



# Home & Tertiary real-time Energy Monitoring Module

D7.3

April 2022

# Deliverable

PROJECT ACRONYM	GRANT AGREEMENT #	PROJECT TITLE
TWINERGY	957736	Intelligent interconnection of prosumers in positive energy communities with twins of things for digital energy markets

## DELIVERABLE REFERENCE NUMBER AND TITLE

# D7.3 Home & Tertiary real-time Energy Monitoring Module

Revision: v1.0

### AUTHORS

Luigi Sechi	Federico Sanna	Alessandro Pietronave
STAM	STAM	STAM



Funded by the Horizon 2020 programme of the European Union  
Grant Agreement No 957736

## DISSEMINATION LEVEL

- ✓ P Public<sup>[1]</sup><sub>[SEP]</sub>
- C Confidential, only for members of the consortium and the Commission Services

# Version History

REVISION	DATE	AUTHOR	ORG...	DESCRIPTION
V0.1	12.09.2021	Luigi Sechi	STAM	Toc
V0.2	15.11.2021	Luigi Sechi	STAM	1 <sup>st</sup> draft
V0.3	25.03.2022	Luigi Sechi	STAM	2 <sup>nd</sup> draft
V0.4	22.04.2022	Luigi Sechi	STAM	Updated draft sent for review
V0.5	29.04.2022	Luigi Sechi	STAM	Final draft submitted to PC
V1.0	30.04.2022	Luigi Sechi	STAM	Final draft submitted to EC by the PC

## Statement of Originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation, or both.

---

## Legal disclaimer

The information in this document is provided “as is”, and no guarantee or warranty is given that the information is fit for any particular purpose. The above referenced authors shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials subject to any liability which is mandatory due to applicable law. The sole responsibility for the content of this publication lies with the authors. It does not necessarily reflect the opinion of the European Union. Neither the CINEA nor the European Commission is responsible for any use that may be made of the information contained therein.

© 2022 by TwinERGY Consortium

# Nomenclature

Abbreviation	Explanation
API	Application Programming Interfaces
AVG	Average
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DoA	Description of Action
DR	Demand Response
DSO	Distribution System Operator
DT	Digital Twin
GUI	Graphical Users Interface
PV	Photovoltaic
RES	Renewable Energy Source
TSO	Transmission System Operator

---

# Summary

The present document is the D7.4 “Home & Tertiary real-time Energy Monitoring” of the TwinERGY project, funded by the European Commission’s Innovation and Networks Executive Agency (CINEA) under its Horizon 2020 Research and Innovation programme (H2020). The main objective of the D7.4 is the delivery of the Module, the respective Web and Mobile application. Moreover, the handbook for the user interfaces and for the other services is provide.

Tenants, building managers and up to the energy providers and the DSOs, are interested in increasing the observability of the energy loads and source at various level of the power grid. The necessity of a monitoring platform for energy consumption arises firstly from the need to minimise energy waste and then to optimise the local electricity grid, which otherwise suffers from high fluctuations in demand but also changes in the direction of electricity flows. It happens for example when PV panels are installed on a building and in general in every distributed generation scenario.

From the tenants’ point of view, the needs are to increase their awareness, to understand the bad behaviours energy related, and to allow remote control with the final focus on the energy savings and environment impact reduction.

Stakeholders agrees on the local users’ empowerment, and the future energy communities implementation is foreseen as the true horizon.

# Index

<b>1 Introduction</b> .....	9
1.1 Overview.....	9
1.2 Structure of the deliverable .....	9
<b>2 State of the Art</b> .....	10
2.1 Energy monitoring platform .....	10
2.2 Domestic and tertiary utilities .....	11
2.1.1 Demand response.....	13
2.3 The IoT era - market analysis .....	14
2.4 Friendliness in the GUI .....	20
<b>3 The Energy Management dashboard</b> .....	22
3.1 Graphical Users Interface - GUI .....	22
<b>4 Management Service implemented</b> .....	24
4.1 Fault Anomaly Detection .....	24
4.4 The DBSCAN algorithm .....	27
4.4.1 How does the DBSCAN works? .....	29
4.5 The HDBSCAN algorithm.....	31
4.6 Services architecture .....	38
4.6.1 Model generation.....	39
4.6.2 Data retrieving.....	39
4.6.3 Data preprocessing.....	40
4.6.4 Model training .....	40
4.6.5 Model saving .....	40
4.6.6 The anomaly detection service .....	41
4.7 Fault Anomaly Detection GUI.....	42
<b>5 TwinERGY Platform Integration</b> .....	44
5.1 Optimal RES selector.....	44
<b>6 Conclusion</b> .....	45

---

Annexes .....	46
Annex A – Input and output of the Anomaly Detection algorithm components.....	46



# List of figures

Figure 1 - Example of a smart grid schema .....	10
Figure 2 Electric energy demand curve of a domestic utility .....	12
Figure 3 Tertiary utility typical energy demand trend .....	13
Figure 4 - Device's energy consumption .....	15
Figure 5 - SEM3 System .....	16
Figure 6 - PowerLogic system .....	17
Figure 7 - Wi-Lem system.....	18
Figure 8 - Techus smart meters.....	19
Figure 9 GUI from [5] and GUI from [7] on the right .....	20
Figure 10 - On the left Wiser app by Schneider, in the center Ned app by Midori and on the right Techus app by Enermed.....	21
Figure 11 - SEM3 Branch meter configuration page.....	21
Figure 12 Twinergy energy dashboard .....	23
Figure 13 Electrical appliances cards .....	23
Figure 14 Concentric circles dataset – example n.1.....	27
Figure 15 Comparison between K-means and hierarchical clustering on example 1 .....	28
Figure 16 DBSCAN clustering on example 1 dataset.....	29
Figure 17 Dataset example n.2.....	30
Figure 18 DBSCAN clustering on example 1 dataset.....	31
Figure 19 First point selection - Core distance definition with $k=5$ .....	32
Figure 20 Second point selection .....	33
Figure 21 Third point selection.....	33
Figure 22 Reachability distance between the blue and the green points.....	34
Figure 23 Reachability distance between the green and the red points .....	35
Figure 24 Minimal tree graph using the Prim's algorithm.....	36
Figure 25 Dendrogram of the edges of the tree sorted in increasing distance .....	37
Figure 26 Reduced tree using a minimum cluster size of value 5 .....	38
Figure 27 Fault anomaly detection service architecture .....	39
Figure 28 Anomaly detection service implementation.....	41
Figure 29 Anomaly Detection service dashboard .....	43

# 1 Introduction

## 1.1 Overview

Following the DoA, this deliverable reports on the development of the tool for the risk management within Twinergy platform. In this deliverable, the release of the tools and the algorithms used for it are described in detail.

The present document describes the Graphic User Interface of the Energy monitoring system of the Twinergy Platform, the outputs, and the possible remote-control actions of the appliances. It is also presented the service of Anomaly detection implemented; This type of service well pair with software application focusing on the energy monitoring and its logic and the outputs explanation is provided. The module authentication and data in the dashboard refer to the single user.

## 1.2 Structure of the deliverable

This document is composed of six chapters as follows:

- Chapter 2: State of art of the Energy Management Platforms
- Chapter 3: The Energy Management Module dashboard in Twinergy
- Chapter 4: The Anomaly detection service
- Chapter 5: The T7.4 module integration phase in the Twinergy Platform
- Chapter 6: Conclusion
- Annexes

## 2 State of the Art

### 2.1 Energy monitoring platform

The past developments of the energy platforms were purely on industrial level, especially in the countries that historically import electricity (such as Italy<sup>1</sup>), due to their necessity for money-saving. The energy generation needs and the energy costs lead to a great boost in the development of energy monitoring and management platforms. The phenomenal has been enhanced by the development of electronic and software technologies, supervisory control system, and data acquisition (SCADA) which have been widely used in plant automation. They provided efficient tools for monitoring and controlling equipment in industrial production processes [1].

Additional features allow the TSOs and the DSOs to view grid data such as voltage fluctuation, energy consumption, power factor, and frequency, also from an operational point of view for real industrial applications.

But the complexity of managing the distribution grid, considering up to the residential buildings (Figure 1) has increased dramatically with the introduction of local renewable energy production systems.

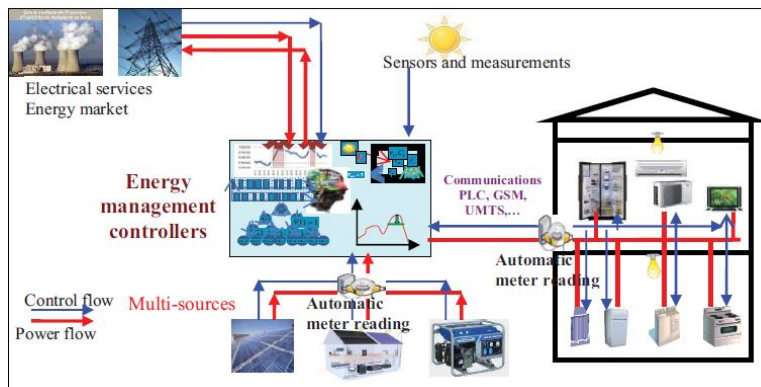


Figure 1 - Example of a smart grid schema

The change to a distributed generation model made up of small self-generation plants, overcoming the traditional power plants logic, was triggered by improvement in environmental impact but also as economic chances for individuals and businesses, which would become an active part of the process.

Hence the need for investments to increase the efficiency of the grid to manage fluctuations in values and directions of electricity during the day [2].

<sup>1</sup> <https://www.mercatoelettrico.org/en/>

Therefore, thanks to the technological improvements, there was the possibility to move from the industrial to the residential and tertiary sectors, whose consumption accounts for about 40% of electricity consumption in Europe, indeed it is a large market on which to act to minimize consumption and energy waste. [3]

Among the first applications, there were public buildings such as in Ferreira et al. 2018, in which the electricity consumption of each room was assessed using wireless sensors for the evaluation of electrical consumption and a raw platform for data analysis.

In this case, however, ad hoc wireless sensors were developed and included because of the lack of smart devices available in the market. Until a few years ago, users were not able to easily find and install the smart devices and to reach a good monitoring detail concerning electrical flows for residential cases.

Thanks again to the technological progress, nowadays these boundaries have been overcome, and a user well-engaged, can find monitoring solutions to provide the energy data up to the single household appliance.

## 2.2 Domestic and tertiary utilities

As said above, the electricity demand of a building is generally non-uniform, it varies continuously during the day and it is characterised by peaks during certain hours of the morning or evening. The type of user is a fundamental aspect that significantly characterises the withdrawal curve of a facility. In particular, the project focused on the distinction into 2 main utilities typologies:

- Domestic buildings
- Tertiary buildings (offices and shops)

Households have a characteristic energy withdrawal curve depending on the occupancy of the building. For this reason, it is important to include an estimation based on the inhabitants and on the weekday. A typical weekday withdrawal curve for a domestic user is shown in Figure 2.

There is a reduction in consumption during the morning, when the user is usually away from home, and an increase and peak in consumption during the evening hours [5].

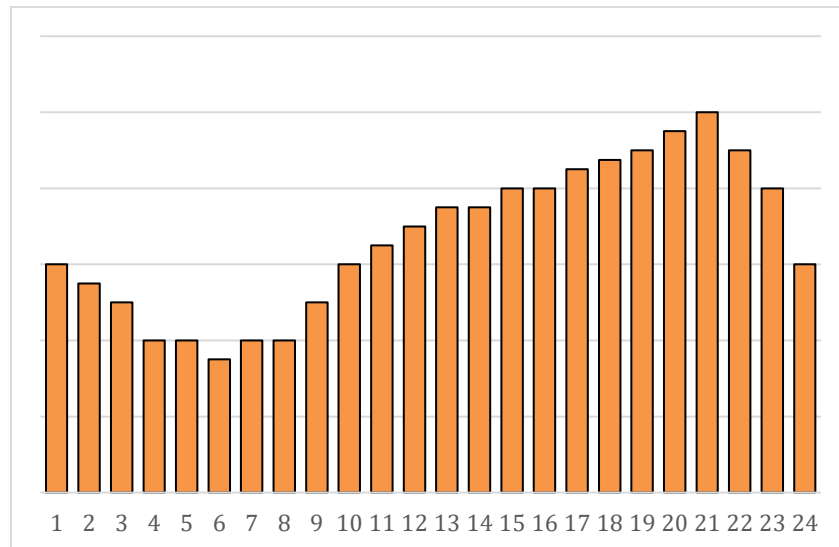


Figure 2 Electric energy demand curve of a domestic utility

This typical household curve reflects the demand trend, especially when considering aggregated domestic users.

Continuous technological progress has led to the development of increasingly efficient appliances that have balanced factors such as:

- Increase in the number of appliances at home (such as washing machine, dryer, air conditioning, television, computer etc.). The rate of ownership has increased significantly compared to the past and it is now common to have several televisions, computers and several fridges and freezers
- Increase in electronic equipment

Other influential factors, mostly behavioural factors, have not been counterbalanced and over the years there was a progressive change in the withdrawal curves due to factors such as:

- Increased use of traditional devices, e.g. the number of hours people watch television or use computers;
- Increase in the density of the elderly population, with a more sedentary lifestyle and therefore more time spent at home, requiring for example certain temperatures in the rooms where they are located (thus increasing heating in winter and air conditioning in summer).
- And recently, thanks to the pandemic, also a huge improvement of the smart working from home

When studying the individual case, as in the present project, a typical profile function of the characteristic curve alone is often insufficient, and it is necessary to go into greater detail in the following paragraphs.

On the other hand, the tertiary utilities can be of various types, such as shops or offices. These kinds of utilities present a characteristic energy daily trend, more predictable, and an energy withdrawal following the canonical working hours, from 8:00 to 17:00, with less daily variability. For them, the canonical consumption characteristics remain fairly faithful to the real behaviour; a typical withdrawal curve of a tertiary user in a weekday is shown in Figure 3. [5].

