D4.1 Enhanced, user-friendly EMS for residential/building flexibility Discovery and delivery (v1)

7 March 2025

OPENTUNITYproject.eu

Deliverable details

Title	WP	Version
Enhanced, user-friendly EMS for residential/building flexibility Discovery and delivery	4	1.0

Contractual delivery date	Actual delivery date	Delivery type*	Dissemination**
M26 (February 2025)	M26 (February 2025)	DEM	PU

*Delivery type: R: Document, report; DEM: Demonstrator, pilot, prototype; DEC: Websites, patent fillings, videos, etc; OTHER; ETHICS: Ethics requirement; ORDP: Open Research Data Pilot.

Dissemination Level: **PU: Public; **CO**: Confidential, only for members of the consortium (including the Commission Services)

Author(s)	Organization
Christos Malavazos	HYPERTECH
Kakavoulis Stefanos-Dimitris	HYPERTECH
Panos Andriopoulos	HYPERTECH
Tsatsoulis Panagiotis	HYPERTECH
Kotalakidis Alexandros	HYPERTECH
Giorgos Pitsiladis	HYPERTECH
Antonis Papanikolaou	HYPERTECH
Ilias Sarantakos	HYPERTECH

Version	Date	Person	Action	Status***
0.1	23.09.2024	Giorgos Pitsiladis	Table of Content	Draft
0.2	17.01.2025	Tsatsoulis Panagiotis	First content added	Draft
0.3	31.01.2025	Tsatsoulis Panagiotis, Pablo Bort, Tomaž Buh, Federico Giani	Section 2 added	Draft
0.4	31.01.2025	Pablo Bort	Section 4 added (NILM)	Draft

0.5	04.03.2025	Tsatsoulis Panagiotis	Section 3 added (BFMS)	Draft
0.6	07.03.2025	Janez Gregor Golja, Sara Vieira, Natalia Fano	Peer review	Final
1.0	07.03.2025	Tsatsoulis Panagiotis	Version ready to submit	Submitted

***Status: Draft, Final, Approved, Submitted (to European Commission).

Authors (organization)
Pablo Bort (ETRA I+D); Álvaro Nofuentes (ETRA I+D); Tomaž Buh (REDUXI); Federico Giani (AEM)

Keywords

Demand Response (DR), Load Flexibility, Energy Management, Grid Interaction, Energy Optimization, Forecasting Algorithms, Load Prediction, Flexibility Modeling, Real-time Data Analytics, Smart Buildings, IoT Sensors, Load Monitoring

Executive Summary

This Deliverable presents the progress made so far regarding the Building Flexibility Management System (BFMS) and the Non-Intrusive Load Monitoring (NILM) components of the OPENTUNITY project. These two components concern the interaction between Home Energy Management Systems (HEMS), Building Energy Management Systems (BEMS), and various flexibility market stakeholders, leading to disaggregation of energy loads and forecasting of baseline and flexibility power consumption.

The deliverable starts with a description of the HEMS/BEMS that are available at the four pilot sites of the project. These systems are necessary tools for the calculation and dispatch of flexibility, as they are responsible for the seamless communication of data from the pilots, as well as the dispatch of control signal based on flexibility forecast. The main functionalities of the available HEMS/BEMS are presented, focusing on hardware/software setup, performed data collection, supporting IoT devices and interfaces (if available).

BFMS collects energy usage data, sensor information, and occupant preferences to forecast baseline consumption and available flexibility. By analyzing this data and using machine learning algorithms, BFMS determines how much energy can be adjusted and when, enabling buildings to participate in Demand Response (DR) programs. It then dispatches control signals to connected devices, ensuring energy is used efficiently while respecting user constraints. The DR Initialization Service is a novel module of the BFMS, used to collect occupant feedback on available assets and comfort preferences. It ensures a user-friendly experience, allowing consumers to participate in DR programs without compromising comfort or altering their behaviors.

The NILM component provides meaningful energy related insights by analyzing total power consumption and disaggregating it into individual appliance usage using variables like active power, reactive power, current, and voltage. Unsupervised and semi-supervised methods like k-means clustering have been explored, offering a trade-off between reduced accuracy and the benefit of not requiring intrusive monitoring. In this way, real-time appliance power consumption can be predicted, creating generalized models that can be applied to any household with metering data.

By generating accurate consumption curves for flexible assets, NILM supports demand response programs, enabling more effective load management and energy market participation. Together, BFMS and NILM create a powerful tool for the monitoring and optimization of energy consumption, enabling greater engagement in demand response initiatives and flexibility calculation and dispatch.

Copyright statement

INDEX

List of figures

List of tables

1. INTRODUCTION

1.1 Purpose of the document

The purpose of the deliverable is to present the work carried out under Task 4.1 "HEMS and BEMS flexibility (including EV and storage) and DR optimization", Task 4.2 "Initial settings algorithm" and Task 4.3 NILM and behavior analytics of the OPENTUNITY project.

Task 4.1 focuses on extending and adapting Home Energy Management Systems (HEMS) and Building Energy Management Systems (BEMS) to fulfill the requirements of the project (e.g., integration of new interfacing protocols, adjustments for pilot site assets, etc.). In addition, the Building Flexibility Management System (BFMS) is also adjusted and extended based on the needs of the project. Task 4.2 complements the work of T4.1 by introducing an initial settings algorithm that allows users to provide their preferences and constraints, altogether enabling the dispatch of human-centric flexibility. Lastly, Task 4.3 investigates current Non-Intrusive Load Monitoring (NILM) algorithms to select and adapt the best load identification techniques for the metering systems of the project. Following that, the selected NILM methodology is used for data acquisition, event detection, learning, and load disaggregation.

The document provides the development of these three tasks by M26 of the project and outlines the future work until the end of the two tasks, i.e., M34.

1.2 Scope of the document

The deliverable details the activities undertaken to achieve the objectives of Tasks 4.1, 4.2 and 4.3. In a nutshell, this concerns the technical and modelling aspects for the forecast and dispatch of flexibility at the different pilot sites of the project, as well as the disaggregation of energy consumption to separate assets' consumption. Figure 1 provides a high-level representation of the OPENTUNITY energy flexibility market, including different components, tools, and services that facilitate data collection from available assets and enable flexibility forecasting, biding and trading. It is presented here to provide a clear picture of the way that the components developed under Tasks 4.1-4.3 are integrated into the OPENTUNITY ecosystem and of their interactions with the other tools of the project.

Figure 1 presents the different layers of this ecosystem, summarized as follows:

- the pilot sites with their respective assets,
- the HEMS/BEMS,
- the data exchange system,
- the flexibility forecasting tools, and
- the NODES market platform.

Starting with the pilot projects and data collection, the different sites of the project in Switzerland, Spain, Greece, and Slovenia will be testing the integration of flexibility assets into the energy market. Each pilot site provides various flexibility assets, including heating systems, air conditioners, domestic hot water heaters, PVs, and EVs. A distinct **HEMS/BEMS** manages these assets at each pilot, such as the BESOS (Spain), the FLEXO platform (Switzerland), the Smart Box (Greece) and the Reduxi

(Slovenia). These systems are presented in detail in Section 3 (OPENTUNITY HEMS/BEMS) of this deliverable. The HEMS/BEMS stream the necessary power consumption data from the pilots.

The Data Space serves as a central data exchange hub that ensures seamless communication between the various tools and services of the project. It is developed within WP3 of the project and will be presented in the respective deliverables. Among other things, it ensures proper communication between the HEMS/BEMS and the tools of the project.

A suite of digital tools facilitates the identification, optimization, and activation of flexibility resources. Based on data received by the pilot sites, the **BFMS** calculates the baseline consumption of the available assets and forecasts their day-ahead flexibility provision. In addition, the **Demand Response (DR) Initialization Service** provides necessary information on the available assets and on the preferences of the users of these assets. These two tools are presented in Section 4 (Building Flexibility Management System) of this deliverable. The **NILM Load Identification Tool** detects individual electrical loads based on total energy consumption. It is presented in Section 5 (NILM and Behaviour Analytics) of this deliverable. The NILM tool is used in case no sub-metering is available at the consumers premises.

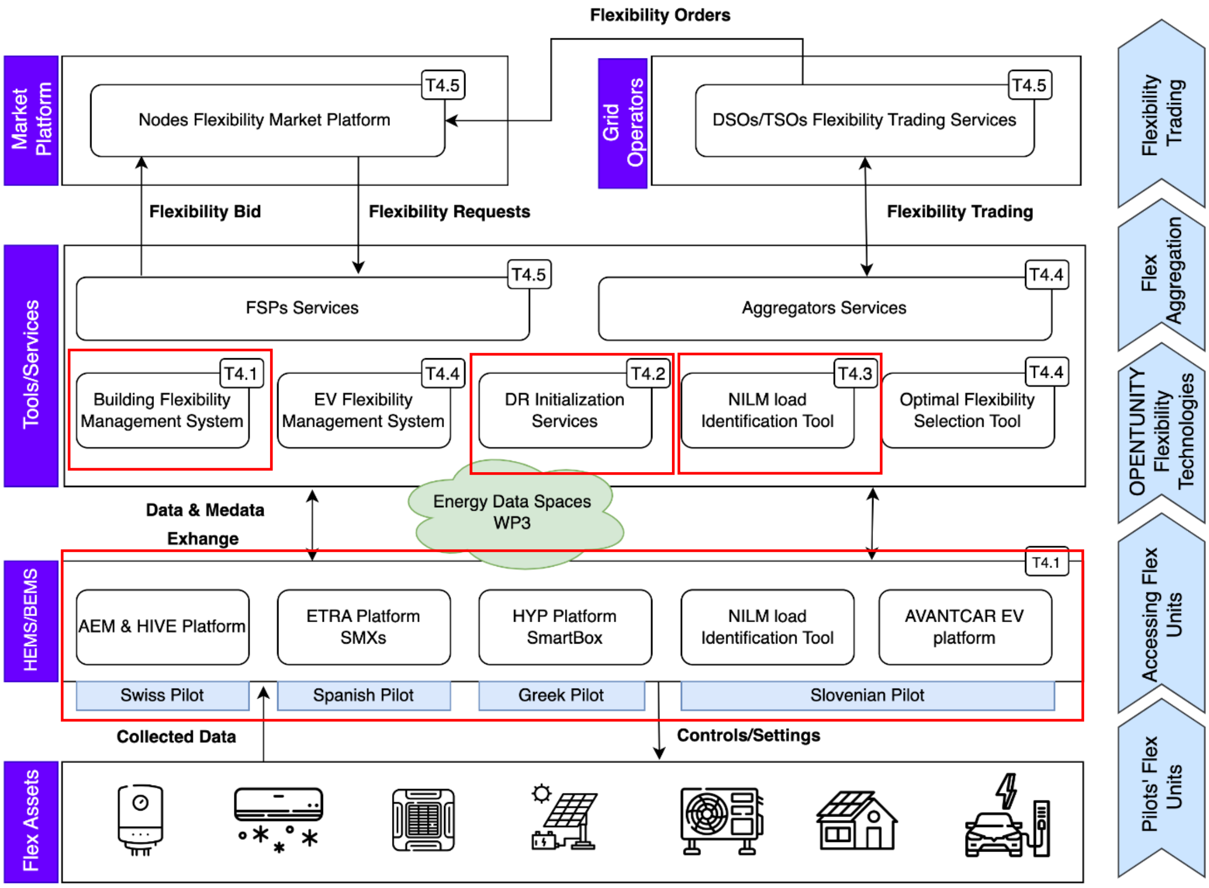


Figure 1: High-level representation of the OPENTUNITY services and tools

1.3 Structure of the document

The deliverable is structured as follows:

The first section, "Introduction", provides an overview of the deliverable, including its purpose, scope, and structure. This section gives a brief overview of the deliverable, highlighting its objectives and relevance to the OPENTUNITY project in general and to the other tasks of WP4 in particular.

The second section, "OPENTUNITY HEMS/BEMS", focuses on the HEMS and BEMS that are used at the four pilots of the project. The section includes a description of the BESOS, Swiss pilot, SMART BOX, and REDUXI platforms, that are providing the necessary data for the development and use of the tools of WP4.

The third section, "Building Flexibility Management System", presents the methodologies and technical developments applied in Tasks 4.1 and 4.2. Here, several subsections provide details regarding the development process of the BFMS, proposed methodologies, technology stack, input/output parameters and development status. The section also describes the assumptions made during the development of the tools, their restrictions, and, lastly, presents preliminary test results.

The fourth section, "NILM and Behaviour Analytics", presents work related to the NILM and behavioral analytics tool. Its structure is similar to the third section by including subsections on prototype development, methodologies, tools, interfaces and initial testing examples.

The fifth section, "Conclusions", summarizes the key findings and achievements presented in the document. This section provides a concise wrap-up of the deliverable's contributions to the OPENTUNITY project and outlines the remaining challenges and future work to be included in the second version of the deliverable.

2. OPENTUNITY HEMS/BEMS

2.1 BESOS

BESOS is a HEMS/BEMS from ETRA, its main functionalities are:

- Optimize the use of energy resources in the environment they operate,
- Optimize the time of use and general operation of the devices,
- Calculate and aggregate the flexibility of the assets and
- Provide predictive maintenance and diagnostics of the devices by means of data analytics

The software is continuously evolving and, up until now, it has been able to connect to different devices:

- Smart Meters,
- PV panels,
- Batteries,
- Electric Vehicle chargers and
- Smart Plugs

The connection to Smart Meters, PV panels and batteries has been done via Modbus TCP, periodically retrieving the status, measurements and control actions bi-directionally to the software platform through MQTT. Other PV panels data retrieval has been done using a Huawei platform API.

For EV chargers depends on the already in use platform for charging strategies. In case the platform can use Open Charge Point Protocol (OCPP) standard, charger is connected to a backend in the cloud deployed by ETRA and the information on transactions and charging parameters are automatically updated. If OCPP is not available, it has been interfaced using Modbus RTU.

Regarding smart plugs, they have been integrated using an instance of openHAB software running on-premises, which provides a RESTful API for retrieving the status and energy parameters.

The Modbus and openHAB data collection have been performed using hardware from ETRA called SMX. This device has some CPU and RAM resources to run Docker containers where instances of openHAB and different SMX configurations and applications are hosted, as the Graphical User Interface. SMX can be configurated to connect to devices via Wi-Fi, Ethernet or Modbus.

The software platform runs in the cloud, providing a web-based application using Meteor, its main benefit is the updates between server and client, providing the end-user with the latest updated values collected. GUI has been developed using React library.

The main dashboard page can be seen in Figure 2, here the monitored buildings or households can be seen and KPIs regarding energy measurements are available.

