R.E.C.A.M.

Renewable **E**nergy **C**ommunity **A**utonomic **M**anager

2024/25

Software Engineering for Autonomous Systems

Prof. Davide Di Ruscio

Realized by:

Daniele Borgna - daniele.borgna@student.univaq.it
Omar Dinari - omar.dinari@student.univaq.it

Table of Contents

1. Introduction

The Renewable Energy Community Autonomic Manager (RECAM) is an autonomous system designed to efficiently manage renewable energy resources, such as solar panels and wind turbines, within Renewable Energy Communities (REC). These communities consist of individuals, businesses, or public entities working together to produce, share, and manage renewable energy at a local level.

The primary goals of RECAM are to maximize self-consumption, minimize dependence on external energy sources, and ensure equitable energy distribution among community members. The system incorporates mechanisms to manage specific "optional" electronic devices within the community, such as appliances or lights, activating them only when sufficient energy is available. If energy is insufficient, users will be notified that certain devices could not be activated for the day. However, devices marked as critical will still be activated, possibly utilizing external energy sources.

RECAM employs a proactive strategy to balance energy production, consumption, and storage. A key feature of the system is its **predictive model**, which forecasts energy trends to optimize resource management.

All functionalities of the system will be elaborated in the requirements section, with priorities assigned to identify the most essential features. Additionally, the technologies employed will be detailed in a dedicated section.

2. Goals

As previously mentioned, the primary objective of the system is to optimize energy usage by:

- Fulfill all the members' requirements.
- Maximizing self-consumption of renewable energy.
- Minimizing reliance on external energy sources.

2.1. Indices

To implement these goals, we define some indices used by the system.

Name	Variable
Number of members of the REC	K
Members of the REC	$m_1 \ldots, m_K$
Producer devices for the member i	$p_{i,1},\ldots, p_{i,N_i}$
Consumer devices for the member i	$c_{i,1},\ldots, c_{i,M_i}$
Instantaneous production in kW of a producer device	$\pi(p_{i,n})$
Consumption in kW of a consumer device	$\omega(c_{i,m})$
Requested time in minutes of a consumer device	$\tau(c_{i,m})$
Requested deadline in minutes for a consumer device	$\delta(c_{i,m})$
Energy in kWh stored in batteries	s

Explanation

Each member m_i of the REC contributes to energy generation through production devices $p_{i,n}$, each of which has a maximum instantaneous production of energy in kW equal to $\pi(p_{i,n})$. The energy that each producer generates is stored in batteries and the total amount of energy stored in kWh is equal to s. Each member also owns consumption devices $c_{i,m}$, each of which consumes a power amount in kW equal to

$\omega(c_{i,m})$. Assuming these devices are smart and can be remotely activated without human intervention, each member will specify a required usage time for each device $\tau(c_{i,m})$ and a deadline in minutes $\delta(c_{i,m})$ within the time has to be satisfied. Consequently, the daily energy consumption in kWh required for the device $c_{i,m}$ will be equal to $\tau(c_{i,m}) \cdot \omega(c_{i,m})$. The deadline in minutes represents the number of minutes within which the required usage time for a given consumer must be met. For example, if a consumer is assigned $\tau = 60 \cdot 2$ and $\delta = 60 \cdot 6$, it means that within 4 hours the device must start, regardless of the energy availability in the battery.

2.2. Goal Formulation

Once all the indices of the system have been defined, we can mathematically represent the two formulas that represent the goals of the system. The two formulas are written in **order of priority** and the first one addresses the requirements of the members:

$$\tau(c_{i,m}) \leq \delta(c_{i,m}) \; \forall \, c_{i,m}$$

This implies that for each consumer, the requested usage time must be fulfilled before the specified deadline. The second formula encapsulates two goals and is as follows:

$$s - \sum_{i=0}^{K} \sum_{m=0}^{M_i} \tau(c_{i,m}) \cdot \omega(c_{i,m}) \rightarrow 0$$

This formula reflects both goals because:

- **When** $s > \sum_{i=0}^{K} \sum_{m=0}^{M_i} \tau(c_{i,m}) \cdot \omega(c_{i,m})$): Approaching zero means that all the stored energy needs to be consumed.
- **When** $s < \sum_{i=0}^{K} \sum_{m=0}^{M_i} \tau(c_{i,m}) \cdot \omega(c_{i,m})$): Approaching zero ensures that the use of external energy is minimized as much as possible.