There is an increase in consumption from 8 a.m. onwards and an almost constant consumption until 5 p.m., whereas outside working hours only the base load of electrical appliances and stand-by appliances is recorded.

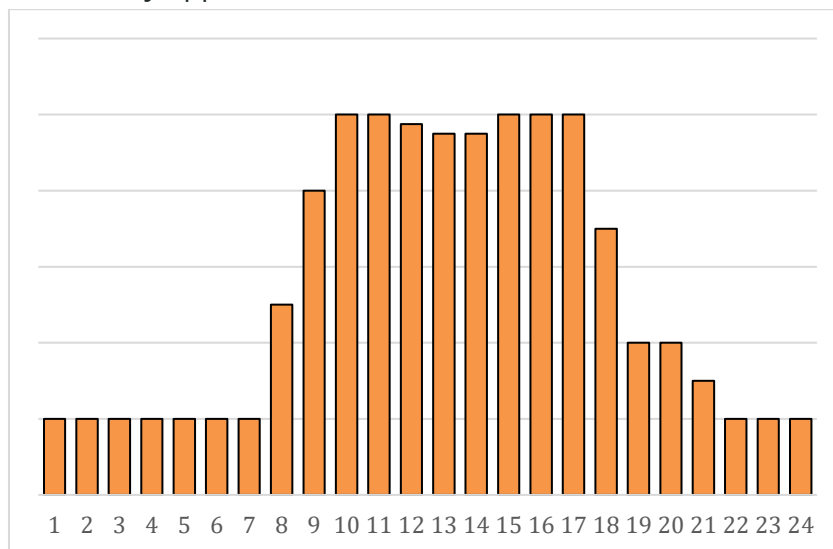


Figure 3 Tertiary utility typical energy demand trend

## 2.1.1 Demand response

As said above, the energy demand is not constant but varies as a function of time (for example, during 24 hours), but also of other factors such as geographical position, climate, season, and many other parameters.

To avoid such problems, various Demand Response (DR) strategies have been studied which aim to distribute the energy demand throughout the day by reducing and possibly eliminating energy peaks. There are different DR policies that, to achieve the same goal, follow different paths:

1. DR based on price: in this type of program, hourly rates are stipulated in the contract with the user, in this way an attempt is made to motivate the user to use

- energy in different time bands than peak times (i.e. with cheaper hourly rates), penalising instead with higher costs the use of energy during peak hours.
2. Incentive or event-based DR: There are “rewards” in this program for users to reduce their demand voluntarily. The DR program manager sends the communication to reduce the load following a particular event (for example, unexpected peak demand) to the users who have signed this type of contract. To provide fair treatment and not to penalize the same users, time thresholds are established for each user (daily or monthly) which, once reached, does not allow them to be further involved.
  3. DR based on free offers for demand reduction: in this program, customers send their offers indicating how much they are willing to reduce energy demand for a final discount

Furthermore, the management of the grid has several problems that can arise from non-linearities such as the actions of the power converters or the imposition of the limitation of the various components of the device, or the difficulty in choosing one that will supply energy to a set of charges, among a set of energy - sources of generation.

## 2.3 The IoT era - market analysis

Nowadays, thanks to the Internet of Things (IoT), a very large number of devices can be remotely controlled and connected to traditional BACS systems, and thanks to their lower price they are easy to purchase. The Internet of Things describes the network of physical objects — “things”— that are embedded with sensors, software, and other technologies to connect and exchange data with other devices and systems over the internet.

The energy meters and sensors are placed in correspondence of the power sockets and electric panels to record the data relating to the energy consumption. The analysis of the collected data allows to formulate energy-saving strategies. This last phase is dealt with the so-called Energy Management System (EMS), which has the function of analysing the collected data by multiple measuring devices.

For instance, there are different attempts in the literature to create complete and simple platforms called HEMS (Home Energy Management Systems); the one made at the University of Pisa and called “GreenBuilding” is an example [6].

In this paper, a room on the university campus with two users is monitored, in which the consumption of the various users (PC, lights, fridge, etc.) is recorded (Figure 4).

Once the recorded consumption data has been analysed, it is possible to set up strategies for reducing energy consumption and, thanks to additional sensors (e.g. Light), introduce the logic of variation in consumption due to external agents.

The goal of “GreenBuilding” is twofold. On one hand, it allows the user to better understand the energy consumption of every single appliance, thus bringing him/her to make more intelligent choices in terms of electricity consumption and then reduce them. On the other hand, it allows an improved energy efficiency using an appropriate management of every single appliance, depending on rules specified by the user. So, in this case, also a control on the actuator of the various appliance is applied, while in the residential case with a market solution is not always possible.

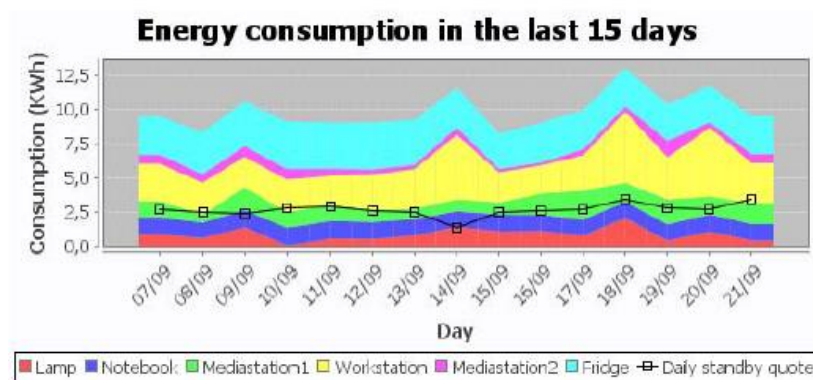


Figure 4 - Device's energy consumption

Since the old houses do not have systems with actuators already included in the operating logic of the electrical system, it is less expensive and then the solution is more attractive to the customer when a non-invasive monitoring system is introduced.

In this case, it is possible to see the consumption of each appliance but without implementing actuation logic [7].

Finally, the energy market includes several companies, such as Wi-Lem, Enermed, Change Electric, Mitsubishi Electric, Siemens Energy, and Schneider electric, or even start-ups such as Midori that provide innovative monitoring systems with their platforms.

Among the most developed solutions, there is Desigo by Siemens which allows users to efficiently manage small and medium-sized buildings where the attention is focused on the visualization and control of lighting systems and low voltage equipment such as domestic appliances. Or also still from Siemens the SEM3 system (Siemens Embedded Micro Metering Module) which is a family of devices designed to measure the current, voltage, energy usage, and many additional parameters for anywhere from 1 to 45 branch circuits. The Siemens Embedded Micro Metering Module (SEM3) product is a solution based on metering modules for energy consumption monitoring, data analysis, and billing. The data collected by the users can be consulted in real-time through a web page. SME3 allows to measure the energy loads and identify the necessary ones to eliminate the superfluous ones. The system can be integrated into existing electrical systems, through a retrofit kit and thanks to the size of this system, into already existing Siemens systems. SEM3 consists of five basic elements (Figure 5):



1. **Controller** - The controller is used to communicate to the user the values measured directly by the measuring devices and modules. A controller can manage about 45 modules and it can communicate with the customer through a web page.
2. **Meter Modules** - Measurement modules have two possible levels of accuracy (1% if standard accuracy and 0.2% if high accuracy). It can read the intensity of the current and voltage signals through its transformers, independently from the other modules. After the measurement, there is a phase of processing the data (through an algorithm) which is then sent to the controller. It is possible to set the phase (A, B, C) through a switch on the module
3. **Meter Rack** - These are the modules containers with variable capacity (3, 9, 15, 21 slots), they allow the internal communication of information between the measurement modules and the controller through Ethernet cables.
4. **Current Transformers (TCs)** - The SEM3 System is integrated with physical TCs used for measuring alternating current (AC). TCs are passive components and the measured current flows in a conductor of the power system. TCs are designed to work specifically with SEM3 and provide 100mA output, conductor wires are 18 AWG (American Wire Gauge), have primary classification 600 VAC (electronic alternating voltage), and overvoltage category CAT IV.
5. **Communication cables** - Communication cables are designed to be isolated in systems of more than 600V, they are two-way communication from the controller to the rack.

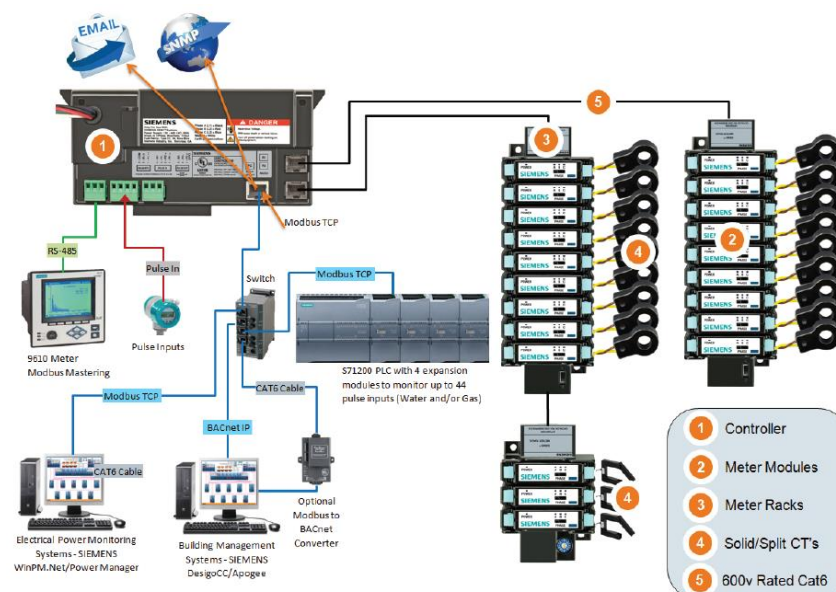


Figure 5 - SEM3 System

Schneider Electric has a system called “Wiser Energy” for real-time energy monitoring at home. It reads the currents from various appliances throughout your house and processes that data to visualize home energy usage.

Once the Wiser Energy system unit is installed in a home's electrical panel, and the linked app is set, it is possible to have real-time access to electrical use, 24/7. Over time, through crowdsourcing and using machine learning, the app senses electrical activity and identifies different appliances. Once appliances are identified, data is elaborated and shown on the app.

So with one single app, it is possible to control the entire home: lighting, shutters, temperature, safety, and energy. Then thanks to an intuitive user interface, it is easy to personalize. The system is modular and scalable.

Another system designed by Schneider Electric is “PowerLogic” (FIGURE 6), which combines precise energy measurements and power quality analysis. This system is suitable for both monitoring of high voltage (for example industrial plants, shopping centres, public networks) and low voltage (for example, for domestic use) systems. According to Schneider Electric, the user who chooses to integrate this system into their electric plant will have a reduction in costs related to the consumption of electricity and optimization of consumption, which in turn leads to an optimization of the life of the electronic devices.

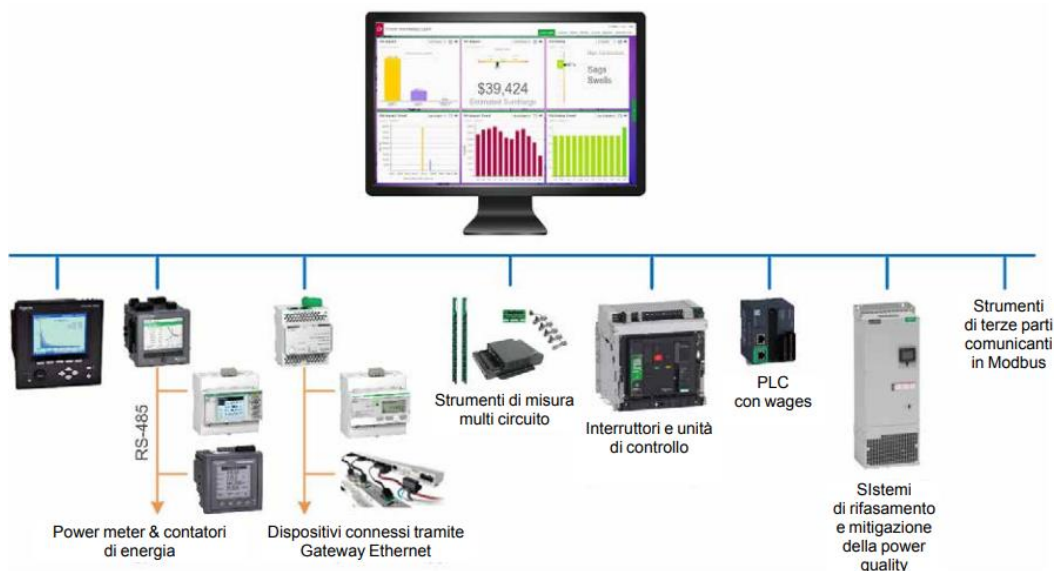


Figure 6 - PowerLogic system

Mitsubishi Electric Corporation announced in 2019 that it has developed a new technology that allows the estimated power consumption of individual home appliances to be extrapolated from the overall power consumption of each household.

LEM proposes the system Wi-LEM (FIGURE 7), which is a complete data acquisition platform able to measure and transmit electrical parameters used for Energy Management

applications. It is an open architecture, so this platform can be easily interfaced with existing data logger and energy monitoring software.

The Wi-LEM meters simply clip around the electrical circuit to be monitored and wirelessly transmit the meter data using a robust mesh wireless protocol that significantly reduces the time, cost, and disruption involved in deploying sub-metering installations. A wireless pulse input unit (Wi-Pulse) is also available for acquiring data from any meter with a standard pulse output facility.

Energy consumption data is automatically saved on a web server that provides powerful interactive energy charting using the simplicity of a Web Browser. Logged data can also be automatically emailed or transmitted via the IP network to a Monitoring & Targeting (aM&T) package or a spreadsheet such as Excel for further analysis.