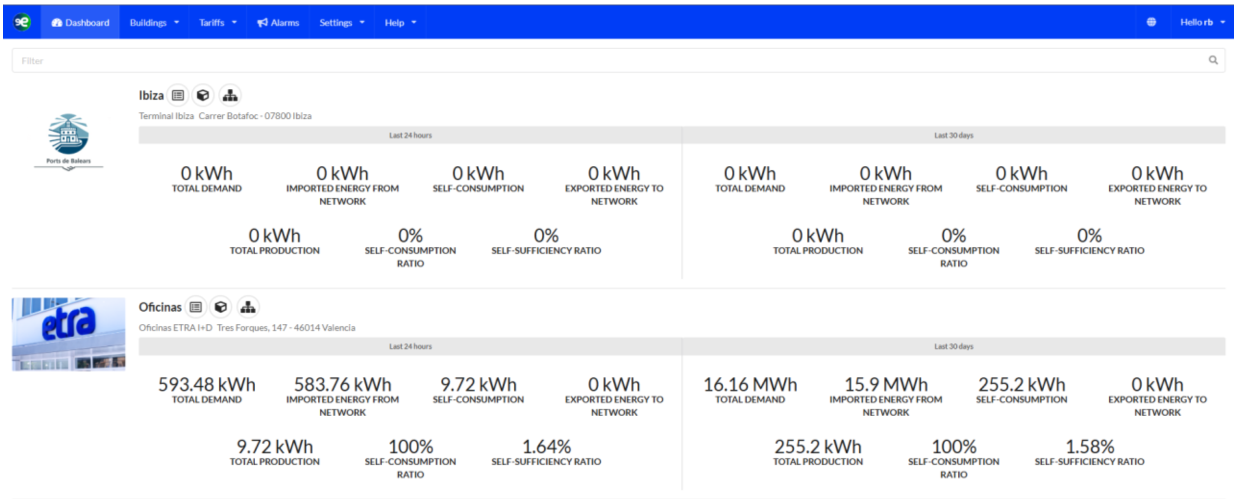


Figure 2: BESOS Graphical User Interface, dashboard page

A button for more detailed information of the building or household can be selected showing building details page, Figure 3.

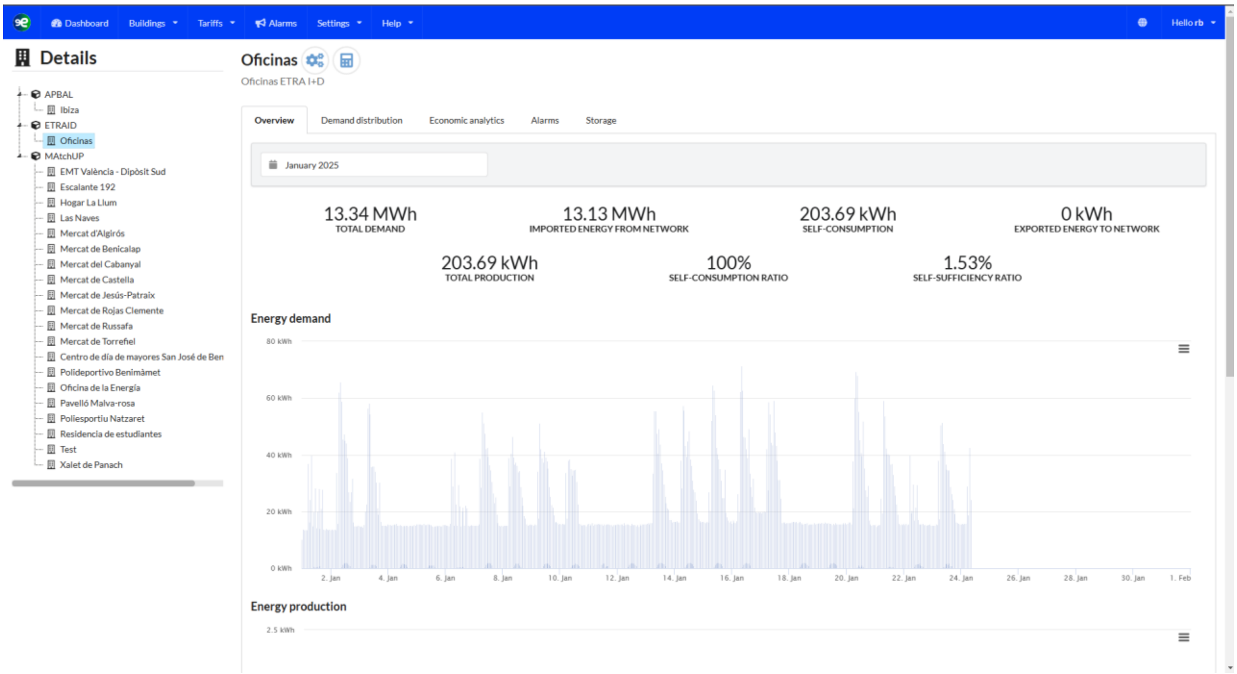


Figure 3: BESOS Graphical User Interface, building details page

Further detailed information about the devices inside the building can be found, see Figure 4.

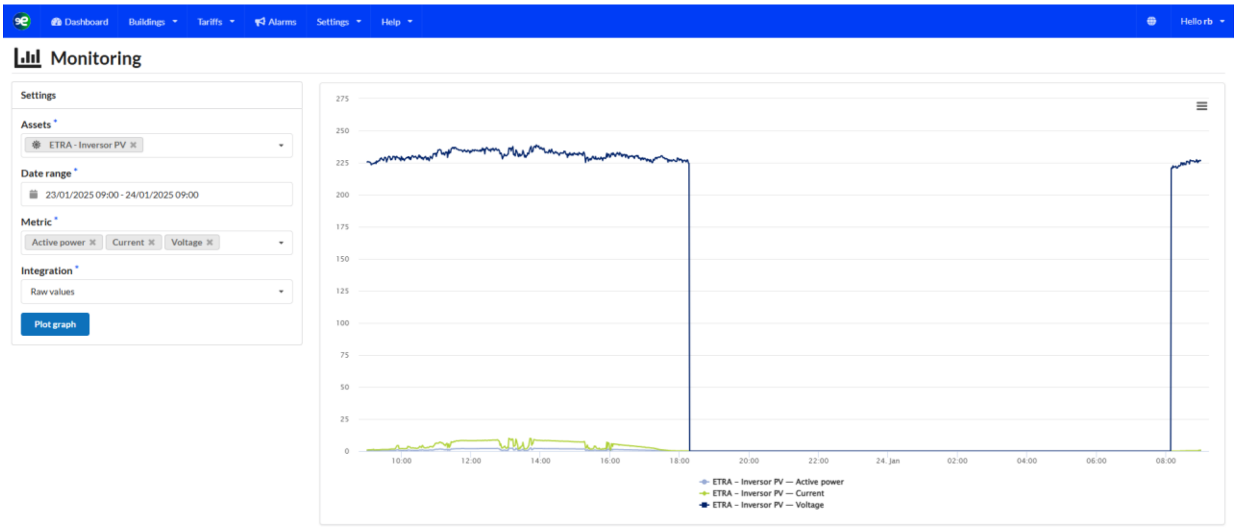


Figure 4: BESOS Graphical User Interface, asset monitoring page

The data managed by BESOS is structured hierarchically in four defined levels:

- **Realm or pilot,** a high-level grouping of different installations.
- **Site,** typically representing a building or location where several devices are installed.
- **Zone,** a sub-division of a site (physical or logical) where a subset of devices is located.
- **Device,** a wide range of elements that provide measurements and other information from the field. Devices can be located inside a zone or directly from the site.

2.2 AEM R&D platform - Flexo

The data from the Swiss pilot site is handled by two different data platforms, the AEM R&D platform and Flexo, both of which are responsible for data collection, storage and pre-processing. Flexo has been developed by Hive Power to manage data from the smart meters, while the AEM R&D platform integrates data from all other sensors installed behind the smart meters at building level and on the LV distribution network, in addition to the control signal forwarding which is currently being tested.

Figure 5 shows the general scheme of AEM's data flows from the field devices to the central servers and the web API used to share data outside the organisation. AEM's R&D platform, which has been operational since January 2024, enables the R&D team to access and manage company data from the distribution grid, energy communities and buildings for research purposes. Hosted in a dedicated virtual machine on AEM's servers, the platform operates independently of the company's main systems, ensuring that no personal customer data or other types of sensitive data are handled. The platform supports data storage, visualisation and secure sharing with authorised third-party users, including OPENTUNITY partners. Internally, AEM is using the platform for asset monitoring and data sharing to third parties in addition to test its capabilities to enable remote control of specific assets for research activities. Data visualisation is based on a dedicated Grafana-powered dashboard, which incorporates an alerting system. Access to the dashboard is restricted to the R&D team. Flexo on the other hand is cloud-based solution based and it is subjected to the same data policy.

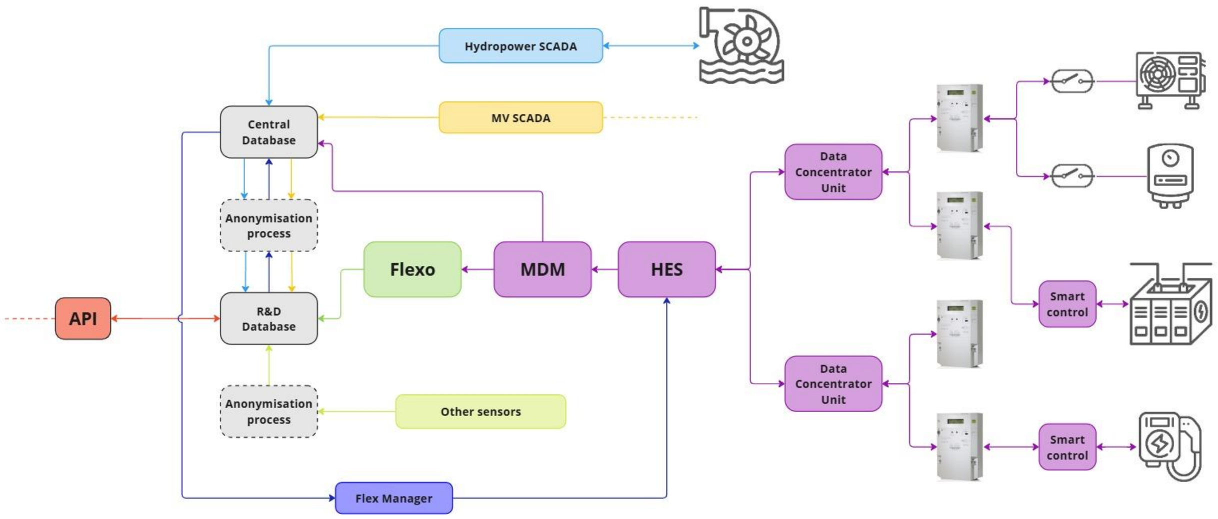


Figure 5: General scheme of AEM's dataflows

The platforms work together for collecting and storing data from all relevant assets and devices installed in the AEM's area of operation, including the energy communities, considering both buildings and the sensors installed in the Low Voltage (LV) grid level. Below is a list of the most relevant sensors:

- Advanced Metering Infrastructure (Landis+Gyr):
    - Type of sensor: smart meter
    - Type of data: net power (W)
    - Data frequency: 15-mins
    - Data updates: 2 hours delays on avg.
    - Asset target: Point of Deliveries (PODs), PV systems, Batteries, EV charging stations

    - Platform: Flexo
- Submeter (Shelly):
    - Type of sensor: energy meter
    - Type of data: Active power (kW), Apparent power (kVA), Voltage (V), Current (A), Power Factor, Frequency (Hz)
    - Data frequency: 1-min
    - Data updates: quasi real time
    - Asset target: PV systems, Batteries, Heat Pumps, El. boilers, AC systems, EV charging stations
    - Platform: AEM R&D platform
- LV grid monitoring devices:
    - Type of sensor: Smart Grid Interface Modul (SGIM)
    - Type of data: Active power (kW), Apparent power (kVA), Voltage (V), Current (A), Power Factor, Frequency (Hz)
    - Data frequency: 1-min
    - Data updates: quasi real time
    - Platform: AEM R&D platform
- Building sensors:
    - Type of sensor: temperature and humidity
    - Type of data: Temperature (°C), Relative humidity (%)
    - Data frequency: 15-mins
    - Data updates: quasi real time
    - Platform: AEM R&D platform

The communication protocols are variable based on the type of device and the specific manufacturer. Compatibility is currently implemented for protocols like MQTT, OCPP, Modbus and HTTPS. Once collected, in both platforms data undergoes a harmonisation process that includes outlier removal.

Despite the two platforms, third-party access is provided using a single web API accessible only through a whitelist of static IPs. Permits are typically limited in time to a specific project duration and a defined list of assets, which may include individual assets or all assets within an energy community. AEM is also working on developing the API to support not only data retrieval but also control actions for eligible devices.

As mentioned above, the platforms guarantee privacy by not storing or processing personal information, as shown in Figure 5 by the blocks of the 'anonymisation process' for the AEM R&D platform, which is instead embedded in the Flexo solution. Sensors are uniquely identified (e.g., ECM46) and all queries and control requests in the near future are filtered to guarantee network stability and user comfort. Comprehensive documentation of the API and list of endpoints, including building metadata, is provided to partners and tailored to their specific needs.

Future developments include the remote control features for suitable partners to test their algorithms or solutions developed during the project. These functionalities, currently being tested internally, will vary depending on the type of asset. In the OPENTUNITY project, the Swiss pilot site will focus on the control of small residential assets (e.g. heat pumps).

2.3 SMART BOX

The HYPERTECH Energy Management System is designed to monitor and control energy consumption and indoor conditions within buildings. It is used at the Greek pilot site of the project. The system involves several actors/components such as Prosumers, who can both produce and consume energy, and Spaces, representing specific building areas. Energy loads include Building Global Load, covering total metering, PV production, HVAC, and Domestic Hot water (DHW), and Space Load, covering space metering, lighting, sensing, HVAC, and other devices. These loads are monitored and controlled through specific and custom-made Monitoring and Control Channels. The system relies on continuous data exchange that is enabled by connected devices, operating through a Wireless Sensor Network (WSN).