3. Managed Resources

As mentioned in the previous chapter, the system consists of various elements, including REC members, energy producers $p_{i,n}$, energy consumers $c_{i,m}$, and batteries. To properly define the system, we will identify which components serve as sensors and which function as actuators.

3.1. Sensors

The sensors used by the system are:

1. **Battery level sensor**: A sensor that measures the energy stored in the REC batteries.
2. **User dashboards**: Dashboards that collect temporal data regarding the consumers, specifically τ (time of use) and δ (deadline).
3. **Energy production sensor (optional)**: A sensor that measures the energy production of each producer. This sensor is optional, as the application primarily requires measuring the energy stored in the battery, which aggregates all the energy produced by the producers. However, it is useful to monitor this data to provide users with an overview of the producers' energy production.

3.2. Actuators

The system includes a single type of actuator: a switch for each consumer. This switch is responsible for starting the operation of the respective device. Each consumer is equipped with its own dedicated actuator, ensuring individual control. Once a device is activated, it is assumed that its operation will continue uninterrupted until the requested usage time is fully satisfied. This design ensures simplicity and reliability in managing the execution of the devices while adhering to the system's operational constraints and goals.

4. Requirements

This chapter provides a detailed overview of all the system requirements, categorized into functional and non-functional requirements. Functional requirements define the specific behaviors and operations the system must perform, while non-functional requirements address the overall quality, performance, and constraints under which the system operates. Together, these requirements establish a comprehensive framework for understanding the system's capabilities and limitations.

4.1. Functional Requirements

The functional requirements of the system are presented in a table format. Each requirement is assigned to one of the four categories **(C)** in the MAPE-K loop, that are Monitoring (M), Analyzing (A), Planning (P), Executing (E). The last column of the table indicates the priority **(P)** of each requirement, ranked on a scale from 1 (low priority) to 3 (high priority). This prioritization helps in identifying the most critical functionalities that need to be addressed during system implementation.

ID	Name	Description	C	P
FR1	Monitor Battery Levels	The system must continuously monitor the energy stored in the REC batteries and update users.	M	3
FR2	Collect Temporal Data	The system must gather and store temporal data regarding usage time τ and deadlines δ.	M	3
FR3	Control Device Activation	The system must activate consumers remotely based on energy availability, usage time, and deadlines.	E	3
FR4	Forecast Energy Trends	The system must use predictive models to forecast energy production and consumption trends.	A	2
FR5	Balance Energy Consumption	The system must ensure that stored energy is consumed efficiently while minimizing reliance on external energy sources.	P	3
FR6	Provide Production Insights	The system must monitor and display the energy production of each producer.	M	1