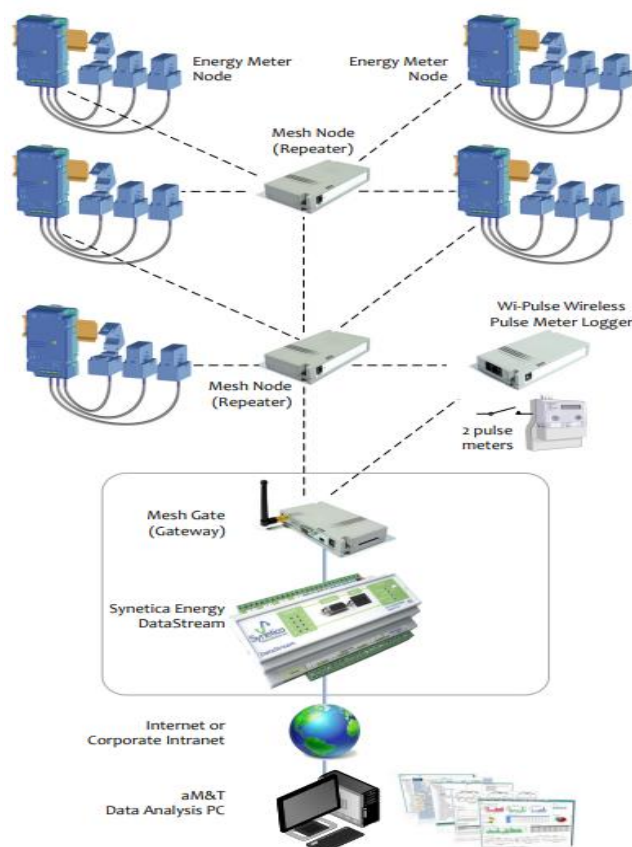


Figure 7 - Wi-Lem system

Techus is a technology developed by Enermed, a company that also proposes itself on the market as an energy provider. Techus is a system that provides a smart meter (with the function of collecting data), a cloud (in which these are stored) which are subsequently sent to a cloud and then returned to users who can consult them on their personal devices (such as smartphones, pc, and tablet). The Techus smart meter (*FIGURE 8*) allows the customer to learn more about his energy behaviour and profile, reporting information regarding:

- Daily energy withdrawals.
- Quantity of kWh consumed.
- Time bands of greater or lesser consumption.
- Power consumed by the user.

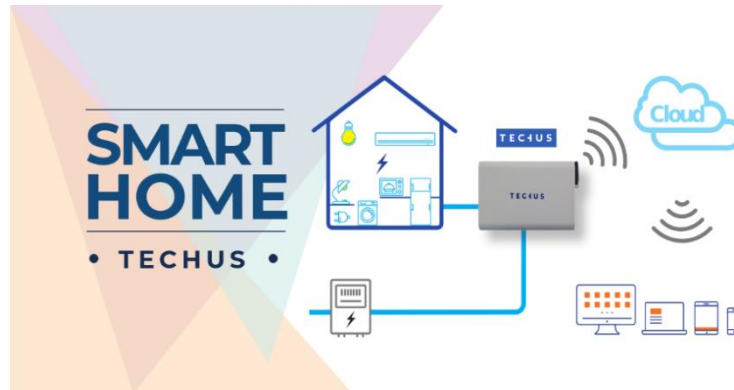


Figure 8 - Techus smart meters

Finally, an example of a start-up product is the one called “Ned” from Midori, whose technology is based on the "electric footprint" that every appliance leaves while it is in operation. Ned searches for these footprints over time and frequency, starting from the analysis of the aggregate consumption of the whole house, then it can distinguish the category of the appliance, and by answering some questions on the app the devices are completely recognized.

### Others

“Accuenergy” has developed several products for monitoring energy consumption, each of which has its own technical characteristics. Among their offerings, the Aculink 710 system is analysed, which connects the meters with third-party devices based on IP systems. This allows to access the parameters that evaluate the performance of the system in real-time directly from specific websites. Furthermore, the data is not stored locally on the device but in the cloud. This system is also able to send notifications to users if the energy levels do not respect the planned ones. The sensors send the collected data using a wireless connection so as not to insert additional cables into the system. Another sub-metering system is provided by CTS Energy Services Metering which also provides monitoring systems for water and gas consumption as well as electricity consumption. The use of this system would make it possible to achieve economic savings of between 20 and 25%, as well as a reduction in carbon emissions.

## 2.4 Friendliness in the GUI

An important prerogative for monitoring systems and their platforms is the user-friendliness. Since these are systems for residential use, the end customer is the common person, therefore it is immediate to think that having a smartphone app is the easiest way to reach as many customers as possible.

Excellent graphics are the instruments used for representing quantitative information. Graphics help people to understand complex things easily. Hence the user interfaces developed should be clear, illustrative, and designed from the user's point of view concerning their applications.

Actually, for the global management of a building, a desktop app with proper software is always used by an energy management company but for the consumption of the final user, a smartphone app is preferred.

The difference between platforms born from university research projects and the market is evident, in fact, the university platforms, not available on the market, have less user-friendly graphics and content since those programs have less customer orientation and are more dated (*FIGURE 9*).

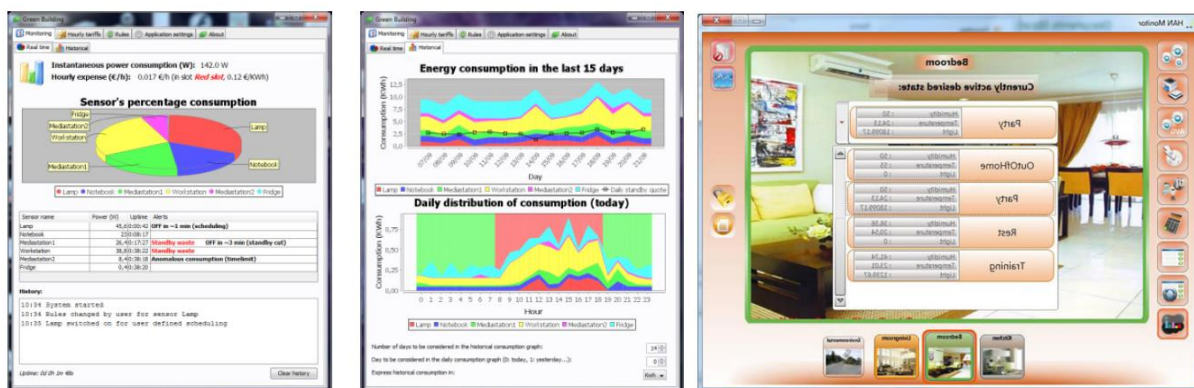


Figure 9 GUI from [5] and GUI from [7] on the right

Another example of the research project is the one made by [8] who develops a monitoring system called SESAME. In which it is tried to introduce different interface approaches with an evaluation of their impact on people.

A lot of modern and continuously improved monitoring systems have a very attractive graphic (*FIGURE 8*), which combines user-friendliness with important capabilities.

However, systems such as SEM3 born for contractors, commercial and complete buildings have a desktop application less intuitive due to a lot of information collected by the system (Figure 11).



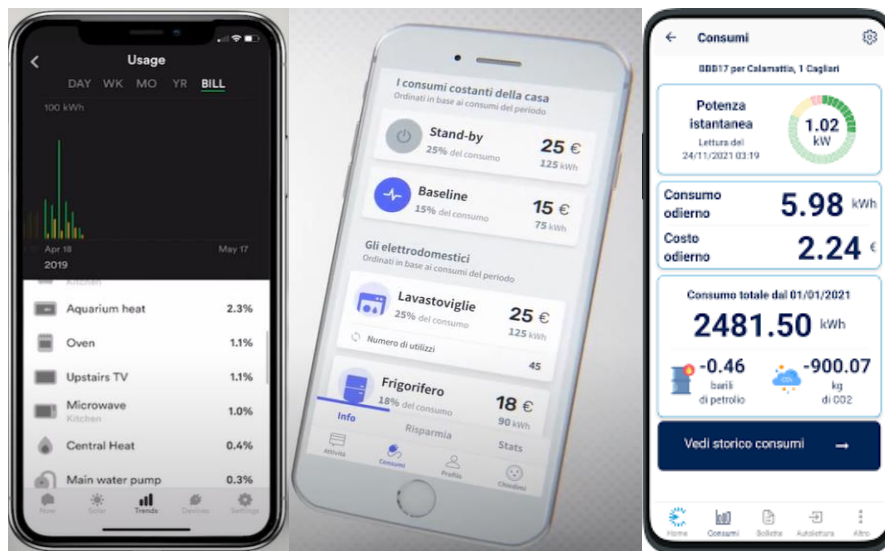


Figure 10 - On the left Wiser app by Schneider, in the center Ned app by Midori and on the right Techus app by Enermed



Figure 11 - SEM3 Branch meter configuration page

From these experiences, it is understandable that the fundamental role of a proper GUI is to be appealing to spread those monitoring systems to all the population, to reach the ultimate goal of electricity consumption reduction.

# 3 The Energy Management dashboard

All the module implementation provided in the T7.4 task, are deployed on cloud, using the Google Cloud platform services which provide infrastructure as a service, platform as a service, and serverless computing environments.

All the work has been developed using the Java coding language.

## 3.1 Graphical Users Interface - GUI

One of the main goals of the GUI developed within the T7.4 task was to reach a representation of the building energy pattern easy to be read by the users.

The focus is currently oriented on the electricity vector and on the electrical loads energy consumption, in order to increase the users' awareness on his energy consumption behaviour.

In *FIGURE 12*, the current dashboard state is shown, deployed online at <https://twinergy.stamtech.dev/dashboard>.

To increase the accessibility of the dashboard, only 2 graphs have been plotted and only the 5 most energy demanding loads are shown.

The charts plotted refer to:

- The energy consumption of the 5 main power lines on the electrical panel
- The voltage of the 5 main power lines on the electrical panel

The other electrical loads are shown aggregated in a unique load called "Others"; for them, the voltage plotted is the average voltage.



Figure 12 Twenergy energy dashboard

The dashboard currently allows to plot the graphs of the last 7 days, with a daily resolution. Further development will follow the integration with the Twenergy pilots, in order to reach the selection of the monthly period and allowing to the users the possibility to verify the correct billing of their energy bills.

Moreover, it is planned to provide also a daily detail with an hourly resolution whenever the monitoring system will be able to provide it.

A second goal of the T7.4 Energy Management Module is to allow the users the possibility to monitor and remotely control their electrical appliances power supplied via a smart plug.

To allow that, a card logic is implemented in the dashboard (Figure 13).



Figure 13 Electrical appliances cards

On the cards it is shown the specific daily energy consumption of the appliances, their status, and a button will allow the switching ON/OFF of the appliances.

The meaning is to introduce the users to a preliminary and autonomous usage of Demand Response actions, based on their own willingness.

The next step is to provide to them best usage notification and/or, act an automatic control whenever the user allows to it.



# 4 Management Service implemented

In the following chapter, the Anomaly Detection services implemented in the module are shown. The detection is focused on the anomalies of the energy monitored dataset especially, the energy generation for the PV system and the energy consumption for the HVAC. For the HVAC, the energy trend will be considered in pair with the system performances in terms of temperature provided.

From now on, for anomalies, will be considered the variation in the energy trend that triggers the detection of a Machine learning algorithm called HDBSCAN; the algorithm provides the anomaly with a score index from 1 to 10 which is related to the magnitude (severity) of the anomaly.

The anomaly detection service is related to the energy monitoring service core of the T7.4 module; the integration between the two services is provided at in end of the document in the Annex A.

## 4.1 Fault Anomaly Detection

The anomaly detection is an important topic that finds application in different domains. Anomaly detection allows to detect anomalous trends of the system under analysis and to identify some of its key parameters that could have caused the anomalies by diverging significantly from a common and wanted trend.

There exist different approaches that can be summarized into three broad categories:

- **Threshold-based approaches** that are the result of statistical analysis on the available data
- **Supervised learning approaches** that involve a Machine Learning (ML) algorithm applied to labelled data
- **Unsupervised learning approaches** that do not make assumptions on the input dataset, but are able to find anomalies by discovering hidden patterns in the data

Threshold-based algorithms are implicitly supported by a probabilistic model, e.g. multivariate Gaussian model. By means of a statistical analysis of the dataset, each input data is checked against the model and, in case they exceed its boundaries, they are labelled as anomalies.

These types of algorithms are usually applied to small datasets that do not show highly complex patterns in them. Because of this, they didn't suit the T7.4 needs as the energy data under analysis is far more complicated than the standard situation where this category usually falls as the most appropriate one to choose.

Therefore, the focus relies on learning and applying the two other types of algorithms, which are supervised and unsupervised algorithms.

## 4.2 Supervised VS Unsupervised Learning

The main difference between the two approaches is that supervised learning has its grounds on a prior knowledge of what the output values of the algorithm should be, whether the unsupervised learning does not have this information given beforehand.

Therefore, the goal of supervised learning algorithms is to learn a function that, given a dataset and possible output values, tries to find possible relationships between the two of them. On the other hand, unsupervised learning algorithms do not have output values as input parameters of the function, thus their objective is to infer eventual patterns in the dataset.

There are two main types of problems related to supervised learning:

- **Classification problems**, where an input value has to be mapped to a single discrete value, typically extracted from an already defined enumeration of values (like evaluating if an animal is a dog or a cat inside an image);
- **Regression problems**, where the predicted values are real numbers, therefore the output set contains continuous data. A typical problem of this type would be, for example, predicting the price of houses.

Common supervised learning algorithms are neural networks, K-nearest neighbour, and support vector machines.

Given said facts, it is possible to use supervised learning to detect anomalies, but prior anomaly situations have to be studied beforehand so that they can be passed as input arguments to the learning function. The main problem with this constraint is the impossibility to have always the records of previous anomalies; instead, it is necessary to study the whole dataset and discover anomalous situations without any labels.