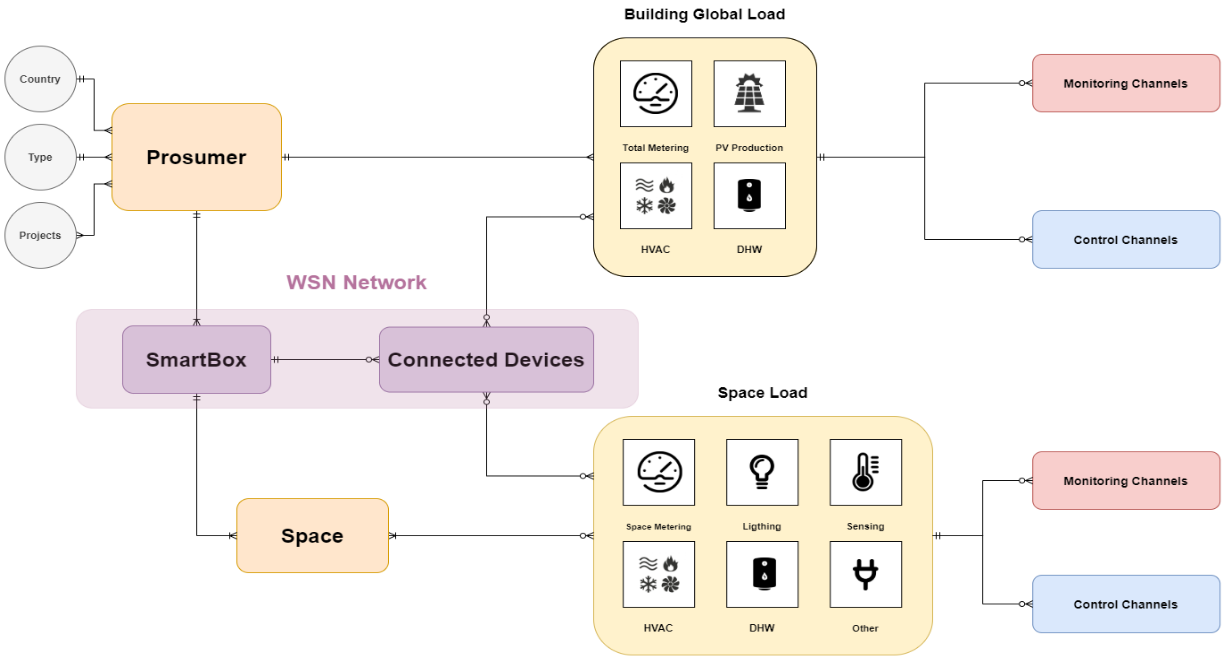


Figure 6: The HYPERTECH Energy Management System

At the center of the system is the **HYPERTECH Smart Box**, developed by HYPERTECH. Its main purpose is to enable seamless, bidirectional communication between a cloud-based system and the various IoT devices installed within the selected dwellings of the project. It serves as a central hub where IoT devices are integrated and managed, therefore enabling wireless and remote monitoring and control of the available flexibility assets. It collects data from various devices, such as indoor conditions, such as temperature, humidity, illuminance, occupancy. In addition, it communicates with controllers and actuators to dispatch control signals, in this way allowing real-time operation following demand response strategies. Moreover, the Smart Box gathers data from metering devices, allowing monitoring energy consumption and appliance performance. The installation of the Smart Box at the pilot sites and its continuous communication with IoT devices transforms the dwellings into smart, energy-responsive buildings, providing necessary data for flexibility forecast and enables the dispatch of flexibility within the OPENTUNITY project.

It is designed under the principles of a Wireless Sensor Network and supports a wide range of IoT devices, including smart meters, sensors and controllers. In this way, real time information on power consumption and indoor ambient conditions from the pilot site is provided to the BFMS. The Smart

Box is responsible for managing the controllable assets, reporting alerts, and facilitating the exchange of information related to demand flexibility. One of its key characteristics is its interoperability, as it is compatible with a wide range of commercially available IoT devices that communicate using standard protocols. In addition, it favors adaptability, modularity, and cost efficiency and it is designed to be easy to install and manage while maintaining a user-friendly aesthetic. The whole system consists of smaller, independent components that can be easily managed, implemented, and maintained. In this way, complexity is reduced, as each module can be designed, tested, and updated separately. Adaptability is also increased, as different modules can be swapped or customized to meet the specific needs of individual users. Modularity simplifies maintenance, making it easier to isolate and address issues, and supports future upgrades or changes without impacting the entire system.

The software stack of the Smart Box consists of multiple layers, including a **Network Layer** for seamless connectivity, a **Protocol Agnostic Layer** for the integrating of various IoT protocols into a unified API, and an **Application Layer** for device discovery, control dispatching, and cloud communication. Moreover, **Security APIs** offer secure data exchange, while **Message Broker** and **REST interfaces** ensure interaction with the cloud infrastructure.

The **Network Layer** is built with a number of key functional specifications, offering reliable and efficient operation. For instance, it operates under low data storage requirements with a minimum capacity of 32GB for local gateways and it prioritizes robustness through regular testing and reconfiguration capabilities. Moreover, it has minimal energy needs, as it features event/threshold-based data exchange (rather than continuous streaming). It can also support large buildings with numerous sensors and actuators, by offering the possibility to use communication extenders or additional gateways. The interoperability feature is ensured via the support of multiple wireless communication protocols, thus, allowing the Smart Box to interface with a variety of sensing, metering and controlling devices, including devices that do not adhere to particular data models. This is particularly interesting for flexibility assets that were not initially intended to have a network interface. The data management model remains extensible while respecting memory constraints of the Smart Box and features short-term data storage to prevent information loss during communication interruptions. The system supports various wireless protocols like Z-Wave, WiFi, and BLE, with the possibility to upgrade to additional protocols such as ZigBee and EnOcean.

The **Protocol Agnostic Lay**er is a middleware component within the Smart Box, as it creates a bridge between various technologies and the smart building automation system. It manages bidirectional communication by collecting data from IoT devices using different protocols and delivering it to the next layer (application layer); it also translates and sends control commands to the flexibility assets. This is implemented through OpenHAB, an open-source automation software. OpenHAB was selected for its Java based functionality, as it has a vendor-neutral design and offers extensive technology support. Additionally, it is highly adaptable and extensible and can therefore integrate different communication protocols while maintaining a unified interface.

The **Application Layer** is one level higher than the protocol agnostic one. It offers several functionalities, such as IoT device management, data security, control dispatching and system operation maintenance. It consists of several sub-modules, the first one being the Commissioning and Configuration app that enables device discovery and registration. Moreover, a user-friendly Smart Box app allows end-users to control their assets and monitor indoor conditions. The IoT

Network Health Check module ensures network integrity by monitoring device status and performing necessary healing processes. Moreover, the Data Management and Backup Module stores data for up to seven-day period, in case there is a cloud connection disruption. The Control Dispatcher Proxy is responsible for the management and execution of control commands. Lastly, the Over-the-air Update Module handles remote firmware updates for core services. The aforementioned modules ensure a reliable system operation while offering remote maintenance capabilities and updates without requiring physical access to pilot sites.

The Smart Box services and applications operate on the Rasberry Pi Operating System, including all necessary drivers. The system's Protocol Agnostic layer is developed under the OpenHAB automation software, using Java extensions to enable communication between the Smart Box and devices at the pilot sites. The Commissioning and Configuration app is developed using Node.js and JavaScript.

The Smart Box uses a Raspberry Pi 3 Model B+ single-board computer, providing a full operating system and development environment (Figure 7). It features a 1.4GHz 64-bit quad-core processor, dual-band wireless LAN, Bluetooth 4.2/BLE, fast Ethernet, and Power-over-Ethernet support. Moreover, a RaZberry 2 GPIO daughter card is used to transform the Raspberry Pi into a Z-Wave Home Automation gateway (Figure 7). RaZberry 2 includes a Z-Wave transceiver, a certified Z-Wave communication stack (Z-Way), and a web-based AJAX user interface, allowing easy control and customization via a browser or mobile device, while offering a communication range of up to 200 meters.

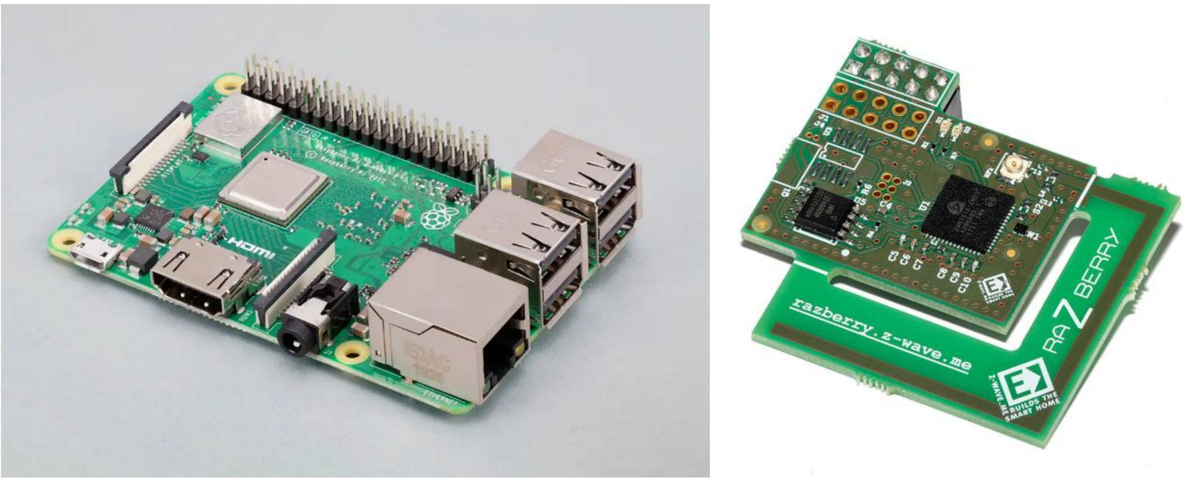


Figure 7: The Raspberry Pi 3 Model B+ (left) and the RaZberry 2 GPIO daughter card (right)

Overall, the Smart Box serves as the central hub that integrates and manages IoT devices deployed in buildings, enabling smart monitoring and automated demand response.

2.4 REDUXI

Reduxi Controller is an innovative energy management system designed to optimize electricity consumption, production, and storage for residential and industrial users. Using different optimization modes, the system provides real-time adjustments to reduce costs, enhance energy efficiency, and increase self-sufficiency, making it an ideal solution for those looking to align their energy practices with sustainability goals. Its intuitive cloud-based platform offers users complete control and insights into their energy systems, driving smarter decisions and long-term savings.

Reduxi Controller is the HEMS. It's the core component of the Reduxi ecosystem, offering robust functionality to optimize energy management and maximize efficiency. Its advanced capabilities include:

- Centralized Energy Management: Seamlessly connects solar panels, energy storage, heat pumps, EV chargers, and more to streamline energy production and consumption.
- Smart Device Integration: Communicates with devices using industry-standard protocols (e.g., Modbus, DLMS, OCPP) for effortless control and compatibility.
- Dynamic Tariff Optimization: Leverages real-time market data to reduce energy costs through automated load shifting and peak management.
- High-Speed Data Processing: Provides second-by-second data acquisition and control for precise and responsive system operations.
- Plug & Play Setup: Simple and user-friendly installation with intuitive configuration tools, enabling fast deployment without technical expertise.
- Scalable Design: Supports an unlimited number of devices for versatile use in residential, commercial, or industrial environments.

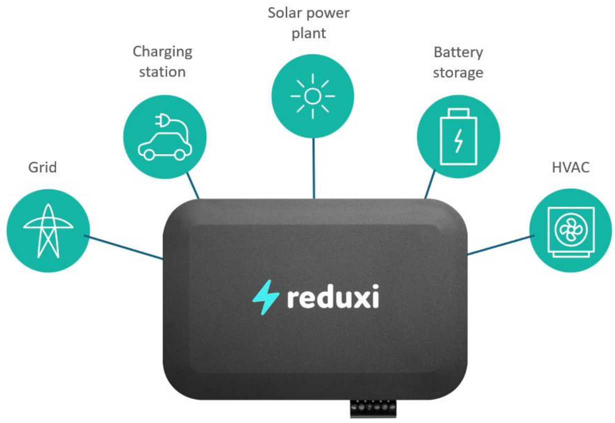


Figure 8: Reduxi Controller connections

Reliable, efficient, and highly adaptable, the Reduxi Controller empowers users to achieve energy self-sufficiency and cost savings while staying connected to the future of energy management. The Reduxi Controller represents the central part of the system, connecting with devices via communication protocols, eliminating the need for additional electricity meters or external units to manage connected devices. The Reduxi system allows the integration of solar power plants, battery storage systems, heat pumps, and electric charging stations. It automatically manages electricity production and consumption while optimizing grid consumption. Using advanced algorithms, it selects the most cost-effective energy usage path and maximizes self-sufficiency.

The Reduxi Controller is designed for easy installation and use, following the "Plug & Play" approach, requiring no technical knowledge. Installation is completed in just three steps:

1. Connect the Reduxi Controller to the electrical grid.
2. Connect it to the internet via wired Ethernet or wireless WiFi.
3. When the green light turns on, the device is correctly connected and installed.

Devices can be connected easily via wired or wireless methods using the Reduxi Configurator with just a few clicks.

The Reduxi Controller is designed for easy installation and use, following the "Plug & Play" approach, requiring no technical knowledge. Installation is completed in just three steps:

4. Connect the Reduxi Controller to the electrical grid.
5. Connect it to the internet via wired Ethernet or wireless WiFi.
6. When the green light turns on, the device is correctly connected and installed.

Devices can be connected easily via wired or wireless methods using the Reduxi Configurator with just a few clicks.

Supported Connections

- **Wired Connections:**
    - LAN (TCP/IP)
    - 2x RS485
- **Wireless Connections:**
    - WiFi (TCP/IP)

Supported Communication Protocols:

- Modbus RTU
- Modbus TCP/IP
- DLMS
- DLMS-push
- M-bus (EN 13757-2)
- OCPP 1.6J
- REST API
- SunSpec
- MQTT

A full list of supported devices and protocols is available on https://support.reduxi.eu.