4.2. Non Functional Requirements

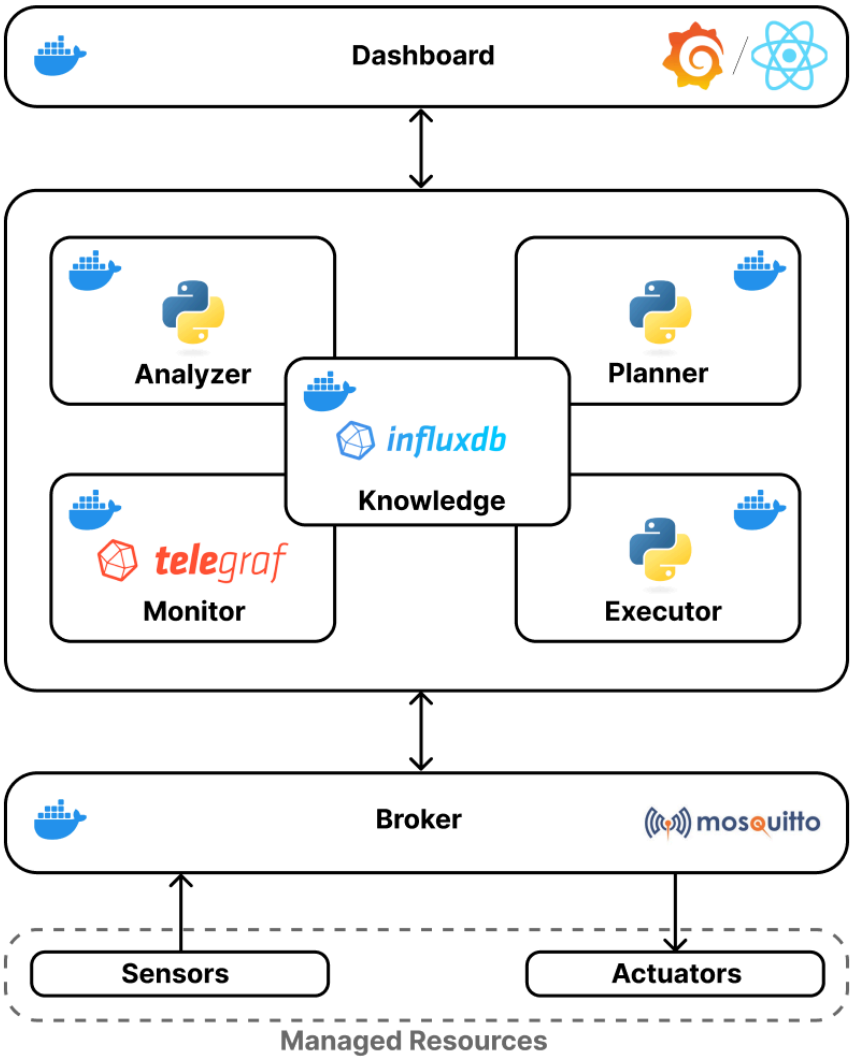
Similarly to what was done for the functional requirements, we present the most important non-functional requirements for our project in a table structured similarly to

the previous one. The main difference is the absence of the MAPE-K category, as these requirements pertain to the system as a whole.

ID	Name	Description	P
NFR1	Scalability	The system must support an increasing number of REC members and devices without issues.	**3**
NFR2	Performance	The system must ensure fast and efficient operation, minimizing delays in energy management actions.	**2**
NFR3	Usability	The user interface must be intuitive and accessible to users with varying levels of technical expertise.	**1**
NFR4	Reliability	The system must ensure continuous and fault-tolerant operation to prevent interruptions.	**3**

5. System Architecture

The following image represents the components of the system.



5.1. Components

Dashboard

In the dashboard, the user can:

1. View raw data related to energy production and consumption.
2. Interact with the system to specify the values of τ (requested usage time) and δ (deadline) for each of their consumer devices.

The first dashboard is developed using **Grafana**, which provides an intuitive and modern interface to display system data through interactive graphs.
The second dashboard is built using **React** for the frontend and a **Go** backend, enabling seamless interaction and efficient management of user inputs.

Managed Resources

The sensors and actuators are simulated using **Python** scripts. Specifically:

1. Sensors: They simulate realistic values for energy production as well as τ and δ for consumer devices.
2. Actuators: They handle the activation of devices requested by the users.

Broker

The broker used in the system is implemented with **Mosquitto**. The following topics are utilized:

1. **/producer/<member_id>/<producer_id>**: The sensor reading the production of a producer sends the amount of energy generated within a specific time frame.
2. **/battery**: The sensor monitoring the battery level sends the energy stored in the battery at regular intervals.
3. **/consumer/activation**: The system sends a message on this topic to activate the consumer with the specified member_id and consumer_id in the payload.

Additionally, for the purpose of simulation, a topic related to τ and δ is defined:

4. **/consumer/taudelta/<member_id>/<consumer_id>**: Simulates the assignment of τ and δ to the specified consumer.

Monitor

The monitor is implemented using **Telegraf** and is responsible for forwarding data from **Mosquitto** to the knowledge base, filtering only the topics that serve as inputs to the system.

Analyzer

The data stored in the knowledge base is analyzed to determine which consumers can be activated and under what conditions. This information is then passed to the planner.

Planner

The planner decides which of the consumers identified by the analyzer should actually be activated. The decision is then forwarded to the executor.