That's the main purpose of unsupervised algorithms: their functions try to learn the structure of the data without explicitly provided labels. Common tasks of these type of algorithms are clustering and density estimation.

## 4.3 Clustering as Unsupervised Algorithms

Clustering can be considered the most important unsupervised learning problem, and it deals, like the others, with finding some structure in a set of unlabelled data. A *cluster* is a collection of objects similar among them in terms of properties, and at the same time dissimilar to objects belonging to the other clusters [9]. So, the algorithm must define a function which, given the dataset, outputs a list of clusters containing all the entries in the dataset.

There exist several approaches to clustering, like:

- **Centroid-based clustering**, where the data is organized into non-hierarchical clusters. They are thought of as one of the simplest yet effective clustering algorithms; they focus on calculating a central vector for each cluster, which is then updated every time an object is added to said cluster. They also make use of some metric to measure the distance between clusters;
- **Density-based clustering**, where the centroid-based distance metric is discarded in favour of one based on density. In this case, clusters are considered the densest regions in the space of data, which are then separated by region of lower density that are not considered clusters. The advantage of this type of clustering is that it can discover clusters of arbitrary shape and which can also contain outliers of noisy data;
- **Distribution-based clustering**, where the clusters are created through probability distributions like Gaussian, Binomial, etc. Data points are grouped based on their likelihood of belonging to the same distribution. This type of clustering has advantages in terms of flexibility and correctness, but the drawback is that the data has to be generated with the assurance that most of it belongs to a predefined distribution, or else the algorithm will overfit.
- **Connectivity-based clustering**, also known as *Hierarchical clustering*, starts with a predefined top-to-bottom hierarchy of clusters and proceeds to decompose the objects focusing on this given hierarchy, hence obtaining the final clusters. The decomposition task can be carried on:

- *top-down* through a divisive approach, it starts putting all the points into one big cluster and then divides the data through some mathematical function until it reaches a termination point;
- *bottom-up* through an agglomerative approach, initially each point is a single cluster, and they get grouped together using some kind of logic (it can be a simple number based criterion, a distance criterion or a variance criterion).

## 4.4 The DBSCAN algorithm

When using clustering algorithms, the typical choices are usually the K-Means algorithm or hierarchical algorithms. But, as we said before, both of them fail in creating clusters of arbitrary shape: they cannot form groups of points with varying density. To solve this problem, we can rely on density-based approaches like the DBSCAN algorithm [10], [11].

A simple example follows to explain this issue. Figure 14 represents a dataset in form of concentric circles:

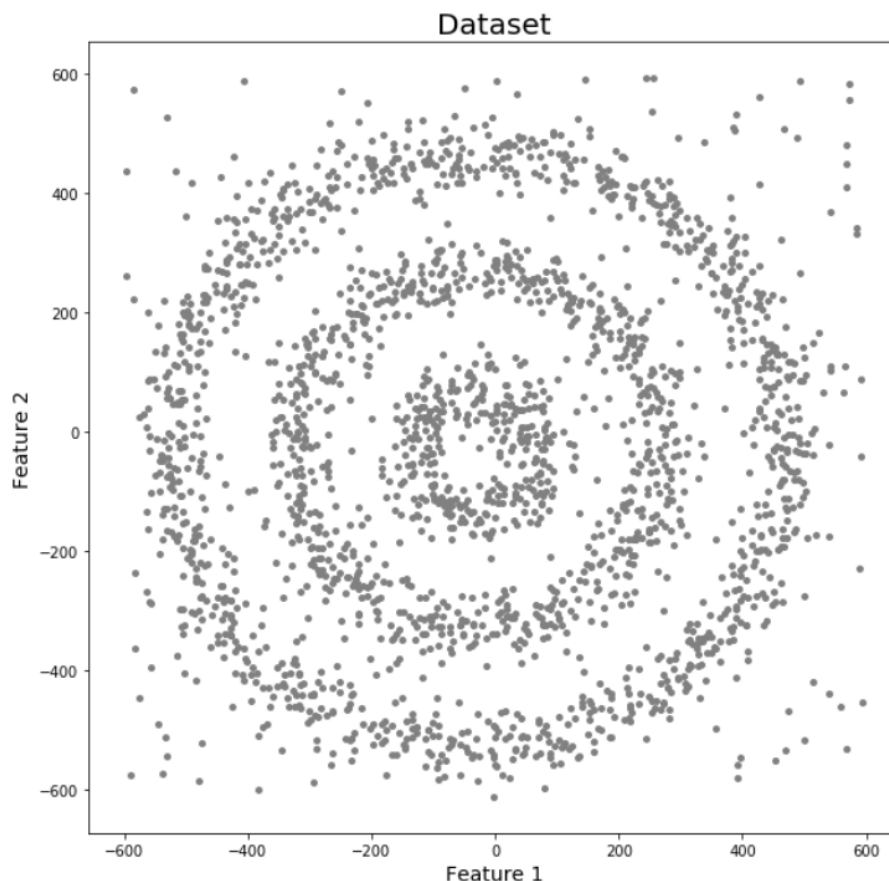
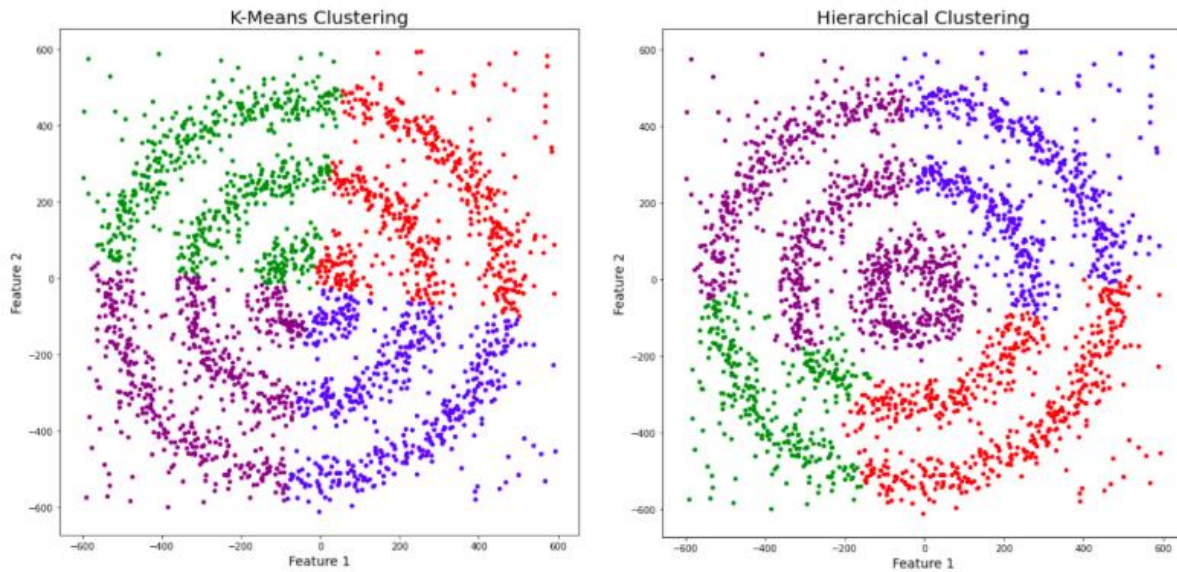


Figure 14 Concentric circles dataset – example n.1

There are three different clusters, which are represented by the concentric circles; also, there is some noise present in the dataset. *FIGURE 15* shows the results of the K-Means clustering and the Hierarchical clustering on the data mentioned above.



*Figure 15 Comparison between K-means and hierarchical clustering on example 1*

The clusters found by the algorithms are presented differentiated by colours. Both algorithms failed to correctly cluster the data points, splitting the data in three clusters plus one containing noise (the purple one), which is clearly not detected in a proper way.

*FIGURE 16* shows the same dataset analysed by the DBSCAN algorithm.



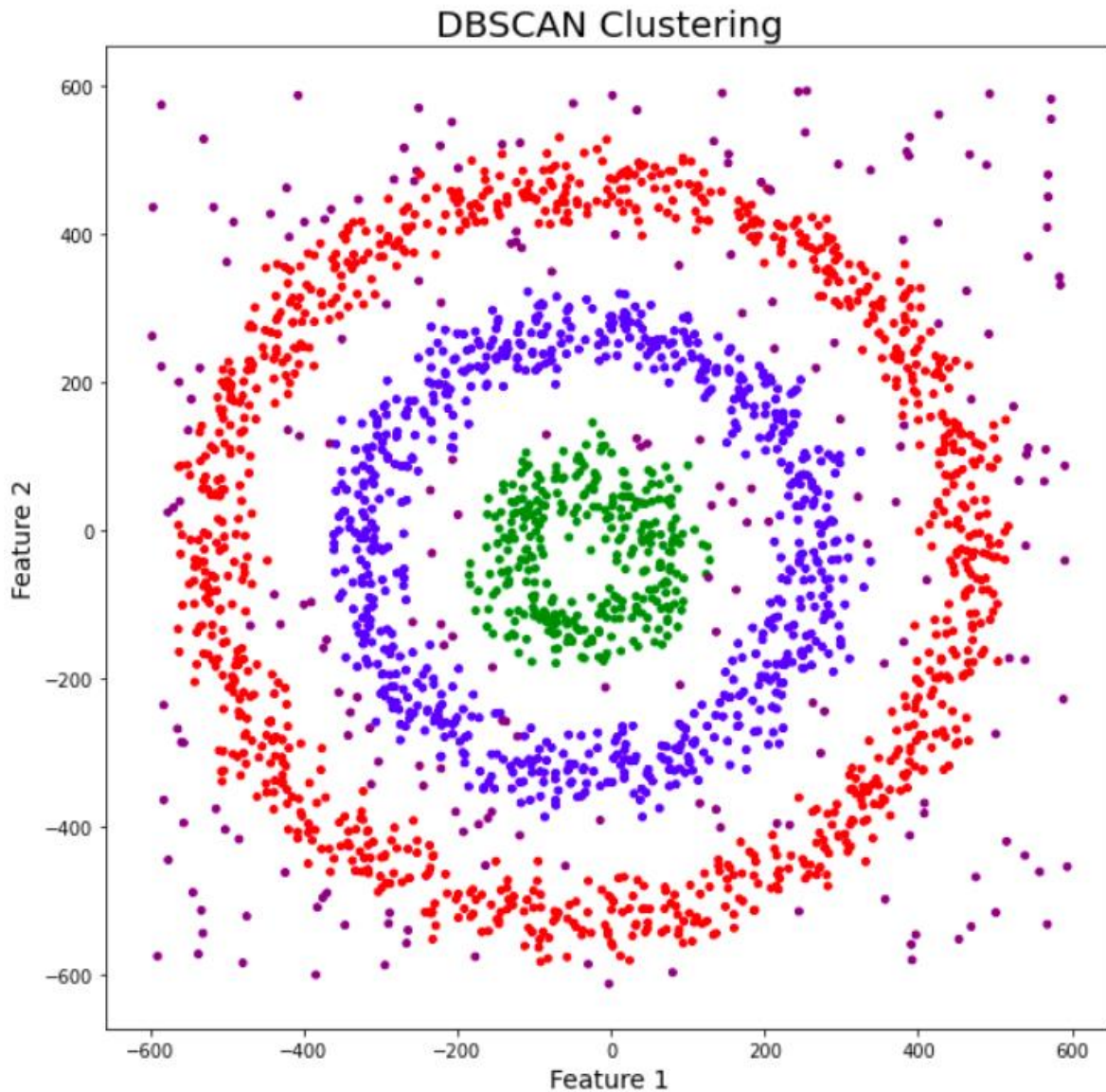


Figure 16 DBSCAN clustering on example 1 dataset

The result obtained is undoubtedly better than the other previous results; there are three concentric clusters, and the noise has also been correctly spotted.

#### 4.4.1 How does the DBSCAN works?

DBSCAN is a density-based clustering algorithm that works on the assumption that clusters are *dense regions in space separated by regions of lower density*. This causes densely grouped data points to be gathered into a single cluster.

One of the most interesting features about this algorithm is that it does not get affected by outliers; also, it is not necessary to specify the number of clusters (or, to be more

specific, the number of centroids), unlike the K-Means which requires this parameter to start the learning process. By all means, it does depend upon two other parameters:

- The radius of the circle that is created around each point to check for density conditions
- The minimum number of points inside that circle for one point to be considered a **Core** point

Another benefit of using the DBSCAN algorithm is that it has to scan the dataset only once, while other procedures have to do it multiple times.

It follows an example of the procedure this algorithm carries out. It is now considered the dataset of:



Figure 17 Dataset example n.2

DBSCAN creates a circle around every point and classifies these points as **core points**, **border points** and **noise**, following these rules:

- A point is a core point if the circle around it contains a number of points large at least the number given as parameter of the algorithm;
- A point is a border point if the circle around it contains a number of points less than the number given as parameter of the algorithm;
- A point is a noise if there are no other points around it within the radius given as parameter.