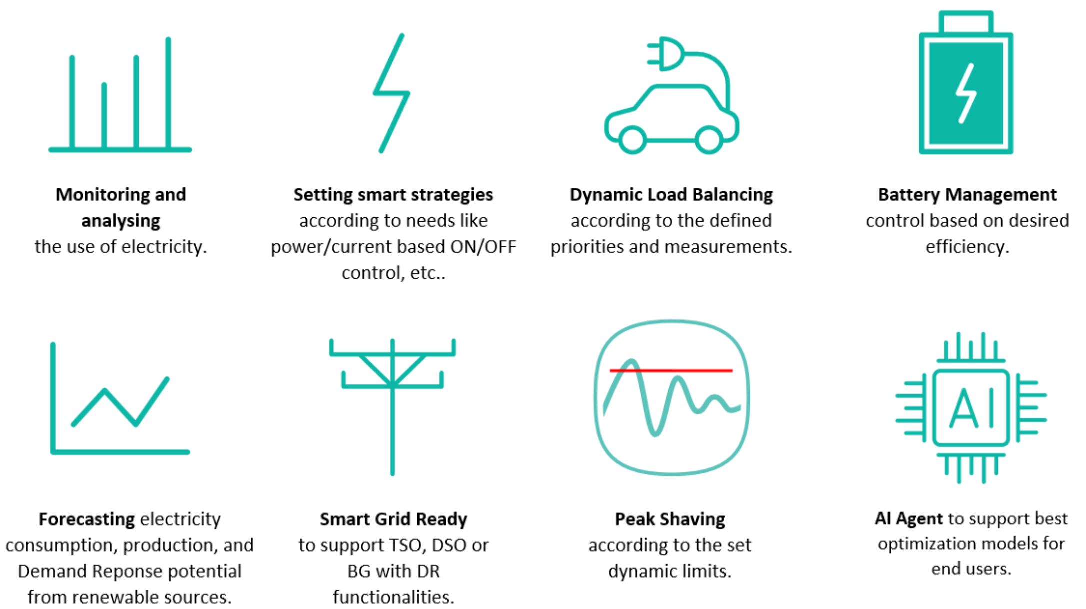
The Reduxi Controller can operate independently without the Reduxi Cloud solution or be connected to the Reduxi Cloud for remote access and management anytime, anywhere. Data is transmitted to the Reduxi Cloud in real-time at a second-based interval and periodically every minute. The Reduxi Cloud is accessible via a web browser on desktops and mobile devices and is free for end users. With advanced encryption, the Reduxi Cloud ensures high security for both data and connected electrical devices, complying with GDPR regulations.

The Reduxi Gateway expands the Reduxi Controller system by converting RS485 serial signals into internet-based TCP/IP communication. It is useful when additional serial inputs are needed or when the connection distance is too long for a wired connection. The Reduxi Gateway also enables third-party device control via two relay outputs, such as controlling a heat pump using SmartGrid Ready (SG Ready) or EVU contacts.

Each Reduxi Controller comes pre-installed with the Reduxi Configurator, accessible via a local network browser or the Reduxi Cloud. Users can easily add devices and configure strategies through drop-down menus. Using the MQTT protocol, data can also be accessed directly through a private or public MQTT broker. The MQTT API is fully open and publicly available at https://mqtt.support.reduxi.eu.

The Reduxi system supports real-time energy meter data collection trough P1 port of smart meter. Users can connect energy meters and obtain billing data every second without additional power supply requirements. The Reduxi Controller maximizes energy self-sufficiency by prioritizing self-generated solar energy and battery storage over grid consumption. It adjusts energy use based on market prices, ensuring cost-effective operation by charging storage systems and EVs during low-cost periods and discharging during high-cost periods.

All functionalities and strategies are executed locally, meaning that even if the internet connection is lost, the Reduxi Controller continues to operate effectively. The Reduxi Controller offers public documentation for MQTT API integration, supporting both public and private MQTT brokers (AWS, Azure, Google Cloud, etc.). Users can customize data transmission frequency and use publish-subscribe methods for real-time device control.



3. Building Flexibility Management System

3.1 BFMS Overview

This section provides a detailed presentation of the Building Flexibility Management System - BFMS component. This component uses energy consumption data and potentially available building sensorial data along with consumer feedback to:

- initialize the DR process, considering both the available flexibility assets and the constraints/preferences of the consumers,
- forecast the day-ahead baseline (business as usual) energy consumption power profile,
- forecast the available flexibility,
- dispatch control commands to the controllable devices (based on the received DR signals), and
- evaluate the results of the flexibility dispatched to the consumers' premises.

To achieve this, BFMS interacts with several other tools/services of the OPENTUNITY ecosystem. Figure 9 presents the flexibility extraction and biding workflow, at the center of which lies the BFMS component.

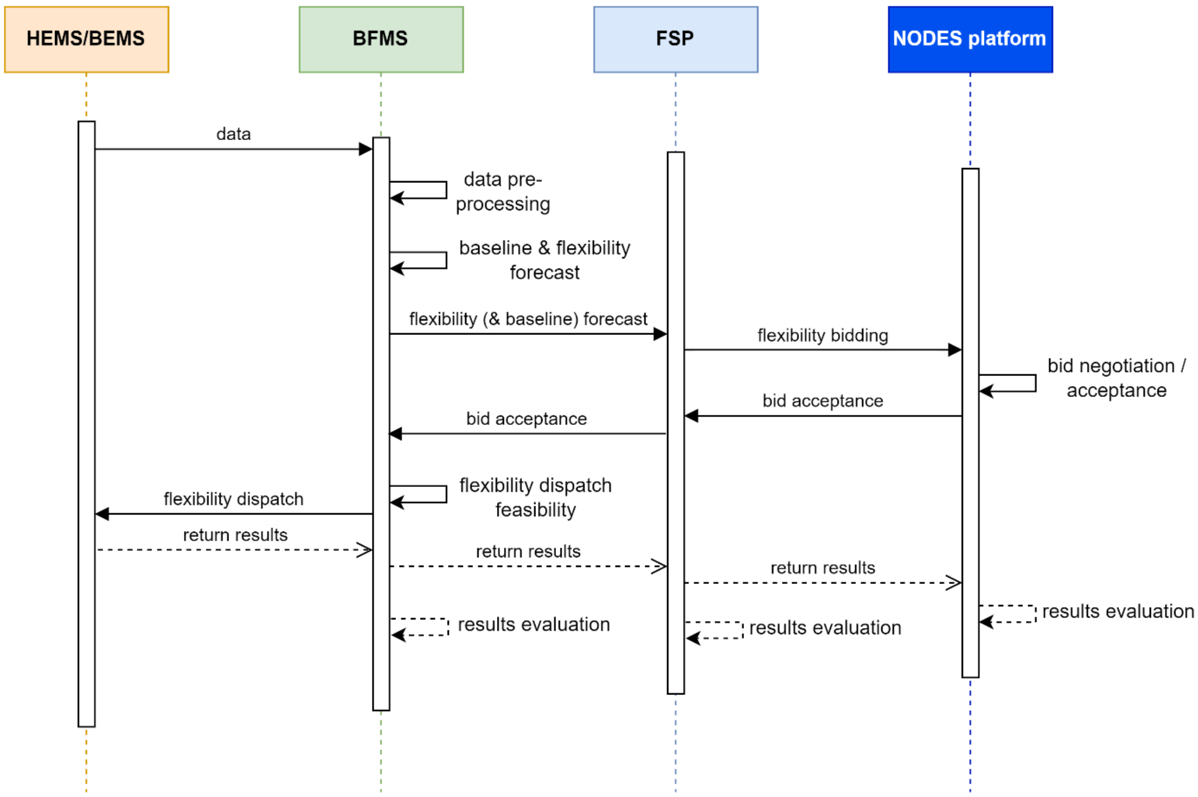


Figure 9: Flexibility management and biding workflow

The four tools involved in this interaction are:

- the HEMS/BEMS at the pilot sites,
- the Building Flexibility Management System,

- the Flexibility Service Provider, and
- the NODES platform.

The process begins with the BFMS receiving energy consumption data from the HEMS/BEMS in the form of power time series. Upon availability, HEMS/BEMS also provide sensorial data (temperature, occupancy, etc.) and/or occupant defined preferences/constraints. BFMS consists of a number of modules (as explained below), that will pre-process this data, proceed with comfort, building and asset profiling and ultimately calculate the baseline and flexibility power consumption. The forecasted baseline and flexibility values are then sent to the FSP, which performs an evaluation of the available aggregate flexibility from the available assets; it will then handle flexibility bidding to the NODES platform. The NODES platform, also considering the Systems Operators' requirements, negotiates and accepts bids, sending bid acceptance information back to the FSP. The FSP then communicates this information back to the BFMS, which will in turn assess the feasibility of flexibility dispatch based on real time information from the assets. Appropriate control command sequences are then generated and sent to HEMS/BEMS. The results of the flexibility dispatch actions are finally returned to the BFMS, the FSP and the NODES platform for evaluation. Throughout this last step, results are evaluated at multiple stages to ensure accuracy in flexibility validation.

The BFMS development is based on the requirements of two Demand Response Use Cases of the project, i.e., Use Case 1.8 "HEMS/BEMS DR optimization and local flexibility management" and Use Case 1.9 "Initialization of HEMS/BEMS Demand Response strategy". A detailed description of the Use Cases of OPENTUNITY can be found in Deliverable 2.3 "Open architecture report".

Figure 10 provides the component diagram of the BFMS. The core of the system consists of the DR Initialization Service, the Profiling Modules, the Flexibility Forecasting Module, and the Control Dispatching Service. These modules and services are described in detail in the following subsections. Several other modules/services ensure the orchestration of the different procedures, as well as data processing and storage.

The DR Initialization Service serves as the entry point, where occupants interact with the system through a front-end UI module. A back-end module manages occupants' assets and preferences, ensuring that flexibility settings align with user-defined preferences and constraints.

Within the Profiling & Flexibility Forecasting Orchestration Service, several modules collaborate to handle scheduling, data ingestion, and execution planning. Profiling modules include Comfort Profiling, Building Thermal Modeling and Asset Profiling Modules, all of which serve as inputs to the flexibility forecasting module. This module is responsible for the forecast of day-ahead baseline and flexibility power consumption timeseries.

For Control Dispatching, a control design module is used, alongside a dispatcher module, ensuring that optimized flexibility strategies are executed efficiently.

The Profiling Storage & Retrieval Service maintains and retrieves profiling and forecasted flexibility results through a persistence service and an external API, allowing data access and integration with external tools and services of the project.

The figure also illustrates the interactions between HEMS/BEMS and BFMS, where building and asset data from the pilot sites, along with control dispatch instructions are exchanged. Furthermore, information on the extracted flexibility and demand response control signals are exchanged with the FSP (Flexibility Service Provider), ensuring smooth coordination within the energy market.

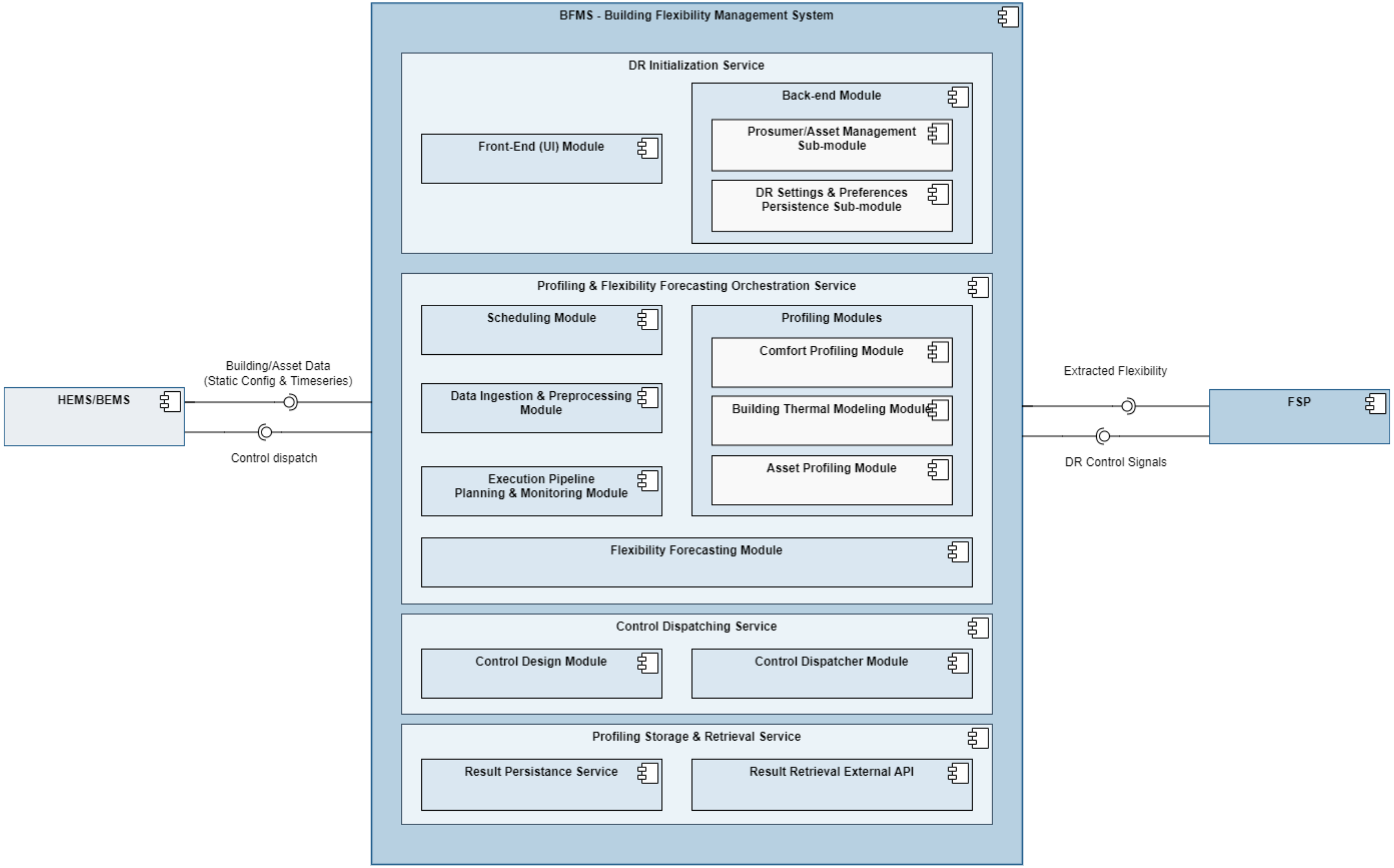


Figure 10: BFMS Component Diagram

3.2 BFMS modules

This section presents the methodologies that have been employed for the core modules/services of the BFMS component. It is divided into the following subsections:

- DR Initialization Service;
- Profiling Modules:
    - Comfort Profiling Module;
    - Building Thermal Modeling Module;
    - Asset Profiling Modules;
- Flexibility Forecasting Module; and
- Control Dispatching Service.

The main innovation presented in this deliverable is the Flexibility Forecasting Module, that implements an optimization problem which is presented in detail in Section 3.2.5. For the sake of completeness, modules that have been developed by HYPERTECH in past EU funded projects are briefly presented (H2020 ACCEPT project [1]); these modules are/could be adjusted for use within the OPENTUNITY context.

In previous versions of the Flexibility Forecasting Module, the aforementioned optimization problem was based on the outputs of the Comfort Profiling, Building Thermal Modeling and Asset Profiling Modules. However, this approach required extensive sensorial data, leading to an expensive, complex, and intrusive solution. Additionally, unless the sensorial and metering devices were properly installed and maintained, the previous approach could lead to biased data and consequently inaccurate model training. Therefore, the Flexibility Forecasting Module has been refactored to address these drawbacks by introducing a new optimization problem. This problem is formulated using only assets' power profiling data and constraints imposed using analysis of this data (i.e., inferred constraints) or occupants' input via the DR Initialization Service (i.e., explicit constraints).

