when no one is home. With ECO mode active we limit the activation of actuators such as air conditioners, dehumidifiers, and smart lighting using a larger range of tolerance.

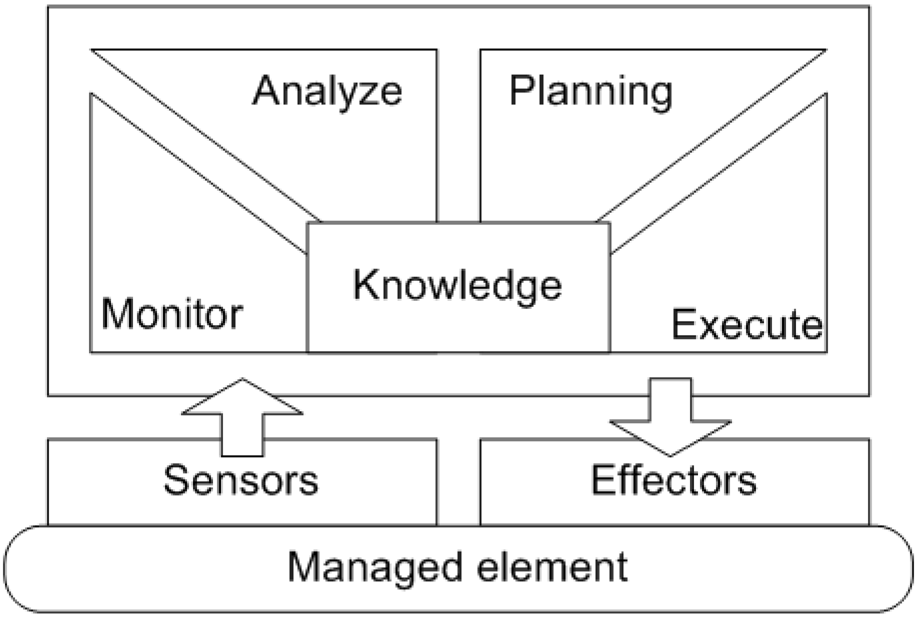
- **Normal.**

This mode, in a real environment, is on average more expensive than ECO mode in terms of consumption, but at the same time it manages to bring the system back to an optimal range more quickly and efficiently.

- **Danger.**

Danger mode is automatically set by the system itself if a critical situation is found. If inside the house some factors have exceeded the range of danger, the system will immediately set this mode and implement the corresponding plan to restore an optimal situation. As soon as the optimal state is reached, the application will automatically reset the ECO mode.

7. MAPE-K Loop Implementation



Our system is based on the MAPE-K Loop, as defined in Paragraph 1, which describes the data flow between the various components:

- **Monitoring**: Component dedicated to recording data such as indoor brightness, humidity, and temperature.
- **Analysis**: Component that compares event data with knowledge base models to diagnose hypothetical dangerous situations and store them, but mainly correlates incoming data with historical data and acts accordingly.
- **Planning**: Component that considers the data monitored by the sensors to produce a series of changes to be made on the managed element. Interpret dangerous situations and/or currently available data to develop a plan, decide on an action plan and implement policies.
- **Execution**: Component that executes the change of the process managed through the actuators and executes the plan.
- **Knowledge**: Component that saves data, such as the days and time slots in which the house is most populated to track people's habits, to which all the other components refer.

8. Implementation of Components

MANAGED RESOURCE

We used Python and defined the ManagedResource directory to implement the rooms and sensors inside the house (for rooms we defined living room, bathroom, bedroom and kitchen and for sensors we defined temperature, humidity, light and motion). Next, we defined the actuators to increase/decrease the light inside the house with the smart lighting, decrease the humidity with the dehumidifier, increase/decrease the temperature with the air conditioner and deactivate/activate the alarm with the sound and light alarm. We also implemented the system mode, established the connection with the InfluxDB database and entered the mode types and target values here: ECO, NORMAL and DANGER. Finally, for this folder, we defined the main file, where we recalled all the defined files and connected the client to MQTT. We simulated the value of the sensors for each location we needed.

MONITOR

We defined the directory Monitor, and we establish the connection with the database InfluxDB, defining bucket, organization, token, and URL. Then we formatted and stored the data inside the DB. Finally, we connected the client to MQTT.

ANALYZER

We defined the directory Analyzer. Based on the measurement made the Analyzer assumes a specific purpose. Regarding the movement, the time slots in which the house is inhabited are checked in order to profile people's habits. For the sensors of light, temperature, and humidity the values are calculated and any symptoms are identified to be sent to the planner.

PLANNER

Then, we created the directory, Planner. Receives the HTTP messages sent by the Analyzer and based on their content chooses the changes to be made that will be communicated to the executor.

EXECUTOR

Finally, we defined the directory Executor. It sends the managed resource the operations it will have to perform.