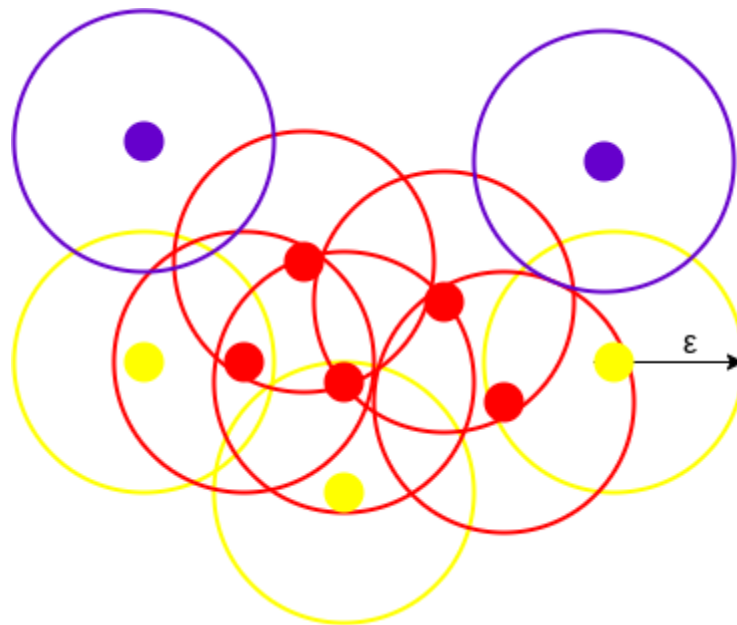


Figure 18 DBSCAN clustering on example 1 dataset

## 4.5 The HDBSCAN algorithm

HDBSCAN is a clustering algorithm which extends DBSCAN by converting it into a hierarchical clustering algorithm, and then using a technique to extract a flat clustering based on the stability of clusters. It is open source and it is developed by Campello, Moulavi, and Sander [12].

Existing density-based clustering methods, one of them being the DBSCAN, can only provide a non-hierarchical labelling of the data objects, based on a single density threshold; a procedure like this can often not properly characterize datasets with clusters of very different densities or nested clusters.

The HDBSCAN algorithm generates a complete density-based clustering hierarchy from which a simplified hierarchy composed only of the most significant clusters can be easily extracted, taking advantage on an ad-hoc measure defined by the authors of the algorithm.

The algorithm starts by identifying as outliers each point that has low density of points in its radius; the aim will be to make these points more distant from each other and from the to-be clusters. As a simple estimation of density, it uses the distance to the  $k$ -th nearest neighbour, which is called **core distance** for parameter  $k$  for a point  $x$  and it is denoted as  $\text{core}_k(x)$ .



To spread apart the points with low density a new metric called **mutual reachability distance** is created and it is defined as the maximum value between:

- the core distance of the first point;
- the core distance of the second point;
- the standard distance between the first and the second point.

Using this new metric, it is possible to be sure that each sparse point will be pushed away from any other point by at least their own core distance. Obviously, the choice of  $k$  is deal-breaking because larger values for this parameter will create more outliers.

To make an example, Figure 19 shows a representation where the core distance is defined using a value of 5 for the parameter  $k$ :

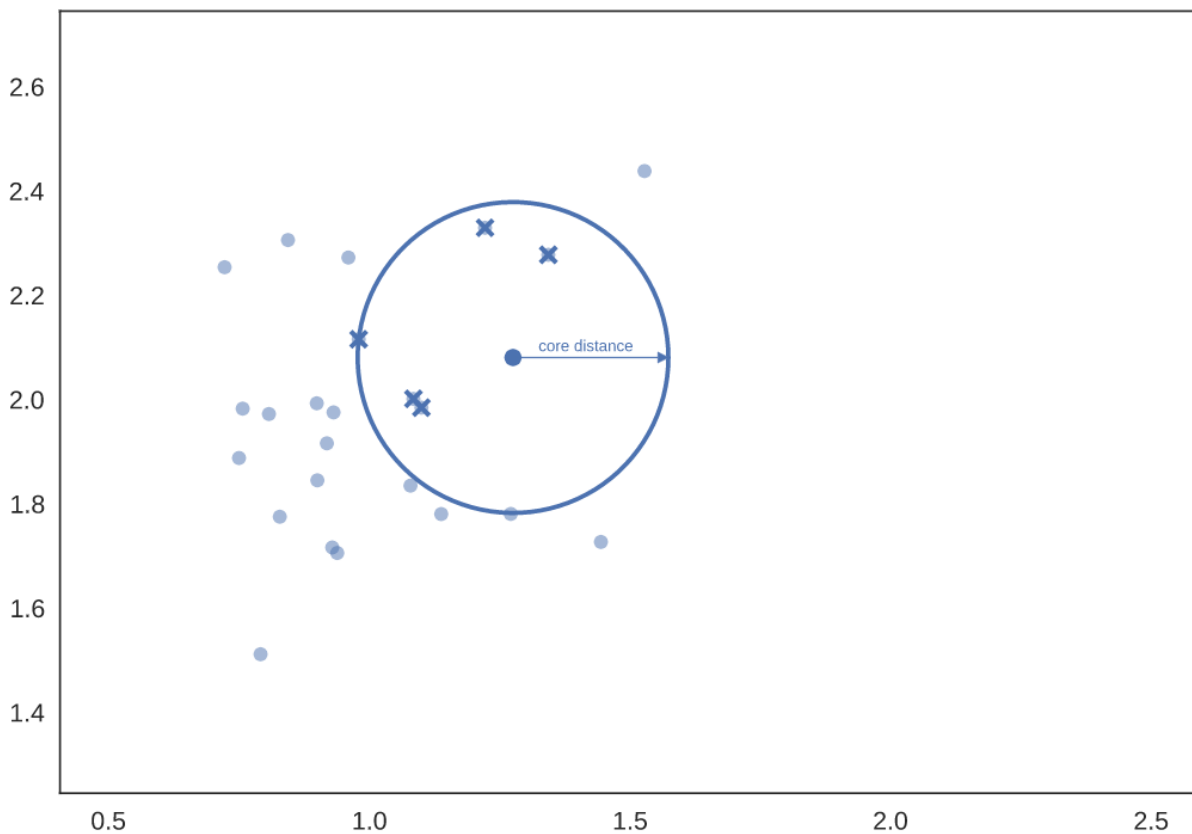


Figure 19 First point selection - Core distance definition with  $k=5$

If another point is selected, it probably gets a circle with a different radius, such as shown in [FIGURE 20](#).

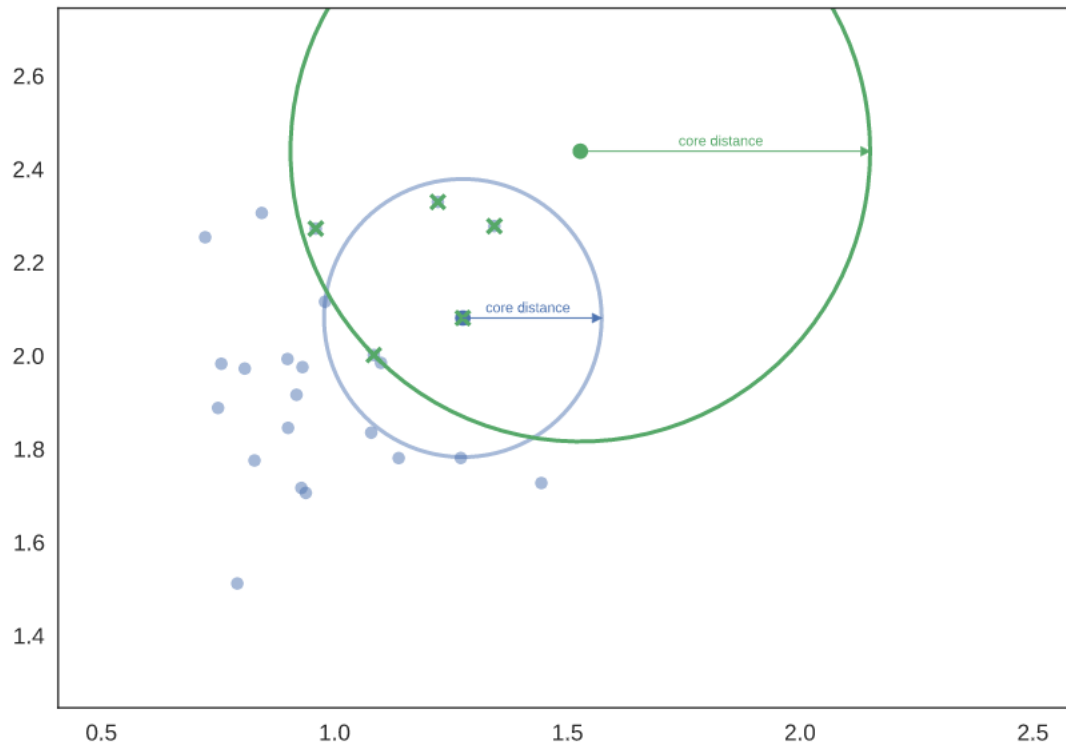


Figure 20 Second point selection

In **FIGURE 21** a third point is selected and its core distance is defined ( $k=6$ ):

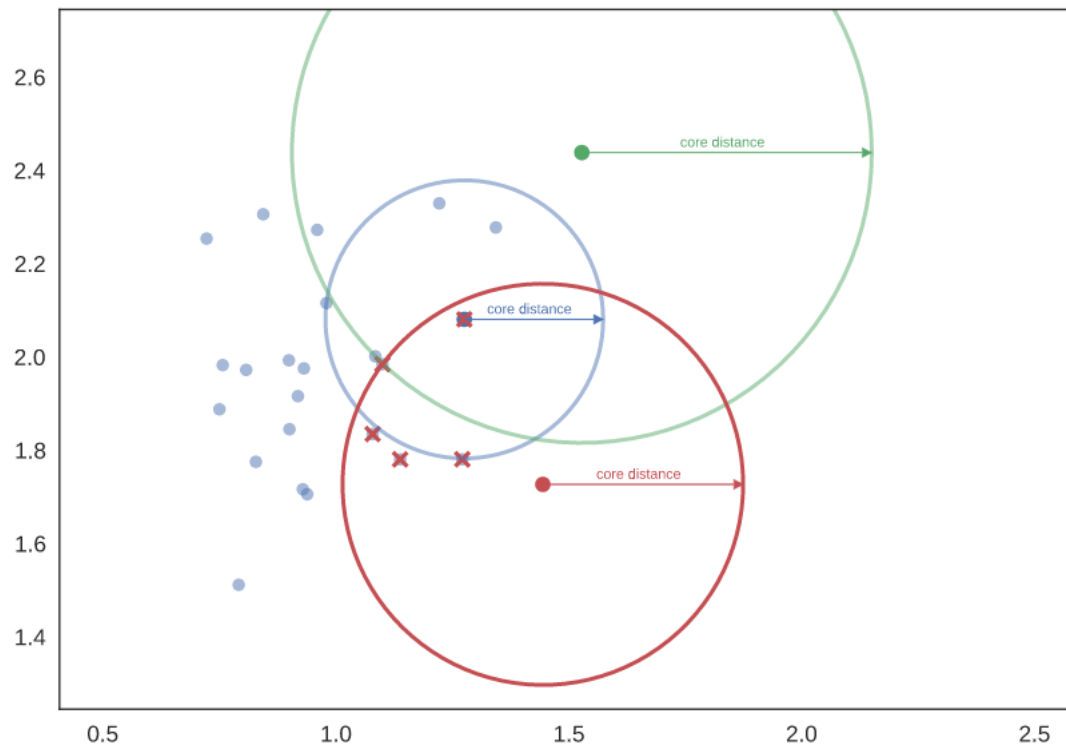


Figure 21 Third point selection

It is possible to know the mutual reachability distance between the blue and the green points, represented by the arrow that goes from one point to the other (FIGURE 22):

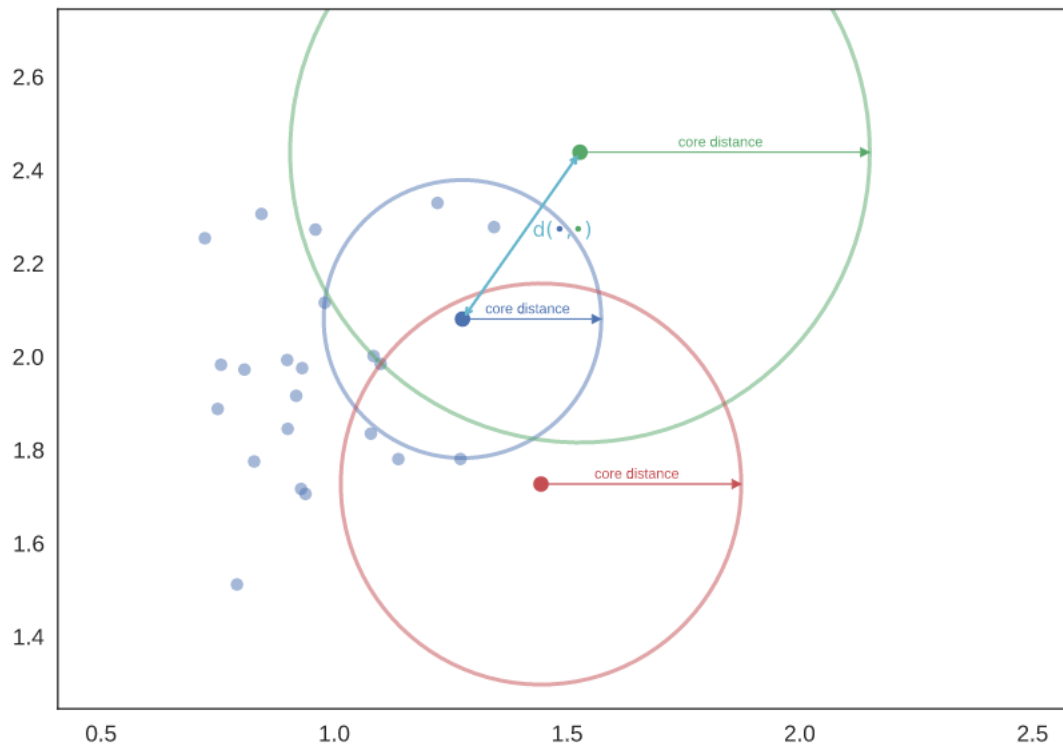


Figure 22 Reachability distance between the blue and the green points

This arrow passes along the blue circle, but not the green (it does not reach the full length of the core distance radius). Therefore, the mutual reachability distance is the green point's core distance, which has the larger value.