Executor

The executor sends instructions to the actuators, specifying which consumers, if any, should be activated.

Knowledge

The knowledge base is maintained in a time-series database, **InfluxDB**, where specific buckets are used to store:

1. Energy production data for each producer.
2. Battery charge levels.
3. δ and τ for the consumers.
4. Activation orders sent by the executor.

6. Implementation

6.1. Communications

The communications between the components are done in two ways (Excluding the communication with the database) that are:

- MQTT: between managed resources and the system.
- API: between the modules inside the system and between sensors and actuators just for a simulation purpose.

6.2. Simulation Timing

For the simulation, we decided to adapt a timing of 1 minute/sec. This means that each second of simulation corresponds to a minute in real life. This behaviour is managed in the Sensors module via the environment variables in the following screenshot:

```
- STEP_DURATION=1
- SECONDS_IN_A_SIMULATION_STEP=60
```

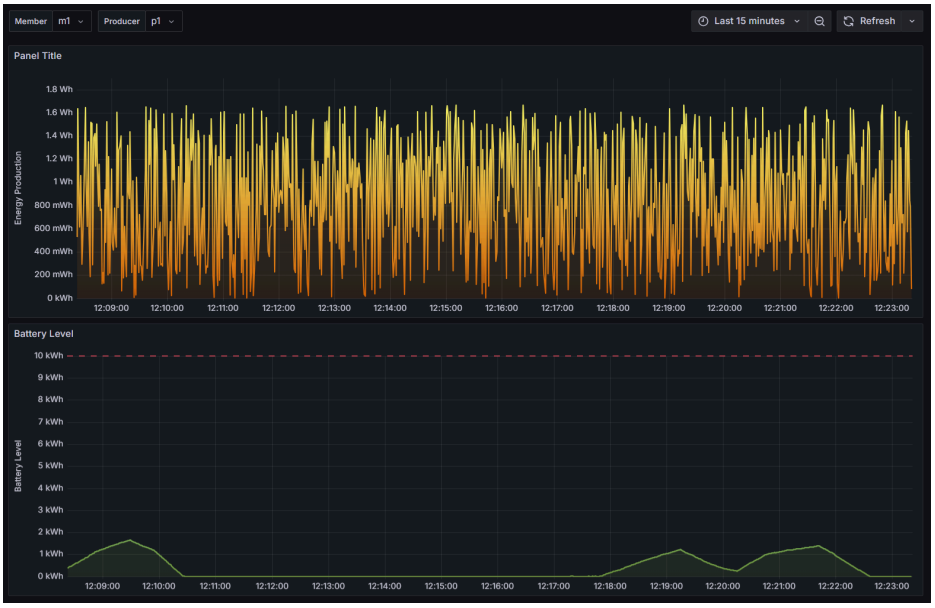
This means that a Simulation Step last one second and in a simulation step there are 60 seconds of real life.

6.2. Final Result

The following are some explained screenshots of the final result.