Regarding the Control Dispatching Service, an overview, along with a rule-based control approach that has been explored for HYPERTECH's Smart Box is provided in section 3.2.6.

3.2.1 DR Initialization Service

The DR Initialization Service Module aims to deliver a user interface (UI) to facilitate the participation of occupants in flexibility activities. It provides a user-friendly way for them to provide input on the assets that will participate in the DR activities, as well as on individualized preferences or even constraints on the control of these assets. For instance, the occupants can define operational constraints, such as temperature set points for HVAC systems during the summer or winter period or a timeframe in which hot water must always be available (Figure 11). In this way, the BFMS ensures that the flexibility actions align with the comfort preferences and daily routines of the occupants, without disrupting their behavior.

This component is not strictly necessary for the end-to-end execution of OPENTUNITY's Demand Response (DR) schemes. However, its integration offers significant added value by incorporating end-users' preferences, ensuring a more tailored and user-centric approach to flexibility management. This feature alleviates the need for constant manual adjustments, allowing occupants to participate in DR campaigns with confidence, knowing that their preferences will be respected.

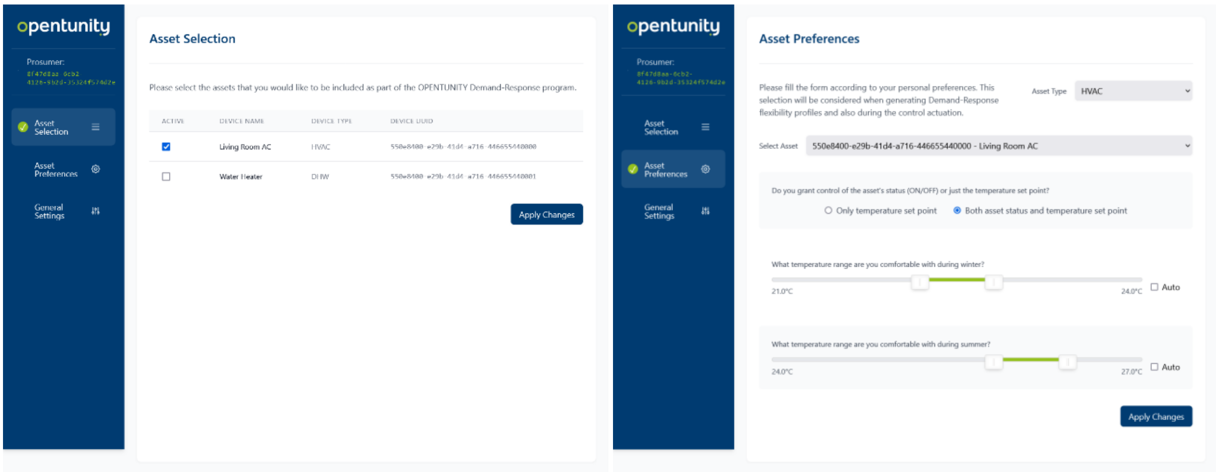


Figure 11: DR Initialization Service UI

3.2.2 Comfort Profiling Module

The Comfort Profiling Module builds on previous European projects like ACCEPT, combining the Random Forest (RF) algorithm with Symbolic Aggregate Approximation (SAX). RF is a commonly used machine learning algorithm that combines the output of multiple decision trees to reach a single result. Its ease of use and flexibility have fueled its adoption, as it handles both classification and regression problems. SAX is another widely used technique for data compression and noise reduction in timeseries data. It simplifies complex datasets by converting timeseries chunks into strings of symbols, making pattern recognition more efficient.

Here, SAX is used to categorize comfort states, while RF is applied for training and predicting them. The integration of these two techniques enhances the ability of the Comfort Profiling Module to analyze and predict user comfort states, increasing the accuracy and responsiveness of the BFMS. As previously mentioned, the Flexibility Forecasting Module can use the outputs of the Comfort Profiling Module for the calculation of the baseline and flexibility energy consumption. The Comfort Profiling Module works in tandem with the DR Initialization Service, together offering a human-centric solution that capture the preferences of the occupants.

3.2.3 Building Thermal Modeling Module

Building upon the insights from the H2020 ACCEPT project [2], the thermal behavior of buildings in the OPENTUNITY project relies on a data-driven, grey-box modeling approach. It uses a second-order lumped element Resistance-Capacitance (RC) network, leveraging a state-space formulation to model the thermal dynamics of each building space based on measurable inputs such as outdoor temperature, solar irradiance, HVAC loads, and internal heat gains. In this way, prior knowledge of the characteristics of the building is not required.

Once parameters are identified, space air temperature predictions are obtained using an analytical solution of the state-space model. This method ensures a balance between accuracy and computational efficiency, making it a viable alternative to conventional white-box thermal modeling methods.

3.2.4 Asset Profiling Modules

This section describes the data-driven methods adopted in OPENTUNITY for modeling HVAC systems (AC split units with inverters, air-to-air constant volume heat pumps and electric heaters, air-to-water and water-to-water heat pumps) and Electric Water Heaters (EWHs) based on ACCEPT D4.8 Building Digital Twin v2 [2]. These models focus on identified types of devices present at pilot sites, ensuring relevance and applicability to real-world scenarios.

For each case, IoT data undergo preprocessing, including outlier detection and removal. This ensures clean and reliable data for model development without requiring further segregation for performance assessment.

3.2.4.1 AC Split Units with Inverters

Due to their dynamic behavior, which is influenced by various external parameters such as weather conditions and thermostat settings, advanced machine learning algorithms are used. Specifically, Gaussian Processes (GP) were selected for their non-parametric nature, enabling flexible and scalable modeling as the dataset grows. Unlike parametric methods, which require manual configuration, GP models adapt automatically to varying system specifications.

The GP-based model forecasts power consumption at each time step, providing a range of possible values (mean, minimum, maximum) based on input parameters. The model can be further extended to support more complex HVAC configurations as needed.

3.2.4.2 Air-to-Air Constant Volume Heat Pumps and Electric Heaters

Building on methodologies explored in prior research, two clustering-based, data-driven approaches have been identified for modeling air-to-air constant volume heat pumps: DBSCAN [3] and k-means [4]. While DBSCAN can produce clusters with lower variability, it has higher computational demands and requires additional preprocessing steps for hyperparameter definition, making it less practical for this context.

The k-means algorithm, widely recognized for its simplicity and efficiency, was selected as the preferred approach. It is an unsupervised machine-learning method that groups data points into clusters based on their similarity, measured using Euclidean distance. The algorithm requires specifying the number of clusters (k), which is determined automatically using methods like the Elbow and Silhouette coefficient analysis [5]. Once initialized, k-means iteratively assigns data points to clusters, recalculates cluster centroids, and converges when no significant changes occur. The enhanced version, k-means++ [4], improves performance by optimizing the selection of initial centroids.

The goal of k-means in this application is to cluster the power consumption patterns of HVAC devices based on available data. Key inputs include the device's power consumption (timeseries data), operational status (on/off), and user-set temperature setpoints. These parameters enable the extraction of clusters that represent different levels of power consumption.

3.2.4.3 Air-to-Water and Water-to-Water Heat Pumps

For the profiling of air-to-water and water-to-water heat pumps, a machine learning method is used [2]. The forecasting process consists of multiple steps: data pre-processing, feature extraction, algorithm selection, fine-tuning model parameters and validation using performance metrics.

To improve accuracy, the Extreme Gradient Boosting machine learning technique is used, building multiple decision trees iteratively. Hyperparameter tuning is conducted using the Optuna framework to enhance model performance. Additionally, model stacking is used as an ensemble method, combining multiple models to improve forecasting accuracy.

This approach provides accurate predictions to support efficient operation and management of air-to-water and water-to-water heat pumps.

3.2.4.4 Electric Water Heaters (EWHs) Model

EWH modeling focuses on non-intrusive solutions using power consumption and on/off status data to recognize patterns in energy use [2]. The process begins with identifying different Power Consumption Levels (PCLs). Among them, four distinct levels are established, but the lowest one—representing standby or off mode—is set aside at this stage. The remaining data is then analyzed to create a matrix that tracks how frequently each power level appears at different times throughout the day.

Once this occurrence matrix a is built, the next step is pinpointing the key time intervals when each power level is most likely to occur. By focusing on the moments when each level appears most frequently, a clearer picture of the heater's daily usage pattern emerges. Finally, this information is used to generate a forecast of power consumption for the following day.

This method provides a practical way to predict how EWHs will behave without requiring complex sensors or intrusive data collection, making it well-suited for IoT applications and flexibility forecasting purposes.

3.2.5 Flexibility Forecasting Module

This section presents an innovative approach for flexibility forecasting, which represents a key advancement in the project's methodology. The development of the flexibility forecasting module primarily relates to T4.1, with overlaps in T4.2, where specific optimization constraints are identified.

User preferences and requirements play an integral role in shaping these constraints, with input collected directly from users via forms containing targeted questions.

The flexibility model also relies on a baseline energy consumption profile, created using historical user data. This baseline is established through machine learning algorithms (mentioned in the previous subsection) trained on these datasets. Further details about the training process and the algorithms employed will be provided in greater detail in D4.2. In the following subsections, the specifics of the proposed flexibility model are explained, detailing its innovative optimization approach and its integration into the broader energy management framework.

This service derives flexibility (power) that can be offered by a device or prosumer (compared to baseline) at each time step, e.g., an hour. The flexibility forecast is the output of an optimization problem, which takes as input the baseline profile, as well as the reward for the service provision, and others.

Briefly, the method is based on baseline plus time shifting of load, which has the following advantage: a baseline is calculated using historical data, and time shifting can be applied to each activation of the power profile, which renders the method more realistic, while also it does not require extensive sensor data (compared to the previously applied approach).

In terms of optimization class, the proposed model relies on a mixed-integer linear programming (MILP) problem. The MILP method is chosen to ensure global optimality and computational efficiency [6]. The decision variables, objective function, and constraints of the model are shown in the following subsections.

3.2.5.1Decision Variables

Firstly, a separate demand variable $dx_{i,j,t}$ for each asset i (ϵ 1,…,N) is defined, for each activation j (ϵ 1,…,$\alpha[i]$), for each time step t (ϵ 1,…,T), where N is the number of demand assets, $\alpha[i]$ is the number of activations of asset i, and T is the number of time steps. Secondly, $y_{i,j,h}$ binary variables are defined, which are used to choose one of some possible time intervals during which an activation can take place. A y variable for each asset i, for each activation j is defined, and each possible time interval h (ϵ 1,…,F). Finally, d_t, is defined, representing the total demand at each time step. g_t is the PV generation at each time step. Note that g_t is a parameter (not a decision variable) in this model.

3.2.5.2 Constraints

This subsection presents the constraints of the model. Firstly, the variable bounds are presented, followed by the rest of the model constraints.

$$dx_{i,j,t} \leq \text{mean\_power}_{i,j} \tag{1}$$

$$y_{i,j,h} \in \{0, 1\} \tag{2}$$

$$d_t \geq 0 \tag{3}$$

where mean_power$_{i,j}$ is the mean power during activation *j* for asset *i*. It should be noted that given the baseline profile for an asset, the demand over each activation period is averaged at first, and then this averaged demand in the optimization problem is considered. Equation (2) denotes that $y_{i,j,h}$ variables are binary, which means that they can either take the value of one or zero. Equation (3) ensures variable d_t is nonnegative.

Constraint (4) enforces $dx_{i,j,t}$ to be zero outside of all possible activation time intervals.

$$dx_{i,j,t} = 0,\ t \notin \{t\_s_{i,j} - F + 1, t\_e_{i,j}\} \tag{4}$$

where $t\_s_{i,j}$ and $t\_e_{i,j}$ are the start and end times for asset *i* and activation *j*, respectively.

Constraint (5) ensures that the optimized consumption profile has the same amount of energy as the original, for each activation *j* for each asset *i*.

$$\sum_t dx_{i,j,t} = \mathrm{dur}_{i,j} \cdot \mathrm{mean\_power}_{i,j} \tag{5}$$

where dur$_{i,j}$ is the duration of activation *j* for asset *i* and is equal to $t\_e_{i,j}$ - $t\_s_{i,j}$ + 1.

Constraint (6) provides the relationship between the total demand d_t at each time step and the separate demand variables $dx_{i,j,t}$ for each activation for each asset.

$$\sum_i \sum_j dx_{i,j,t} = d_t \tag{6}$$

Constraints (7) and (8) ensure that one of the *F* possible time intervals is chosen for each activation for each asset. Constraint (7) can be informed from user preferences that will be captured through T4.2.

$$\sum_{t=t\_s_{i,j}-F+1+h}^{t=t\_s_{i,j}-F+1+h+\mathrm{dur}_{i,j}} dx_{i,j,t} \geq \mathrm{dur}_{i,j} \cdot \mathrm{mean\_power}_{i,j} \cdot y_{i,j,h} \tag{7}$$

$$\sum_{h=1}^{h=F} y_{i,j,h} = 1 \tag{8}$$

Finally, a given amount of time between consecutive activations is ensured. Specifically, this time (as a default option) is set to be one hour for HVAC and three hours for DHW (domestic hot water DHW and electric water heater EWH terms are used interchangeably). This is performed by enforcing specific y variables to be zero, while also ensuring that at least one activation is possible.

In the case of energy storage systems, the following constraints can be added [7]:

Equation (9) relates the state of charge (SOC) of the battery between consecutive time steps with the charging/discharging power. Equation (10) constrains SOC between zero and its maximum value. Equations (11) and (12) enforce charging and discharging power limits, as well as prevent simultaneous charging and discharging through the use of binary variable α.

$$SOC_{t+1} = SOC_t + P_{ch,t} \cdot \eta - P_{dch,t}/\eta \tag{9}$$

$$0 \leq SOC_t \leq SOC_{\max} \tag{10}$$

$$0 \leq P_{ch,t} \leq \alpha_t \cdot P_{\max} \tag{11}$$

$$0 \leq P_{dch,t} \leq (1 - \alpha_t) \cdot P_{\max} \tag{12}$$

where SOC_t is the state of charge of the battery at time *t*, $P_{ch,t}$ is the charging power of the battery at time *t*, $P_{dch,t}$ is the discharging power of the battery at time *t*, η is the efficiency of the battery, $P_{\max}$ is the power limit of the battery, and α_t is a binary variable which guarantees that charging and discharging do not occur at the same time. If α_t = 0, then (11) enforces $P_{ch,t}$ to be zero, and $P_{dch,t}$ is allowed to vary between zero and its power limit by (12). If α_t = 1, then (12) enforces $P_{dch,t}$ to be zero, and $P_{ch,t}$ is allowed to vary between zero and its power limit by (11).