It is followed considered the distance between red and green points (Figure 23).

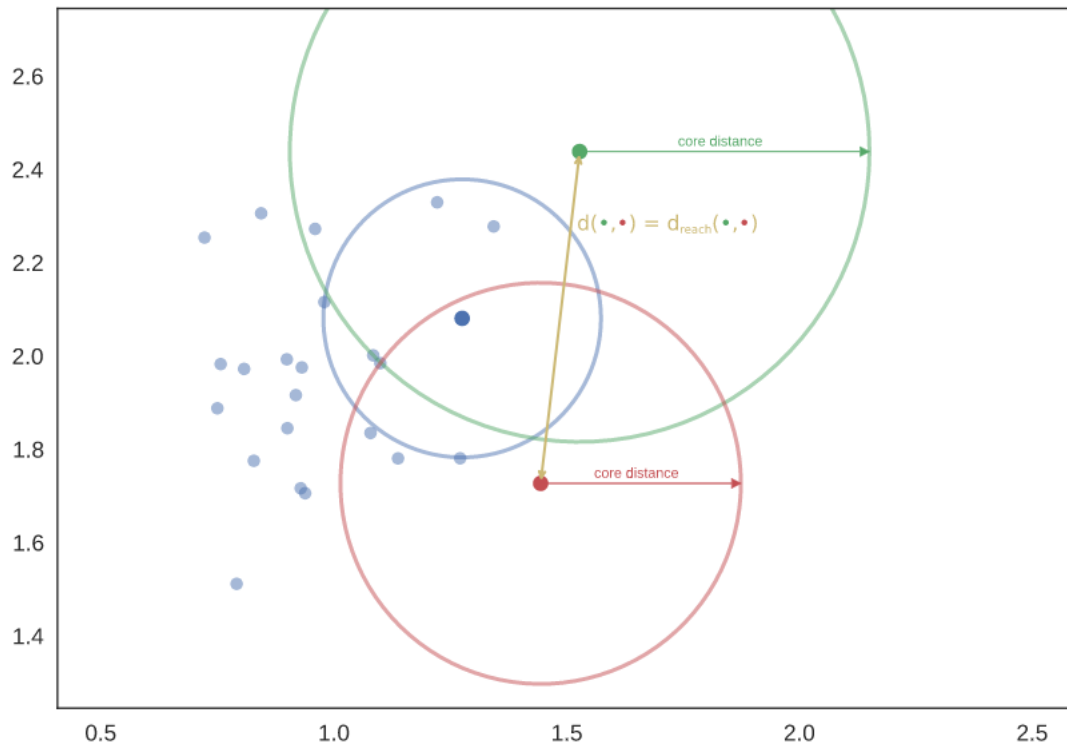


Figure 23 Reachability distance between the green and the red points

The standard distance between the two points is actually greater than the core distance of each of these points, therefore the mutual reachability distance will be the value of  $d(\text{red}, \text{green})$ .

To build the clusters, it will consider the data as a weighted graph with data points as vertices; the graph will have an edge for each two points and its weight will be the MRD between those points.

Now, set a high threshold value, and drop any edges with weight above said threshold; then lower the threshold and repeat the drop task. It will get a hierarchy of connected components at different threshold levels, from completely connected to completely disconnected.

This process is actually really computationally intense; it is necessary to find a minimal set of edges such that dropping any of them will cause a disconnection of components. It is possible then to use the minimum spanning tree of the graph. To create it, the Prim's algorithm has been exploited, adding one edge at a time, and it will always be the edge with lowest weight that connects the current tree to a vertex not yet in the tree (FIGURE 24).

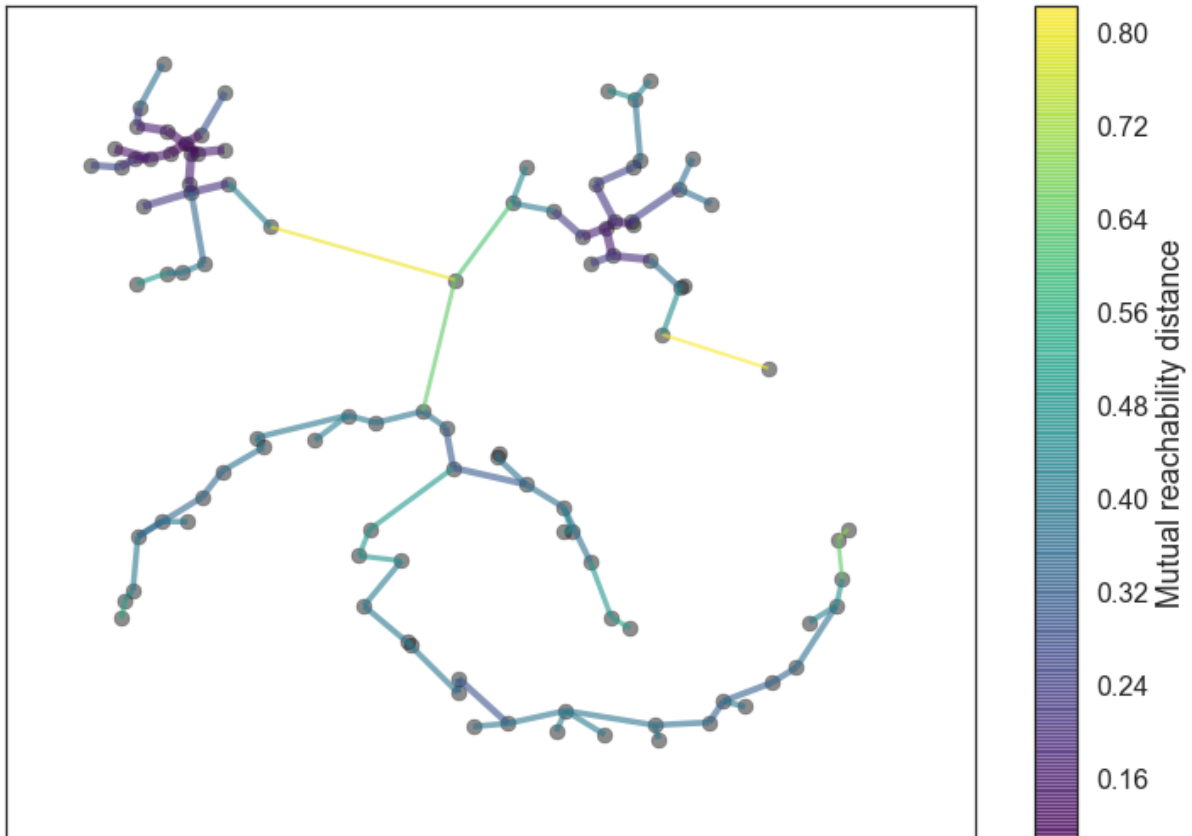


Figure 24 Minimal tree graph using the Prim's algorithm

Now that the spanning tree is created, it is possible to convert it into the hierarchy of connected components. This can be done sorting the edges of tree in increasing distance, and then sequentially creating a new merged cluster through the iteration of this ordered list. In [FIGURE 25](#) shows a dendrogram of the result.

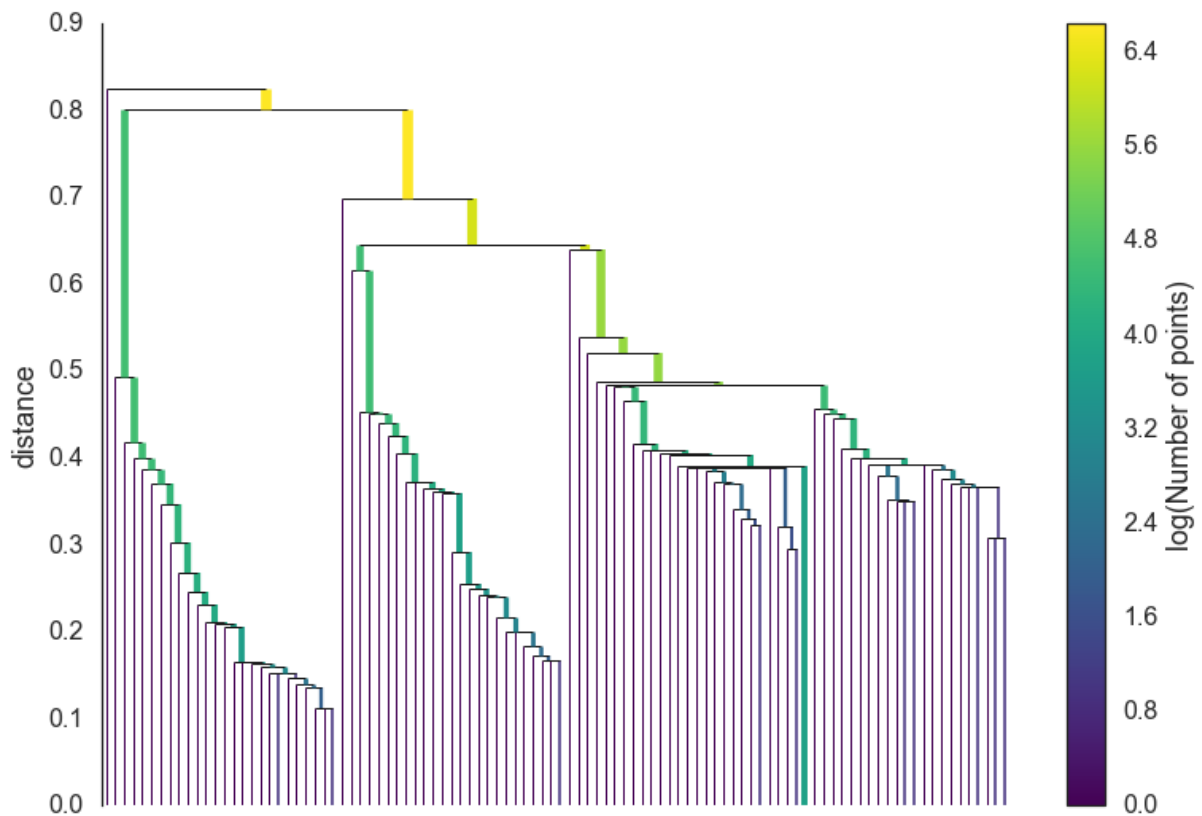


Figure 25 Dendrogram of the edges of the tree sorted in increasing distance

The next step is to extract a set of flat clusters. To do that, the hierarchy is condensed into a smaller tree where each node will be given additional data. This task can be accomplished first by selecting a minimum cluster size (just like the DBSCAN algorithm) and then iterating each cluster split.

If a split creates two new subclusters, both of which have fewer points than the minimum value, then the larger cluster stays is labelled as a "parent" and is also enriched with two new properties: which points have been removed from it during that split and the distance value for which the split is referring.

On the other hand, if the split generates two new clusters at least as large as the minimum cluster size, then we let that split persist on the final hierarchy, and we get rid of the parent cluster.

The process ends up with a smaller tree whose nodes have data about points lost during a specific split. We can again use a dendrogram to view the result using a minimum cluster size of 5 (Figure 26).

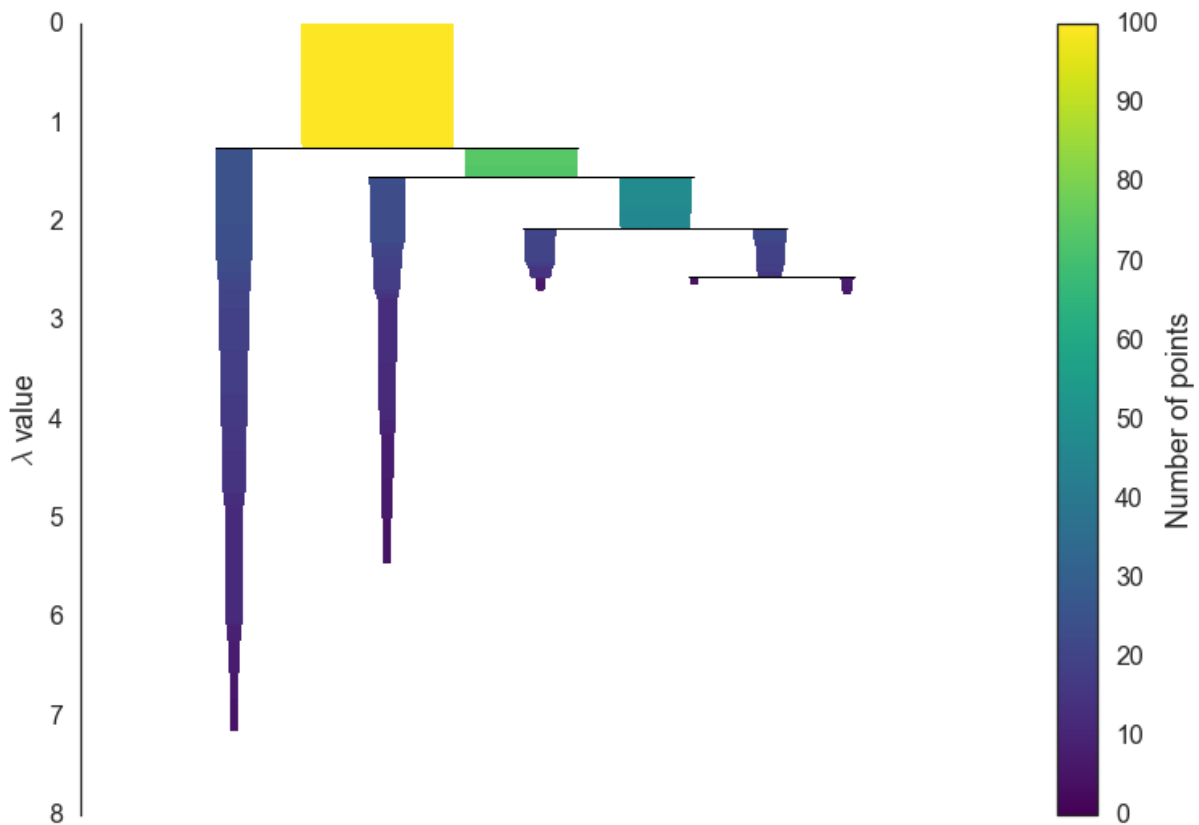


Figure 26 Reduced tree using a minimum cluster size of value 5

The final task is to pick out the clusters to use as flat clustering. Of course, clusters that persist and have a longer lifetime will be chosen, because short lived clusters are probably artifacts of early phases of the algorithm. Looking at the graph, it is reasonable to say that it is necessary to choose the clusters that have the greatest area. To make it a flat clustering, however, it is not allowed to select any cluster that is a descendent of an already selected cluster.