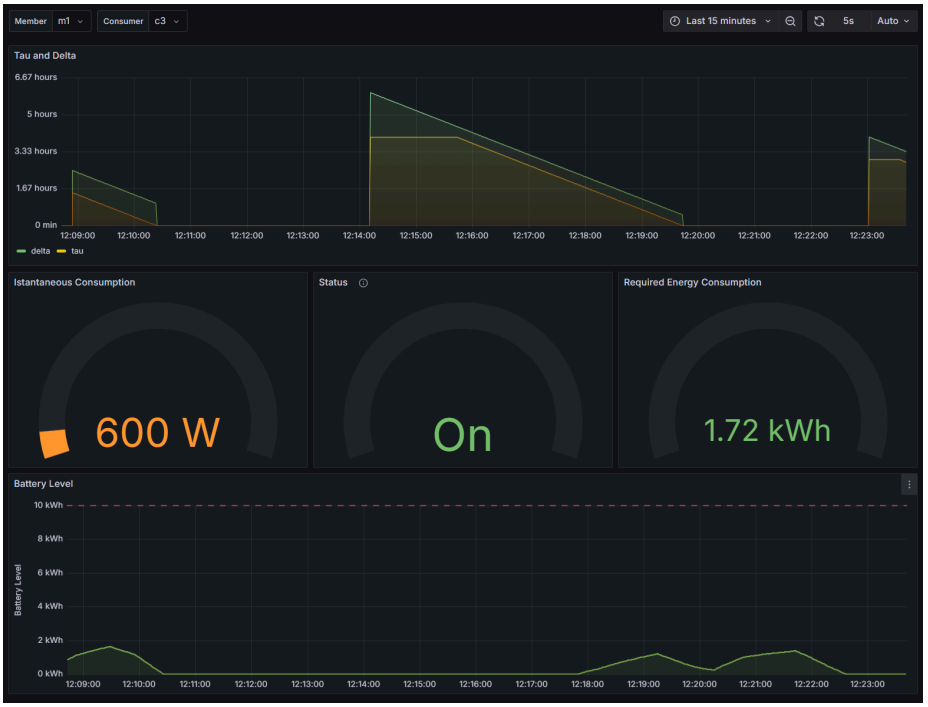
Grafana Dashboards

These Grafana dashboards explain how the system works in an instance of the simulation. First of all we have the **producers dashboard**: here we can select a producer and see how it is generating energy in the time. The second panel is just a window on the battery level of the entire REC.



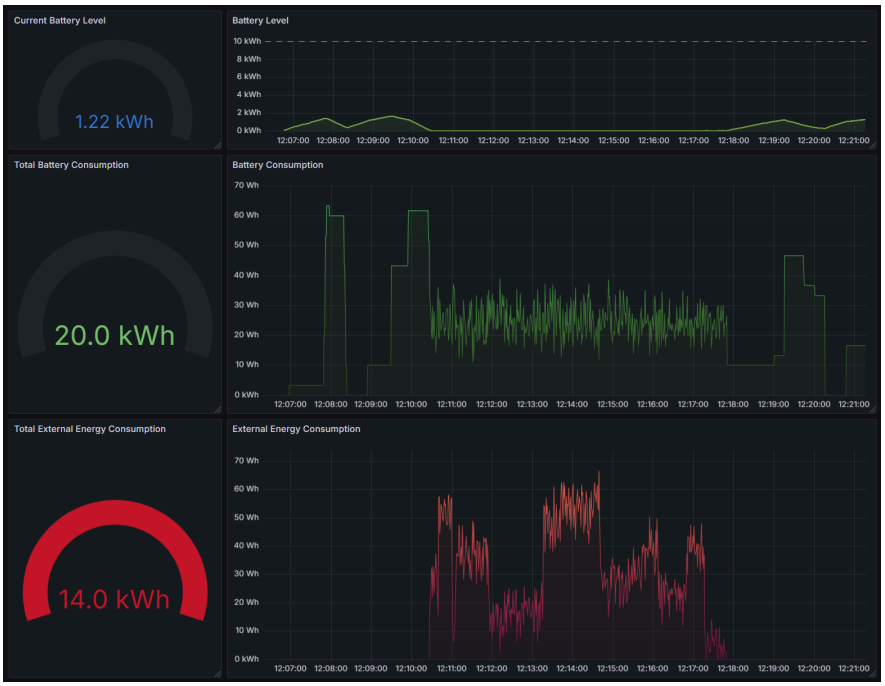
The second is the **consumers dashboard**, where there is a panoramic on the selected consumer. There are five panels:

1. **Tau and delta overview**: you can see how the tau and delta changes in the time.
2. **Instantaneous Consumption**: static panel that shows the instantaneous consumption of the consumer.
3. **Status**: whether the consumer is on or off at that moment.
4. **Required Energy Consumption**: the amount of energy currently required by the consumer.
5. **Battery Level**: another overview on the battery level.



The final dashboard provides a comprehensive overview of the entire Renewable Energy Community (REC). It features three distinct sections organized vertically:

1. First Row: Displays the **current battery charge level** and its operational status.
2. Second Row: Visualizes real-time energy consumption metrics, specifically highlighting **power supplied by the battery** storage system.
3. Third Row: Mirrors the layout of the second row but focuses on **energy sourced from external providers**.



User Dashboard

The really simple user dashboard allows users to specify tau and delta for a consumer.