3.2.5.3 Objective Function

The objective function is:

$$\max \ Reward - Cost \tag{13}$$

where

$$Reward = \pi \cdot P_F \cdot \Delta T_{FW} \tag{14}$$

and

$$Cost = \sum_t P_{sch,t} \cdot p_t \tag{15}$$

The reward is constructed using the flexibility compensation price π, the power that can be offered P_F for the required duration, and ΔT_{FW} is the duration of the flexibility window. The cost is formulated using $P_{sch,t}$, which is the optimized power exchange of a prosumer with the grid at time t, which delivers the required reduction/increase in demand at the required time period, and p_t is the electricity price at time t. The constraints of the problem include either:

$$P_F \leq P_{bs,t} - P_{sch,t} \tag{16}$$

which indicates demand reduction, or:

$$P_F \leq P_{sch,t} - P_{bs,t} \tag{17}$$

for demand increase. $P_{bs,t}$ is the baseline power profile of the prosumer.

3.2.6Control Dispatching Service

The Control Dispatching Service evaluates incoming DR signals alongside the outputs of the flexibility forecasting optimization processes. It then generates and dispatches appropriate control sequencies based on the capabilities of building assets and the available IoT controllers.

For the moment, the module is configured to manage HVAC systems and EWHs, as these are energy-intensive, shiftable loads well-suited for demand-side management and flexibility extraction. The system can also be extended to manage other building loads if needed.

The module is composed of two modules:

- **Control Design module**: Registers, monitors, and manages all controllable devices and IoT controllers deployed on-site. It handles DR requests and load dispatching from external stakeholders (e.g., aggregators) or other project components. It also generates appropriate control sequencies based on device types and their control options, ensuring a response to flexibility or load-shifting requests.
- **Control Dispatcher module**: Dispatches the generated control commands to IoT controllers to actuate the building assets timely and effectively.

The Control Dispatching Service is designed to automatically adjust HVAC and EWH operations in real-time, based on data from on-site monitoring systems, flexibility forecasts, and user comfort preferences. The system prioritizes non-intrusive operation, ensuring comfort levels are preserved while executing flexibility or load-shifting scenarios.

During operation, the module integrates direct flexibility dispatch requests with flexibility forecasts (e.g., power consumption curves). For example, HVAC operation may be adjusted by modifying thermostat setpoints while making sure that comfort is maintained. EWHs may be switched on or off based on flexibility needs. The Service takes into account the current status of an asset. For instance, If the device is already in the requested mode/status, a command is not dispatched (e.g., if asked status is ON and the device is currently ON, no command will be dispatched). Additionally, if a DR signal violates comfort boundaries, it will not be dispatched (e.g., only change the temperature set point for HVAC but not turn ON/OFF nor change operation mode). Also, in the case of EWH, as per the pilot users request, a pre-heating sequence might taking place to ensure minimal intrusion to their daily routine.

Furthermore, the system offers the possibility of a manual override by the user (e.g., setting the controlling relay to manual ON or OFF or adjusting the HVAC setpoint), in this way allowing the occupants to increase their comfort level if needed.

Once the DR event concludes, subsequent control actions return devices to their baseline operation.

Depending on the HEMS/BEMS configuration, the system supports bi-directional communication with the building's infrastructure, enabling real-time control and continuous monitoring of asset performance. End-users retain the ability to override any system-generated control.

Additionally, control actions are logged for later meta-analysis and improvement of algorithms where necessary (e.g., to update control boundaries to better reflect user preferences).

The Control Dispatching Service is the part of the BFMS that interacts with the different HEMS/BEMS of the project. Upon acceptance of a flexibility bid in the NODES platform, this service will evaluate the feasibility of forecasted flexibility actions and will send control signals to the selected pilot assets.

This subcomponent will be presented in greater detail in the second version of this deliverable.

3.3 Technology stack and Implementation tools

The BFMS component utilizes state-of-the-art tools and technologies to deliver a modular, efficient, highly configurable and replicable solution; most of the technologies used are free and/or open source, A concise summary of the individual libraries and technologies used is presented in Table 1 below.

Table 1: Libraries and Technologies used in the BFMS

Library/Technology Name	Version	License
Python	3.12	Free, Open Source PSF License (Python Software Foundation License)

Python main libraries used: numpy, scipy pandas, scikit-learn, prefect, docker, pydantic, pika, apscheduler, orjson, httpx, python-cron	(latest versions at the time of writing)	Free, Open Source Various open-source licenses (e.g., BSD, MIT, Apache 2.0)
Java	21 LTS	Free, Open Source GPL v2 with Classpath Exception
Java main libraries used: spring MVC, hibernate	(latest versions at the time of writing)	Free, Open Source Various open-source licenses (e.g., Apache 2.0, LGPL v2.1)
PostgreSQL	17.x	Free, Open Source PostgreSQL License (Permissive, similar to BSD)
Redis	7.4	BSD 3-Clause License
Docker (with docker compose)	28.0, 2.33	Apache License 2.0
Nginx	1.27	BSD 2-Clause License
Tomcat	8.5	Apache License 2.0

3.4 Input/Output parameters

The following table outlines the input and output requirements for the successful operation of the BFMS component.

Table 2: Input and output parameters of the BFMS component

Comfort profiling	
Input parameters	
Space air temperature	Indoor temperature collected from building sensors (°C)
Outdoor air temperature	Outdoor temperature sourced from weather data (°C)
Occupancy data	Optional data collected from the occupancy profiling component
Output parameters	
Baseline thermal comfort boundaries for each space	Occupants' comfort zone (lower and upper boundaries) during occupied periods
Flexibility thermal comfort boundaries for each space	Extended comfort zone (lower and upper boundaries) enabling flexibility during occupied periods

Building thermal modelling	
Input parameters	
T_i	Indoor temperature in ˚C (timeseries, received from OPENTUNITY HEMS/BEMS)
T_{ext}	Outdoor temperature in ˚C (timeseries)
Q_s	Global horizontal irradiance in W/m² (timeseries)
Q_g	Power consumption of other building electric loads
$Q_{h1, \ldots n}$	HVAC power consumption (n = number of HVAC systems)
Output parameters	
Trained RC model for each space	Matrices ***A*** and ***B*** estimated during system identification
AC Split Units with Inverters	
Input parameters	
x_1	Boolean for device status (0 = off)
x_2	Thermostat setpoint temperature in ˚C (timeseries)
x_3	Hourly outdoor temperature in ˚C (timeseries)
x_4	Hourly global horizontal irradiance in W/m² (calculated from cloud coverage and weather data)
x_5	Hourly outdoor relative humidity in % (timeseries)
x_6	Indoor temperature in ˚C (timeseries)
f(x)	Device power consumption in W (timeseries)
Output parameters	
GP(x)	Trained GP model for forecasting HVAC energy consumption
Air-to-Air Constant Volume Heat Pumps and Electric Heaters	
Input parameters	
k	Number of clusters, determined during preprocessing
Power consumption (x_1)	Time-series data from metering devices
Operational status (x_2)	Boolean indicator (0 for off, 1 for on) from monitoring devices
Output parameters	

Power clusters	Different levels of device power consumption, used for optimized control
Air-to-Water and Water-to-Water Heat Pumps	
Input parameters	
N_k	Cluster number assigned by k-means for the timestamp
T_m	Minute of the hour (e.g., 0, 15, 30, 45)
TH	Hour of the day [0, 23]
T_D	Day of the week [0, 6]
Frequent kWs	Aggregated consumption data by minute, hour, and day
External temperature	Outdoor temperature from local weather stations
Seasonality	Periodic time-series feature from seasonal decomposition
Output parameters	
Forecasted Power Profile (P_1, P_2, … P_n)	Day-ahead predicted power consumption (time series) for the heat pumps
EWH Model	
Input parameters	
k	Number of clusters for k-means clustering
x_1	Device power consumption in W (timeseries)
x_2	Boolean for device status (0 = off)
Output parameters	
$P_{1,2,\ldots n}$	Forecasted EWH power profile for n timesteps in W (timeseries)
Flexibility Forecasting Service	
Input	
Baseline Profiles	Baseline power day-ahead profile in W (timeseries)
Consumer Preferences (from DR Initialization)	User defined comfort preferences
Output	
Flexibility Forecasted Profile	Upward and downward power day-ahead profile in W (timeseries)

3.5 Interfaces & API documentation

Documentation is available through Swagger as illustrated in Figure 12, providing a clear and structured overview of the system's functionalities. The primary outputs include baseline forecasts and flexibility predictions, ensuring accurate energy management insights. For reference, an example payload schema is illustrated in Figure 13, demonstrating the data structure and expected format.

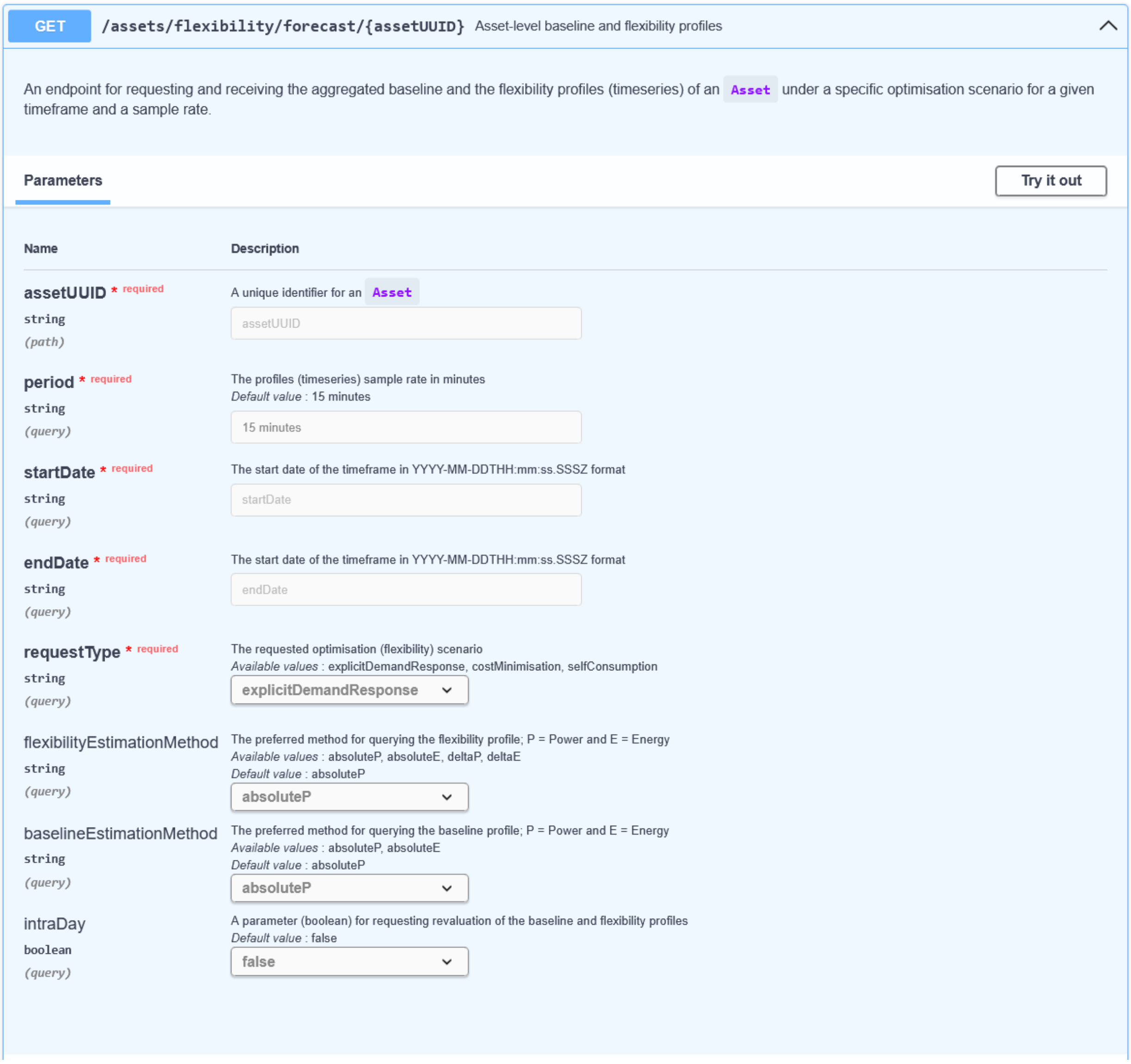


Figure 12: Swagger documentation of the BFMS


```
{
  "requestInfo": {
    "uuid": "47318ae4-39ad-4c43-b078-00446524e023",
    "flexibilityEstimationMethod": "absoluteP",
    "requestType": "explicitDemandResponse",
    "requestScope": "assetLevel",
    "assetUuid": "82895f8c-4458-11eb-9464-0024e84abf93",
    "startDate": "2024-02-16T00:00:00.000+0000",
    "endDate": "2024-02-17T00:00:00.000+0000",
    "createdAt": "2024-02-15T18:30:57.934+0000",
    "period": "15 minutes",
    "intraDay": false
```

```
  },
  "baselineForecasting": [
    {
      "dtStart": "2024-02-16T00:00:00.000+0000",
      "dtEnd": "2024-02-16T00:15:00.000+0000",
      "mean": 0,
      "unit": "W",
      "createdAt": "2024-02-15T11:41:36.463+0000"
    },

                              . . .
  ],
  "flexibilityForecasting": [
    {
      "dtStart": "2024-02-16T00:00:00.000+0000",
      "dtEnd": "2024-02-16T00:15:00.000+0000",
      "meanUp": 0,
      "meanDown": 0,
      "unit": "W",
      "createdAt": "2024-02-15T11:41:36.463+0000"
    },
                              . . .

  ]
```


Figure 13: Payload schema of the BFMS

3.6 Pre-validation activities & Preliminary tests

3.6.1 Theoretical pre-validation case studies

In this section, HVACs, DHW devices, and PV generation are considered as a general pre-validation example, not necessarily reflecting the final operation of the BFMS. The final operation will also depend on the list of available assets from the pilot sites of the project. Power generation is assumed to be not controllable. Having presented the problem formulation and the input/output parameters of the model, data preparation is outlined, and the results of the case study are presented further below.

The flexibility forecasting algorithm initially identifies the number of demand assets and their corresponding type. Then for each asset, the number of activations is obtained, together with start time and end time for each one, and the mean power for each activation interval. Note that if there is only one 15-minute time interval between two consecutive activations, they are combined into a single one.

The case study considers three demand assets (two DHW devices and an HVAC) and one generation asset (PV); PV generation is not controllable. For the given input, $N = 3$ (number of demand assets), $a = [2, 2, 3]$ (activations per demand asset), and $T = 96$ (number of time steps). Figure 14, Figure 15, and Figure 16 illustrate the baseline power consumption of DHW 1, DHW 2, and HVAC, respectively. Finally, Figure 17 shows the total demand along with the PV generation throughout the day.