9. Data Visualization With Grafana

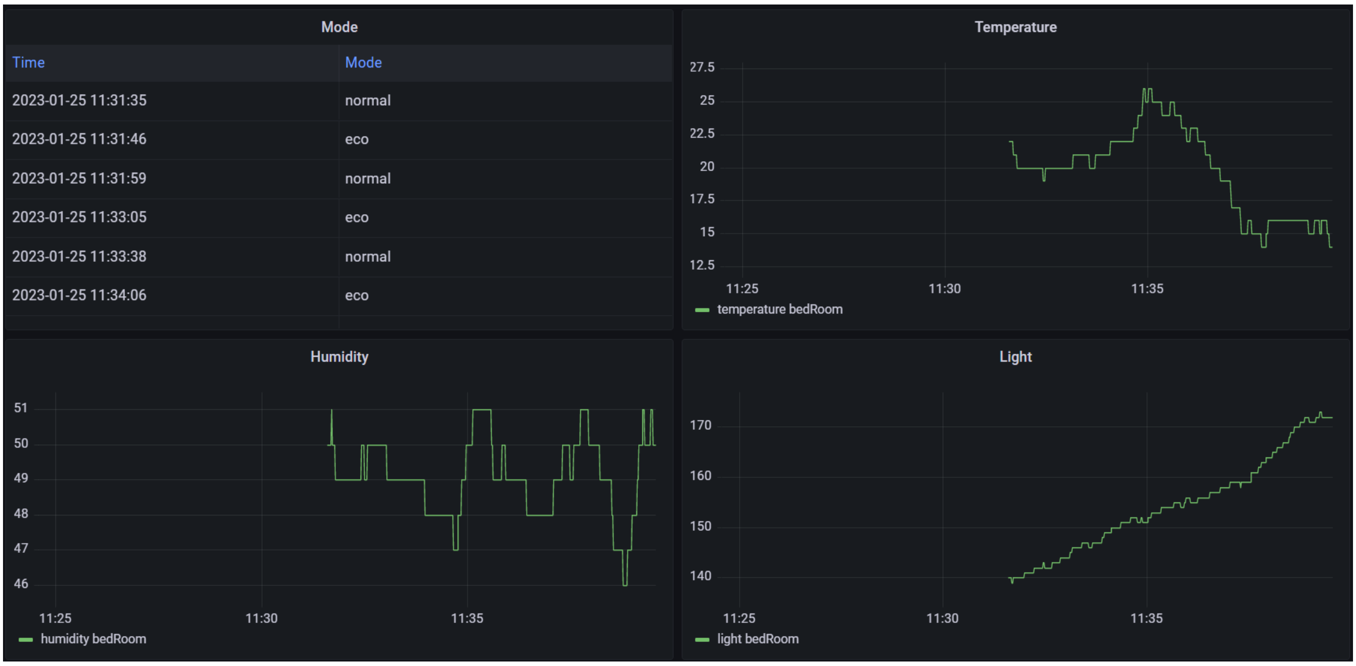
In Grafana we defined the table and charts for the house. In particular we have 4 dashboards:

- **Bathroom.**

In this section, we defined the graphs for temperature, humidity, and light, and we insert also the mode of the room: Bathroom.