As soon as the final clusters are selected, it is possible to finally say that any point not in them will be an anomaly.

## 4.6 Services architecture

Inside the Energy monitoring module, the Anomaly detection service is designed as depicted in the following paragraphs. Figure 27 shows a representation of the whole architecture.

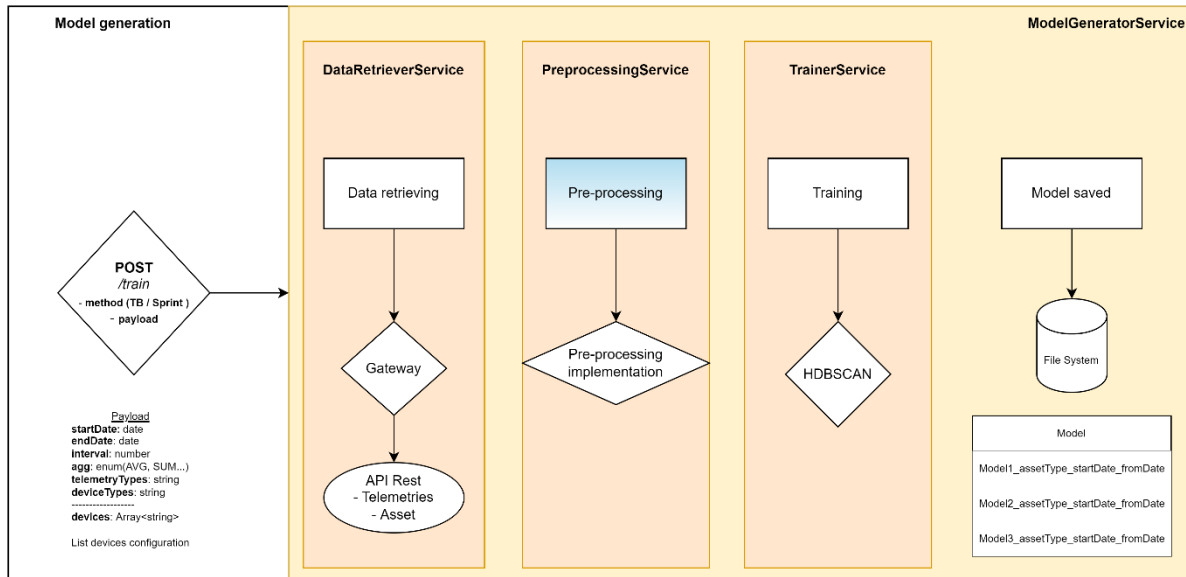


Figure 27 Fault anomaly detection service architecture

## 4.6.1 Model generation

The first step is the generation of the model related to a specific system; in particular, the work at the moment focused on the HVAC asset.

The *ModelGeneratorService* take care of the different components that orchestrate the creation and training of the model, and save it in the file system.

## 4.6.2 Data retrieving

The DataRetrievingService controller allows to create a REST API /POST request, to retrieve the following parameters:

- method, a *string* (from an enumeration) which explain how to retrieve the telemetries;
- startDate, a *timestamp* to select the starting timeframe of the telemetry;
- endDate, a *timestamp* to select the ending timeframe of the telemetry;
- interval, a value to specify the time resolution of the data retrieved;
- agg, a string value (from an enumeration) to specify post processing action on the retrieved data (e.g.: *AVG*, *SUM*, *MAX*, *MIN*);
- telemetryTypes, an array of strings to specify the type of telemetry (e.g.: *energy*, *power*);
- deviceTypes, an array of strings to specify the type of device (e.g.: "smart-meter", "smart-plug");
- devices, an array of strings to specify the device ID;



- `assetTypes`, an array of strings to specify the type of assets (e.g.: "HVAC", "PV system");
- `modelConfiguration`, a JSON object defining the specs configuration of the machine learning model.

After the controller settings, the `DataRetrieverService` will come into play, whose task is to retrieve the various telemetry using the specifications identified by the payload above. The service exploits the `payload` method to obtain the different telemetry, as well as the asset associated with these telemetry. It is assumed that the required devices are all of the same asset type (example: HVAC) as the model will base its training on this asset type.

To ensure that the `ModelGeneratorService` passes the correct information to the `DataRetrieverService`, it is useful to define a `Object` representing the resource, so that it can then be passed between the services in a concise way.

The `DataRetrieverService` outputs are:

- The asset
- The related telemetry list;

### 4.6.3 Data preprocessing

Once the data has been retrieved, the `PreprocessingService` will refine the data and segment it. The service take care of obtaining the necessary configuration for the entire component (not yet defined way, it can be a properties file or a remote file), carrying out cleaning operations and grouping the telemetry into segments, as well as performing (possibly) the extraction of the features useful to create the model.

The result of the preprocessing is returned as output by the service, in particular an array will be provided for each telemetry, specifying the features associated with the telemetry.

### 4.6.4 Model training

Here, the `ModelGeneratorService` calls the `TrainerService`, whose task will be to train the model with the refined data through by the `preprocessingService`.

### 4.6.5 Model saving

After the model creation, it is necessary to save it, for example, in the filesystem. Metadata will allow the model to be identified, in order to be able to reuse it in the anomaly detection phase.

At the moment, only the model related to the HVAC system has been created.

The metadata to record are:

- The device type related to the model
- The time interval of the model (fromTime - toTime)

Figure 28 recaps the flowchart of the Anomaly detection service implementation.

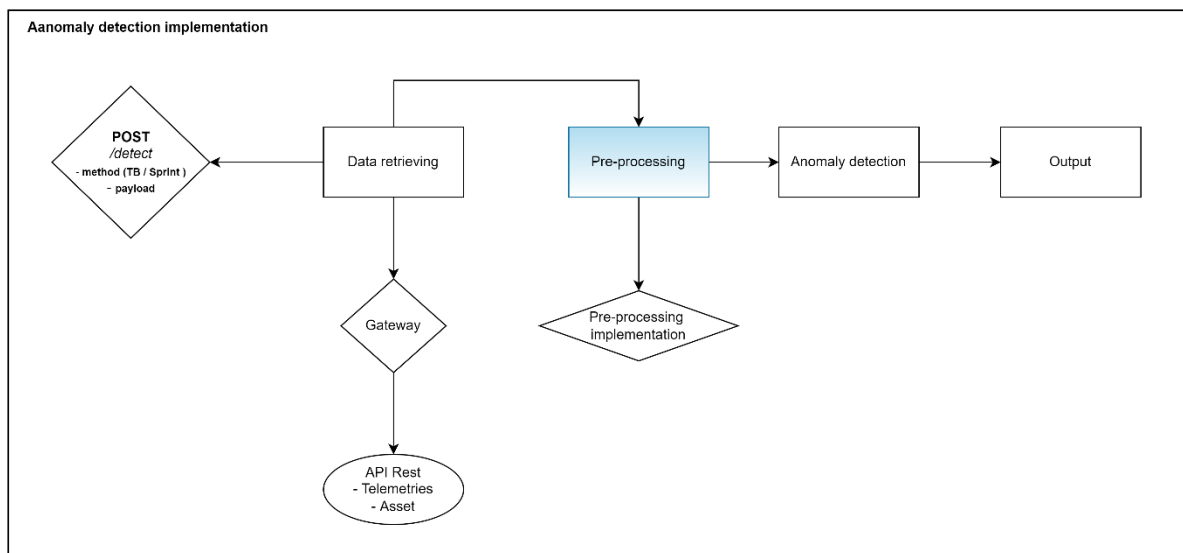


Figure 28 Anomaly detection service implementation

## 4.6.6 The anomaly detection service

Finally, the AnomalyHandlerService can run. Its functioning is largely similar to the ModelGeneratorService, as it also uses the DataRetrieverService and PreprocessingService services to retrieve data and carry out its refinement, segmentation and feature extraction.

The main difference lies in the following steps, where instead of the TrainerService the AnomalyDetectionService is called. It takes a model suitable for analysing the data obtained through the data retrieving service and using it to identify possible anomalies. By suitable model, a model with an asset type identical to that to which the retrieved telemetry refers is meant, and with a timeframe that is effectively useful for identifying anomalous points.

Once the anomalies have been identified, they are returned in the form of an array, taking care to specify, for each of them, the following proprieties:

- startDate – a timestamp describing the starting timeframe of the anomaly;
- endDate - a timestamp describing the ending timeframe of the anomaly;
- anomalyType – the type of anomaly (overpower, overconsumption, ecc.);
- expectedValue – the expected value of the telemetry (as explained in the previous chapter, it is normally considered the cluster centroid);
- registeredValue – the registered value of the telemetry;
- score, a value describing the distance of the registered value from the centroid of the nearest cluster

Another point to highlight about the differences of the AnomalyHandlerService service compared to the ModelGeneratorService service is that the payload requested by the former must not contain any specifications on the configuration of the model, since the latter will be retrieved autonomously by the AnomalyDetectionService service.

The model requires a general configuration (the minimum duration of the anomaly, the maximum iterations to achieve model convergence) and a specific configuration. The general configuration can be entered, for example, via properties file.

Instead, the specific configuration of the ML model can potentially be set in two different ways:

- Requesting the configuration in the payload of the initial POST, so that it is then passed to the TrainerService service;
- Leaving the TrainerService to retrieve it on its own.

The choice of one of these two options conditions the presence of the modelConfig property both in the payload of the HTTP request and in the input of the TrainerService.

In the Annex A, at the end of the document, the input and output JSON files exchanged by the Anomaly Detection algorithm components are listed.

## 4.7 Fault Anomaly Detection GUI

In a specific page of the GUI developed within the T7.4 deploys the anomaly detection results dashboard.

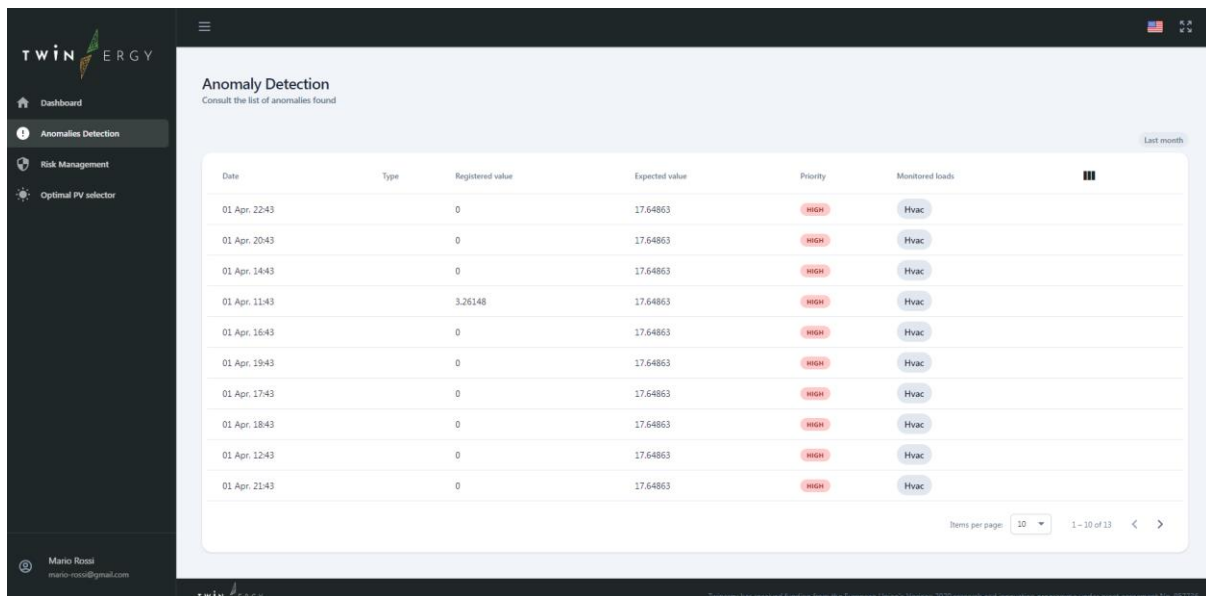
With a 24h scheduling, the detection algorithm runs, and the results are shown in a table (*FIGURE 29*)

Every time an anomaly is detected, it increases the list in the table. In the table provide information about:

- The datetime of the anomaly

- The type of anomaly detected
- The registered value
- The expected value
- The priority – intended as severity (based on the score index in output from the detection algorithm)
- The system involved

The dashboard is deployed online at <https://twenergy.stamtech.dev/anomalies>.



Date	Type	Registered value	Expected value	Priority	Monitored loads
01 Apr, 22:43		0	17.64863	HIGH	Hvac
01 Apr, 20:43		0	17.64863	HIGH	Hvac
01 Apr, 14:43		0	17.64863	HIGH	Hvac
01 Apr, 11:43		3.26148	17.64863	HIGH	Hvac
01 Apr, 16:43		0	17.64863	HIGH	Hvac
01 Apr, 19:43		0	17.64863	HIGH	Hvac
01 Apr, 17:43		0	17.64863	HIGH	Hvac
01 Apr, 18:43		0	17.64863	HIGH	Hvac
01 Apr, 12:43		0	17.64863	HIGH	Hvac
01 Apr, 21:43		0	17.64863	HIGH	Hvac

Figure 29 Anomaly Detection service dashboard

## 5 TwinERGY Platform Integration

To allow the Energy Management Module a fast and easy implementation on the pilots it is mandatory to reach a big amount of the information storage inside the Digital Twin.