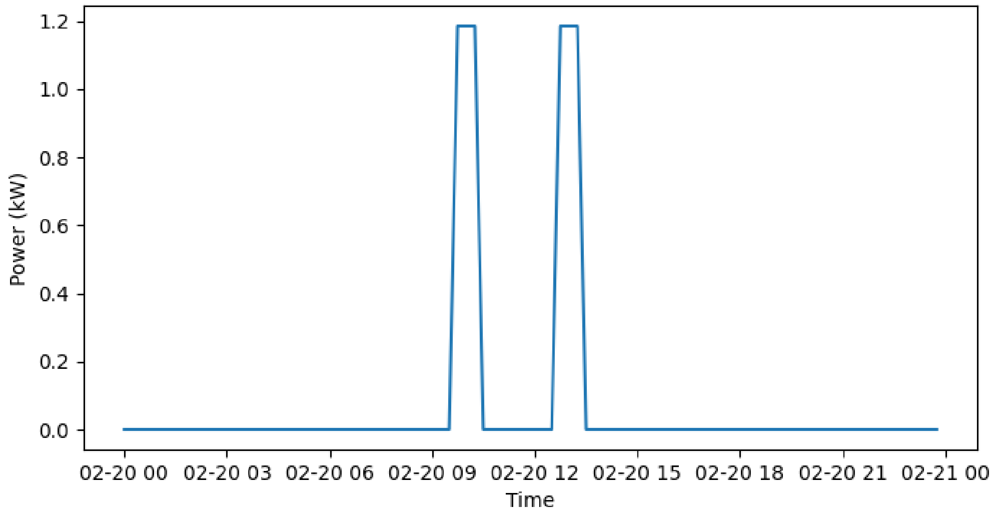


Figure 14: Power consumption of DHW 1 – Baseline

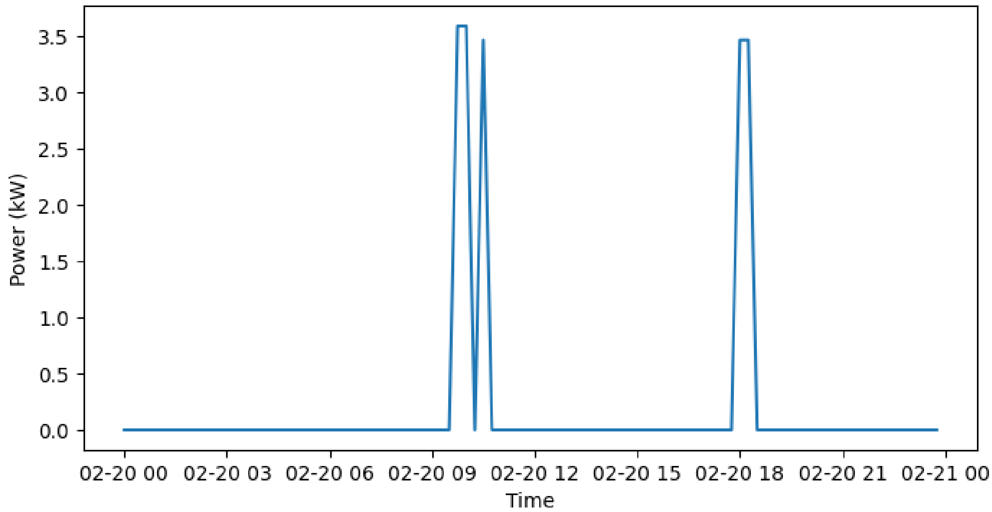


Figure 15: Power consumption of DHW 2 – Baseline

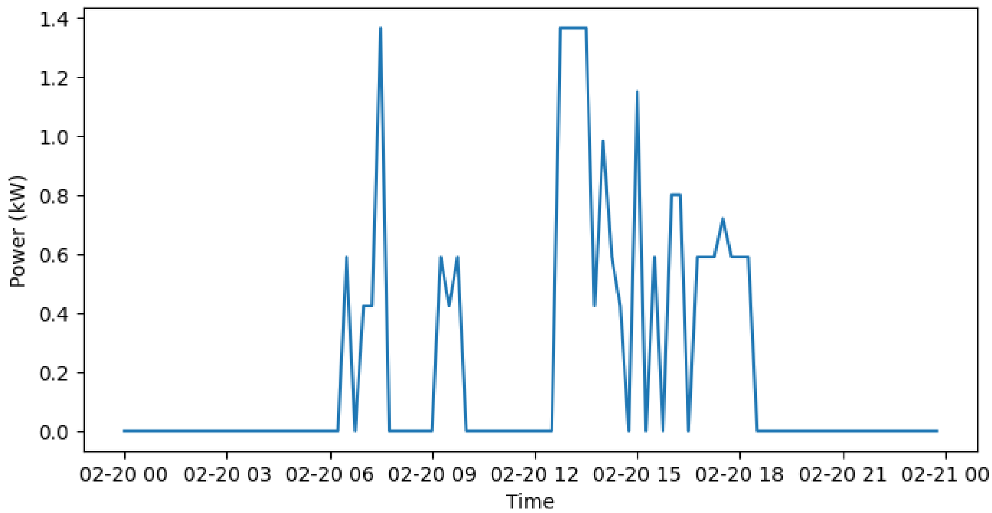


Figure 16: Power consumption of HVAC – Baseline

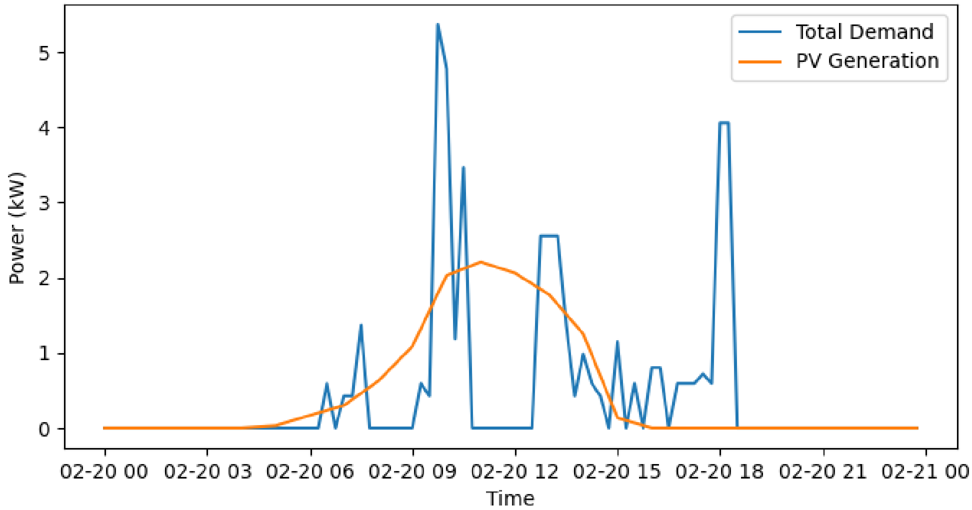


Figure 17: Total demand and PV generation– Baseline

In the following subsections the results of two scenarios are presented, the first one with the assets mentioned above plus a 2 kW/4 kWh energy storage system (ESS) to demonstrate the effect of these devices on flexibility provision, and the second one only with the assets mentioned above (without storage – Section 3.6.1.2).

3.6.1.1Scenario with storage

As mentioned earlier, the baseline profile evaluation process will be described in detail in the second version of this deliverable. In this version a base case of minimum cost is considered. For this reason, initially, cost minimization is executed (without any flexibility provision), and the baseline power profile (net demand) of the system for each time step is obtained, defined as:

$$P_t = d_t + P_{ch,t} - g_t - P_{dch,t} \tag{18}$$

where d_t represents total demand of DHW and HVAC devices at each time step t, $P_{ch,t}$ is the charging power of the ESS at time t, g_t is the PV generation at time t, and $P_{dch,t}$ is the discharging power of the ESS at time t. P_t can be either positive (indicating import), or negative (indicating surplus/export). This power profile can be considered as a baseline profile (see Figure 18).

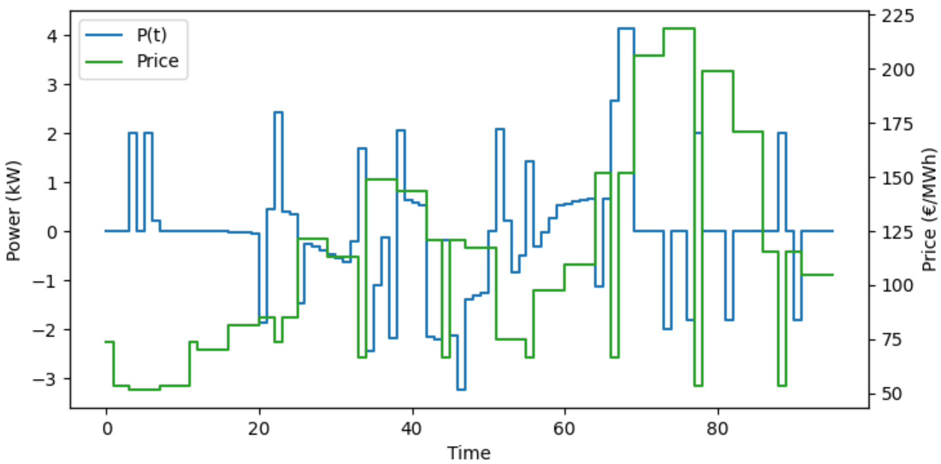


Figure 18: Base case power profile of the system for cost minimization

It can be observed that the power profile (blue line) responds to the price profile (green line), going up when the other goes down. The total cost in this case is -1.35€, i.e., benefit of 1.35€.

Having run cost minimization with no flexibility provision, considering that the resultant power profile can serve as baseline (blue line in Figure 18), flexibility provision is executed for a specific time interval for different reward prices. The optimization problem considers a specific time interval, e.g., 17:00 - 18:00 (time steps 68 - 71), and a power P_F that can be provided throughout the whole flexibility window. The result for a reward price (or availability fee according to [8]) of 1€/kW/h (see Figure 19) is presented at first. The available flexibility that can be offered throughout the whole interval is P_F = 1.9 kW; and although during the first two time steps a greater amount of power can be provided, this cannot be sustained for four time steps. A table for different prices is also provided (Table 3).

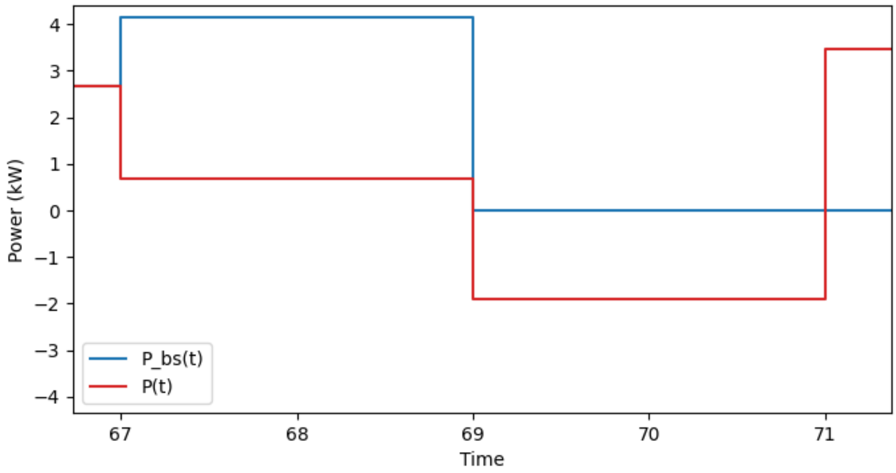


Figure 19: Flexibility provision for an availability fee of 1€/kW/h

Table 3: Available flexibility for different availability fees

Availability Fee (π)	Available Flexibility (P_F)
0.01 - 0.15 €/kW/h	0 kW
0.16 - 0.28 €/kW/h	0.95 kW
0.29 – 10 €/kW/h	1.9 kW

This section demonstrates the need for running an optimization for each time interval, given that the total energy is maintained, and therefore there is a rebound effect, which usually occurs right after the DR event. The optimization for the time interval 17:00 – 18:00 has already been executed. Let's now take a look at the DR event and the corresponding rebound effect, which occurs right after the DR event is over (see Figure 20). The surface between the lines is the same before and after the DR event (occurs during time steps 68-71). The rebound effect occurs during time steps 72, 73, 74, and 77.

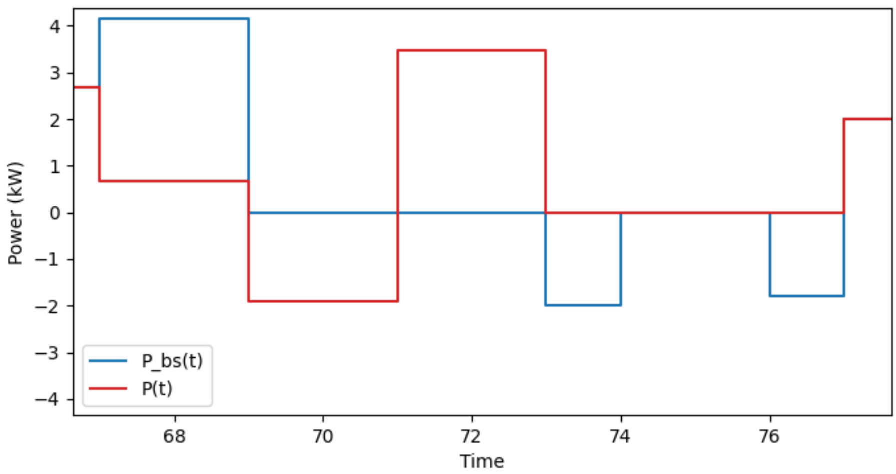


Figure 20: DR (red line below blue line) for 17:00 - 18:00

This means that a given time interval for DR corresponds to a specific DR profile (for the whole day, or at least some hours after the DR event). In other words, the interval of the DR affects not only the power of the DR profile during this time, but also the time steps beyond the DR event. This means that an optimization needs to be executed for each time interval that flexibility must be offered.