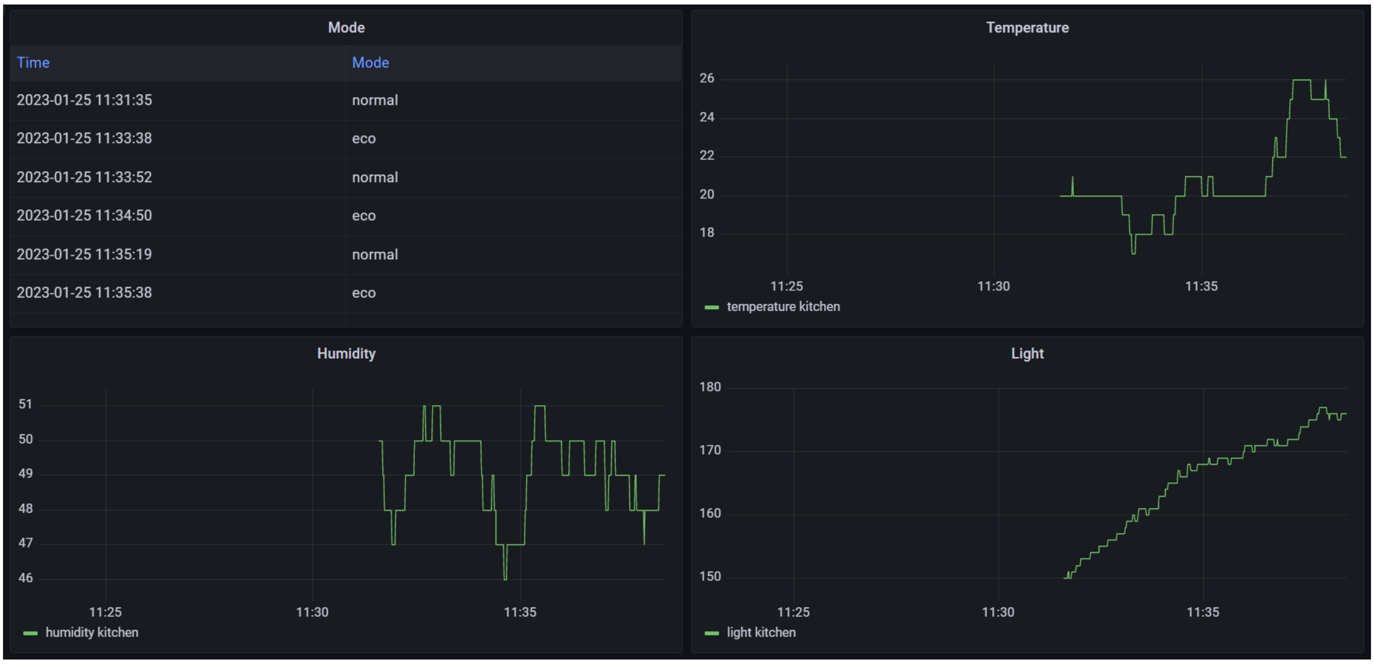
- **Bedroom.**

In this section, we defined the graphs for temperature, humidity, and light, and we insert also the mode of the room: Bedroom.



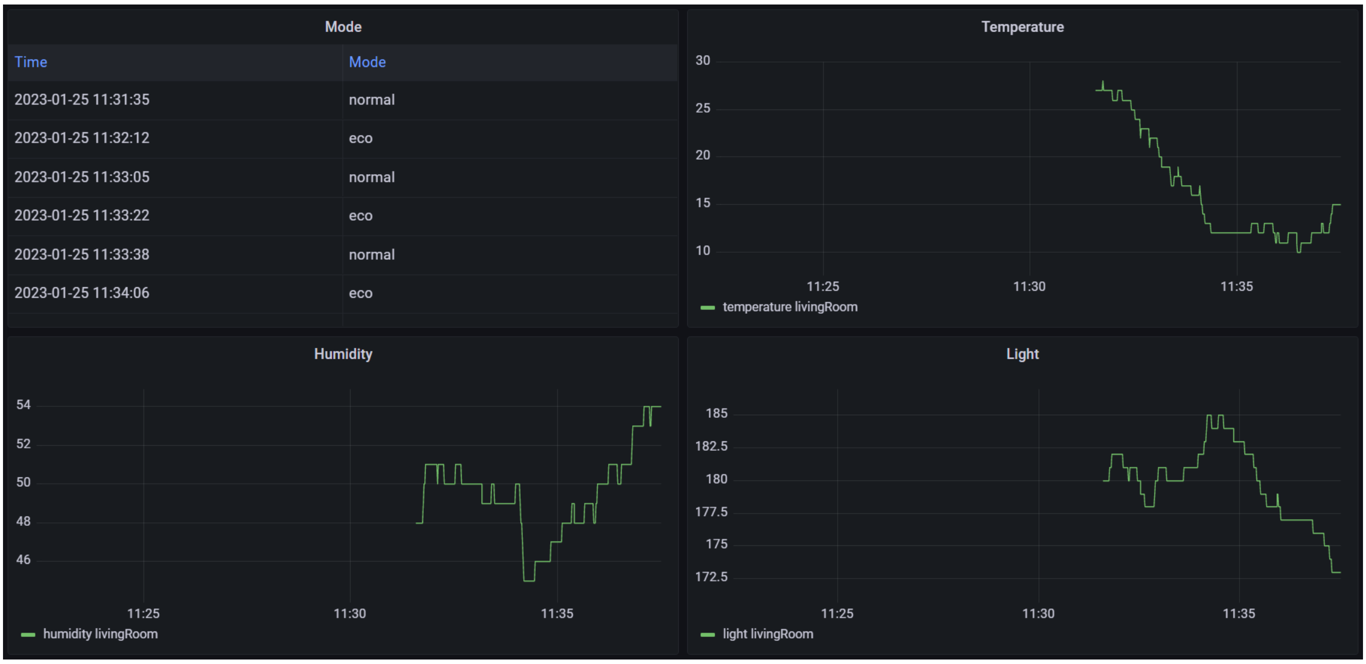
- **Kitchen.**

In this section, we defined the graphs for temperature, humidity, and light, and we insert also the mode of the room: Kitchen.



- **Livingroom.**

In this section, we defined the graphs for temperature, humidity, and light, and we insert also the mode of the room: Livingroom.



10. Conclusions

Our system is capable of autonomously taking the appropriate initiatives for the current situation.

In the reactive phase, the application makes efficient choices based on the analyzed context, respecting the times necessary for the regular functioning of the system.

Having said that, we can therefore say that our application meets all the requirements necessary for the creation of an autonomous system within the specified context.

11. Instructions

- git clone https://github.com/Martina99188/SE4AS.
- In the cmd of the directory run -> docker compose up.
- Only at the first running you need to access Grafana (admin:admin) (localhost:3000).