It is necessary to know:

- The building configuration
- The appliances inventory
- The pairing between appliances and smart meter

To do that a preliminary work on the Saref ontology<sup>2</sup> took place in the D7.1- Interoperability deliverable, in order to discuss with the TwinERGY partners on the Data models their information objects definition.

For the T7.4 module in particular, the Saref4Bldg<sup>3</sup> and Saref4Ener<sup>4</sup> ontology extension have been exploited. Modification and update in the data model will take place in agreement with the other partners during the integration phase, after M18.

### 5.1 Optimal RES selector

A second service would provide suggestion about the best usage of the Renewable energy source (with a focus on the photovoltaics system). The goal is to wrap up some of the main outputs of the T7.3 and T7.5, related to the renewable energy generation and to provide them in a user-friendly way in the platform.

This action has been postponed and it will take place during the integration phase.

---

<sup>2</sup> <https://saref.etsi.org/>

<sup>3</sup> <https://saref.etsi.org/saref4bldg/v1.1.2/>

<sup>4</sup> <https://saref.etsi.org/saref4ener/v1.1.2/>

---

## 6 Conclusion

This document has described the functional aspects of the H&T Energy Management module. Specifically, the goal of the module is to increase the users' awareness about their energy consumption patterns and habits. The lack of clear and easy to read information often leads the users on bad practices, energy waste and high energy bills.

The main goal of the module was to avoid this path, show the monitored data, and to introduce the users to a preliminary manual control of the appliances. The second step foreseen by the module is the analysis of the energy monitored data, to understand the anomalies in the energy demand of the appliances and in the energy generation of the PV system. At the moment, the Anomaly Detection algorithm still lack in the recognition of the specific anomaly type detected. The lack will be covered during the operational phase of the Twinergy projects in the pilots.

The module is deployed online at <https://twinergy.stamtech.dev/dashboard>.

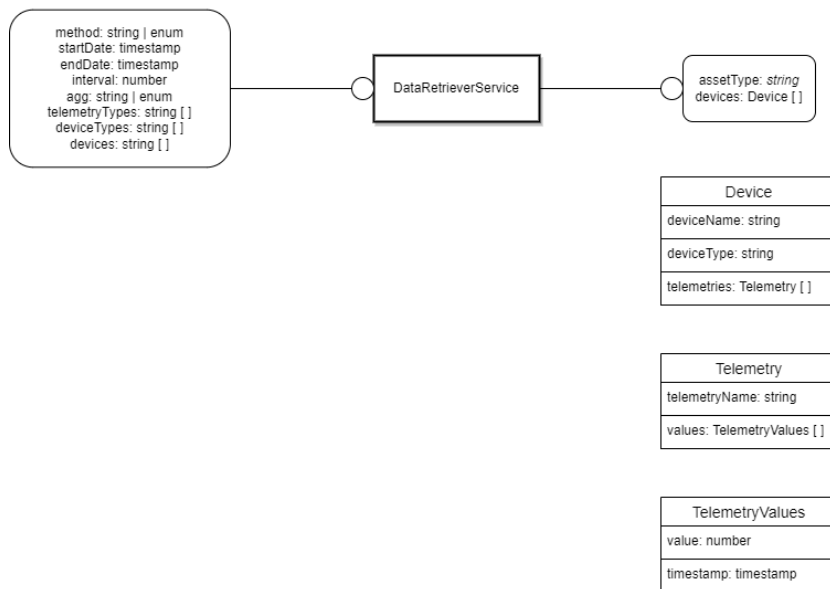
Currently no authentication is required. It will follow the integration phase.

# Annexes

## Annex A - Input and output of the Anomaly Detection algorithm components

In the present annex A are shown the the diagrams are shown describing the various input and output values of the components described above. For each, a sample information object in JSON format is also provided.

### DataRetrieverService



JSON input:

```

{
  "startDate": "20211222T153000+0100",
  "endDate": "20101014T121200+0200",
  "interval": 5,
  "agg": "AVG",
  "telemetryTypes": ["energy", "power"],
  "deviceTypes": ["shelly-em", "tp-link"],
  "assetType": "HVAC",
  "devices": [
    "cb5a0c1d-2666-4b77-b7be-8dd4c895ce55",
    "20c8ea8b-b927-46fc-b45c-8faddf3a839c"
  ]
}
  
```

}

JSON output:

```
{
  "assetType": "HVAC",
  "interval": 5,
  "devices": [
    {
      "deviceName": "device1",
      "deviceType": "shelly-em",
      "telemetries": [
        {
          "telemetryName": "energy",
          "values": [
            {
              "value": 250,
              "timestamp": "20211222T161000+0100"
            },
            {
              "value": 249,
              "timestamp": "20211222T162000+0100"
            },
            {
              "value": 251,
              "timestamp": "20211222T163000+0100"
            }
          ]
        },
        {
          "telemetryName": "temperature",
          "values": [
            {
              "value": 25,
              "timestamp": "20211222T160200+0100"
            },
            {
              "value": 24,
              "timestamp": "20211222T160800+0100"
            },
            {
              "value": 24,
              "timestamp": "20211222T161400+0100"
            }
          ]
        }
      ]
    }
  ]
}
```

*PreprocessingService*





Device
deviceName: string
deviceType: string
telemetries: Telemetry []

PreprocessingResult
telemetryName: string
segmentDuration: number
segmentFeatures: SegmentFeature []

Telemetry
telemetryName: string
values: TelemetryValues []

SegmentFeature
featureName: string
featureValue: number

TelemetryValues
value: number
timestamp: timestamp

### JSON input:

```

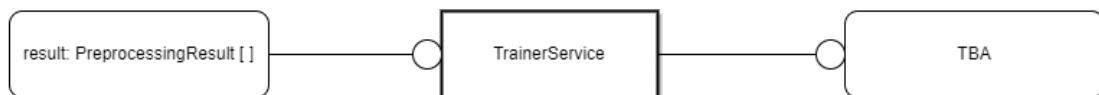
{
  "assetType": "HVAC",
  "interval": 5,
  "devices": [
    {
      "deviceName": "device1",
      "deviceType": "shelly-em",
      "telemetries": [
        {
          "telemetryName": "energy",
          "values": [
            {
              "value": 250,
              "timestamp": "20211222T161000+0100"
            },
            {
              "value": 249,
              "timestamp": "20211222T162000+0100"
            },
            {
              "value": 251,
              "timestamp": "20211222T163000+0100"
            }
          ]
        }
      ]
    },
    {
      "telemetryName": "temperature",
    }
  ]
}
  
```

```

    "values": [
      {
        "value": 25,
        "timestamp": "20211222T160200+0100"
      },
      {
        "value": 24,
        "timestamp": "20211222T160800+0100"
      },
      {
        "value": 24,
        "timestamp": "20211222T161400+0100"
      }
    ]
  }
}

```

### TrainerService



PreprocessingResult
telemetryName: string
segmentDuration: number
segmentFeatures: SegmentFeature []

SegmentFeature
featureName: string
featureValue: number

### JSON input:

```

{
  "kmeanConfig": {
    "minAnomalyDuration": 10,
    "maxIteration": 5,
    "clustersCount": 5,
    "distanceFunction": "EUCL",
    "scoreThreshold": 15
  },
  "preprocessingResult": [
    {
      "telemetryName": "energy",
      "segmentFeatures": [
        {

```

```

    "featureName": "feature1",
    "featureValue": 1
  }
]
}

```

JSON output TBA

### TrainerService with model



PreprocessingResult
telemetryName: string
segmentDuration: number
segmentFeatures: SegmentFeature []

SegmentFeature
featureName: string
featureValue: number

JSON input:

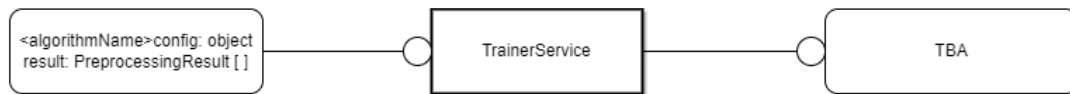
```

{
  "kmeanConfig": {
    "minAnomalyDuration": 10,
    "maxIteration": 5,
    "clustersCount": 5,
    "distanceFunction": "EUCL",
    "scoreThreshold": 15
  },
  "preprocessingResult": [
    {
      "telemetryName": "energy",
      "segmentFeatures": [
        {
          "featureName": "feature1",
          "featureValue": 1
        }
      ]
    }
  ]
}

```

JSON output TBA

## TrainerService con configurazione modello



PreprocessingResult
telemetryName: string
segmentDuration: number
segmentFeatures: SegmentFeature []

SegmentFeature
featureName: string
featureValue: number

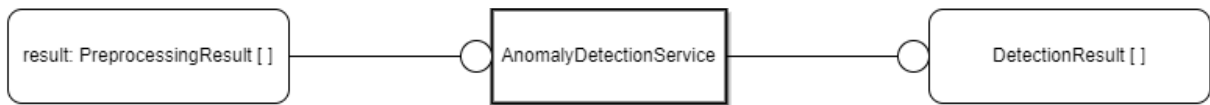
JSON input:

```

{
  "kmeanConfig": {
    "minAnomalyDuration": 10,
    "maxIteration": 5,
    "clustersCount": 5,
    "distanceFunction": "EUCL",
    "scoreThreshold": 15
  },
  "preprocessingResult": [
    {
      "telemetryName": "energy",
      "segmentFeatures": [
        {
          "featureName": "feature1",
          "featureValue": 1
        }
      ]
    }
  ]
}
  
```

JSON output TBA

## AnomalyDetectionService



PreprocessingResult
telemetryName: string
segmentDuration: number
segmentFeatures: SegmentFeature []

SegmentFeature
featureName: string
featureValue: number

DetectionResult
anomalyType: string
startDate: timestamp
endDate: timestamp
expectedValue: number
registeredValue: number
score: number

JSON input:

```

{
  "preprocessingResult": [
    {
      "telemetryName": "energy",
      "segmentDuration": 10,
      "segmentFeatures": [
        {
          "featureName": "feature1",
          "featureValue": 1
        }
      ]
    }
  ]
}
  
```

JSON output:

```

[
  {
    "anomalyType": "OVERPOWER",
    "startDate": "20211222T153000+0100",
    "endDate": "20211222T155000+0100",
    "expectedValue": 10,
    "registeredValue": 15,
    "score": 12
  },
  {
    "anomalyType": "OVERPOWER",
    "startDate": "20211222T153000+0100",
    "endDate": "20211222T155000+0100",
    "expectedValue": 10,
    "registeredValue": 15,
    "score": 12
  }
]
  
```

---

```
{
  "anomalyType": "OVERPOWER",
  "startDate": "20211222T153000+0100",
  "endDate": "20211222T155000+0100",
  "expectedValue": 10,
  "registeredValue": 15,
  "score": 12
}
```

# References

- [1] A. W. L. Yao and C. H. Ku, "Developing a PC-based automated monitoring and control platform for electric power systems," 2004. [Online]. Available: [www.elsevier.com/locate/epsr](http://www.elsevier.com/locate/epsr)
- [2] M. Sechilariu, "Intelligent energy management of electrical power systems," *Applied Sciences (Switzerland)*, vol. 10, no. 8, Apr. 2020, doi: 10.3390/APP10082951.
- [3] R. Bayindir, E. Irmak, I. Colak, and A. Bektas, "Development of a real time energy monitoring platform," *International Journal of Electrical Power and Energy Systems*, vol. 33, no. 1, pp. 137–146, Jan. 2011, doi: 10.1016/j.ijepes.2010.06.018.
- [4] J. C. Ferreira, J. A. Afonso, V. Monteiro, and J. L. Afonso, "An energy management platform for public buildings," *Electronics (Switzerland)*, vol. 7, no. 11, Nov. 2018, doi: 10.3390/electronics7110294.
- [5] M. Alabiso, A. Ardito, and A. Capozza, "Contributo delle elettrotecnologie per usi finali al carico di punta," *ECORET/workpackage 1 (PRECA)/milestone 1.2 (CAREL)*, pp. 1–90, 2005.
- [6] Giuseppe Anastasi and Francesco Corucci, *An Intelligent System for Electrical Energy Management in Buildings*. 2011.
- [7] A. M. Vega, F. Santamaria, and E. Rivas, "Modeling for home electric energy management: A review," *Renewable and Sustainable Energy Reviews*, vol. 52. Elsevier Ltd, pp. 948–959, Aug. 22, 2015. doi: 10.1016/j.rser.2015.07.023.
- [8] A. Fensel, V. Kumar, and S. D. K. Tomic, "End-user interfaces for energy-efficient semantically enabled smart homes," *Energy Efficiency*, vol. 7, no. 4, pp. 655–675, 2014, doi: 10.1007/s12053-013-9246-2.
- [9] SUNIT PRASAD, "Different Types of Clustering Methods and Applications." <https://www.analytixlabs.co.in/blog/types-of-clustering-algorithms/> (accessed Mar. 26, 2022).
- [10] GitHub, "How HDBSCAN Works." [https://hdbscan.readthedocs.io/en/latest/how\\_hdbscan\\_works.html#how-hdbscan-works](https://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html#how-hdbscan-works) (accessed Mar. 26, 2022).
- [11] Abhishek Sharma, "How to Master the Popular DBSCAN Clustering Algorithm for Machine Learning." <https://www.analyticsvidhya.com/blog/2020/09/how-dbscan-clustering-works/> (accessed Oct. 08, 2020).
- [12] R. J. G. B. Campello, D. Moulavi, and J. Sander, "Density-Based Clustering Based on Hierarchical Density Estimates," in *Advances in Knowledge Discovery and Data Mining*, 2013, pp. 160–172.