3.6.1.2 Scenario without storage

This section presents the case study results without storage. First, cost minimization is executed (with no flexibility provision) to get the baseline profile. This is shown in Figure 21. P_t is defined here as:

$$P_t = d_t - g_t \tag{19}$$

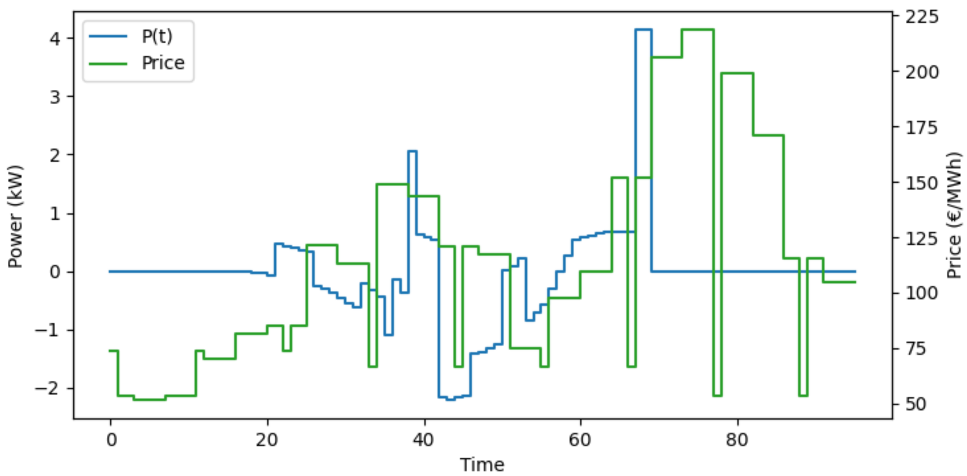


Figure 21: Baseline power profile (P_t) along with electricity price

Flexibility provision for 17:00 - 18:00 is then executed for an availability fee of 1 €/kW/h, which results in the situation illustrated in Figure 22. This corresponds to no flexibility provision, possibly because it cannot be sustained for the entire time interval. A shorter time interval is tested, specifically for 17:00 - 17:30 (time steps 68 and 69), the result being presented in Figure 23. A 3.47 kW flexibility provision can be observed during this time interval, i.e., demand reduction during time steps 68 and 69 (17:00 - 17:30), as well as a demand increase 30 minutes later (during time steps 72 and 73, i.e., 18:00 - 18:30).

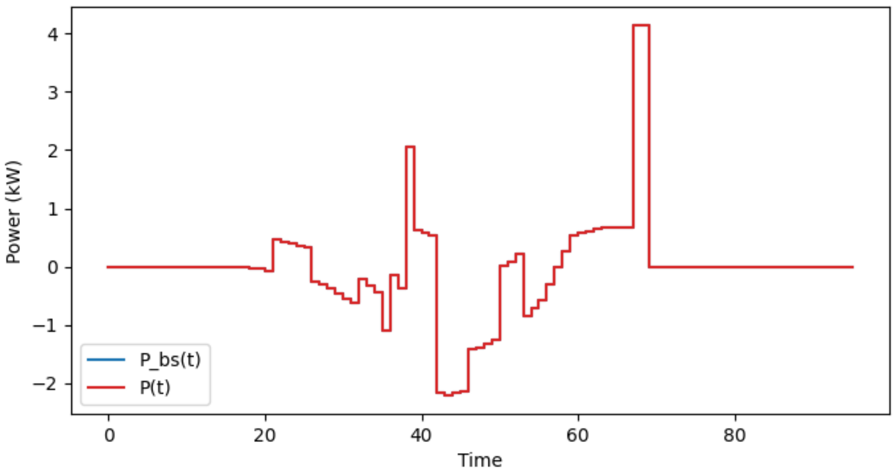


Figure 22: Flexibility provision during 17:00 - 18:00 for an availability fee of 1 €/kW/h

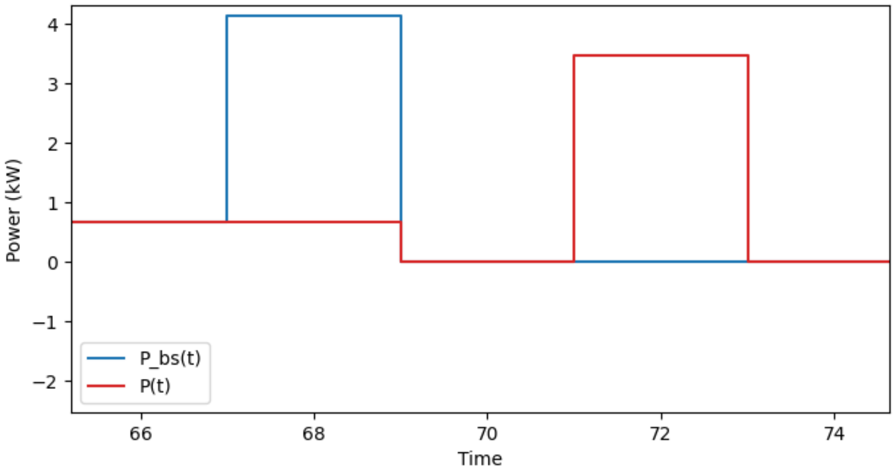


Figure 23: Flexibility provision during 17:00 - 17:30 for an availability fee of 1 €/kW/h

Simulation results demonstrate the need that a single optimization run is required for each possible DR time interval of the next day.

The impact of maintaining the same energy on time intervals beyond that of the DR event also needs to be highlighted, further indicating why optimization should be performed for each time interval, as this results in a different DR profile for each interval.

3.6.2Actual end-to-end test results

This section provides end-to-end test results, as executed to HYPERTECH's demo site.

3.6.2.1 Sample results: BFMS - Baseline & Flexibility Profiling (AC unit)

The two following figures, Figure 24 and Figure 25, illustrate flexibility profiles for an AC unit, specifically showing downward and upward flexibility, respectively. Figure 24 illustrates how the power consumption of the AC unit decreases for specific timeslots compared to the baseline consumption, as calculated by the BFMS. Figure 25 shows the opposite scenario, where the AC unit increases its power consumption for specific timeslots.

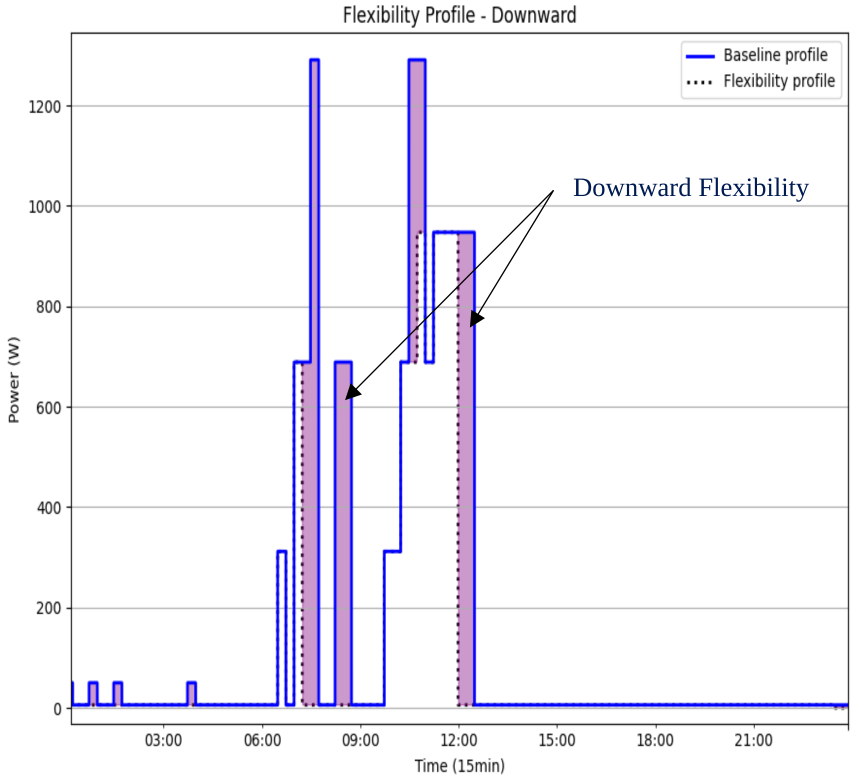


Figure 24: Flexibility profile, downward

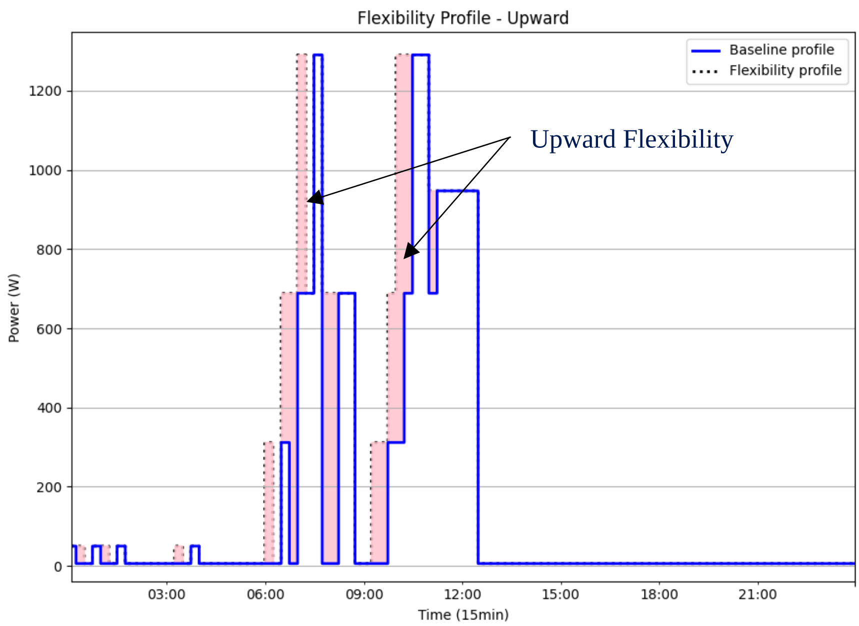


Figure 25: Flexibility profile, upward

3.6.2.2 Sample results: BFMS – Flexibility request & Dispatching (AC unit)

3.6.2.2.1. Flexibility Profile/Requests

Figure 26 shows another example of upward and downward flexibility calculation for an AC unit, primarily concentrated around midday, with magenta bars representing upward flexibility and purple bars showing downward flexibility.

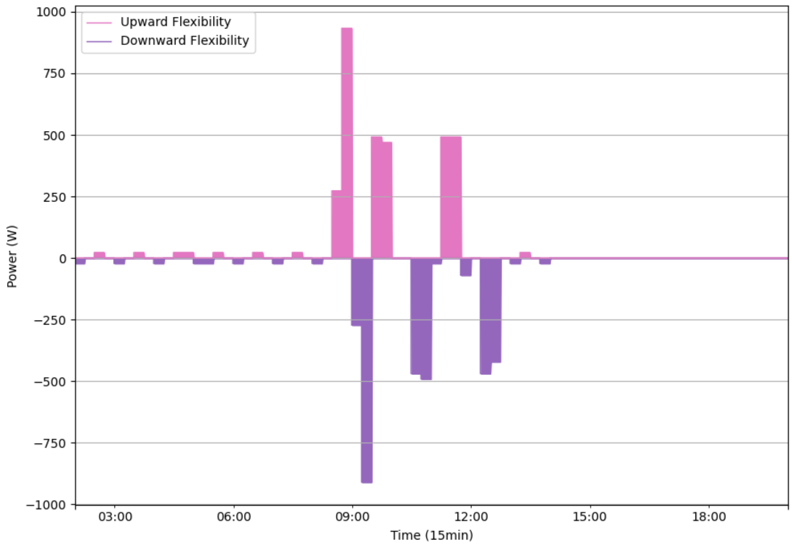


Figure 26: Flexibility profiles, requests

3.6.2.2.2. Flexibility Dispatching

Figure 27 provides an example of control dispatching based on the forecasted flexibility values, showing baseline power consumption (blue line), baseline estimation (dashed line) and DR events (red line).

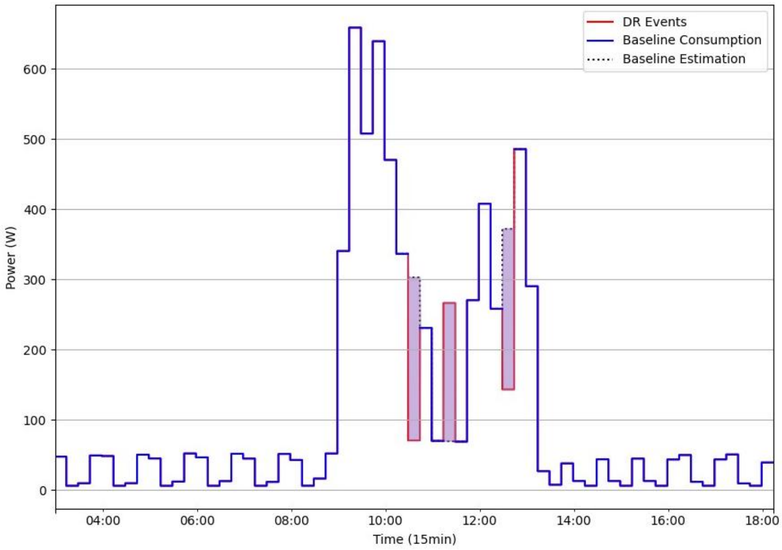


Figure 27: Flexibility dispatching

3.6.2.3 Sample results: Asset power consumption profiling (Inverter-driven HVAC)

Figure 28 depicts results of the power consumption profiling for an inverter-driven HVAC over a period of several summer days. Measured (blue line) and predicted (red line) values are in good agreement, showcasing the effectiveness of the applied approach.

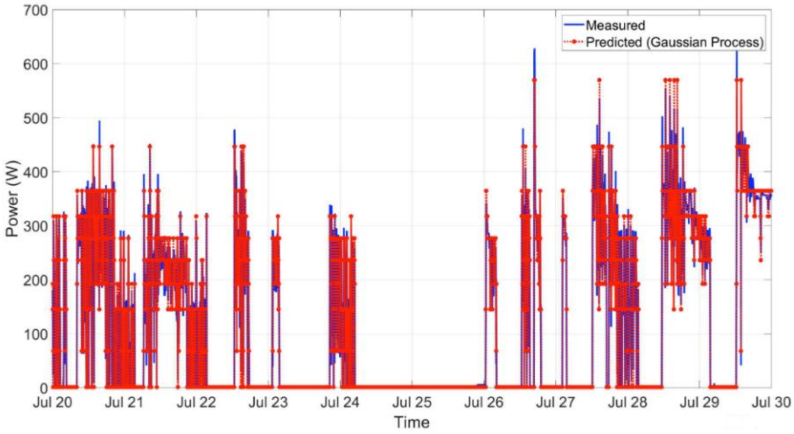


Figure 28: Asset power consumption profiling (Inverter-driven HVAC)

3.6.2.4 Sample results: Thermal comfort profiling

Lastly, Figure 29 shows a thermal comfort profile using SAX, focusing on indoor temperature change. With the temperature values being normalized, the blue dots represent actual temperature data, while the orange markers show the SAX approximation. During the occupied period (illustrated in grey), the temperature is maintained inside the thermal comfort zones (highlighted in green). Once again, a good agreement between measured and calculated data can be observed.

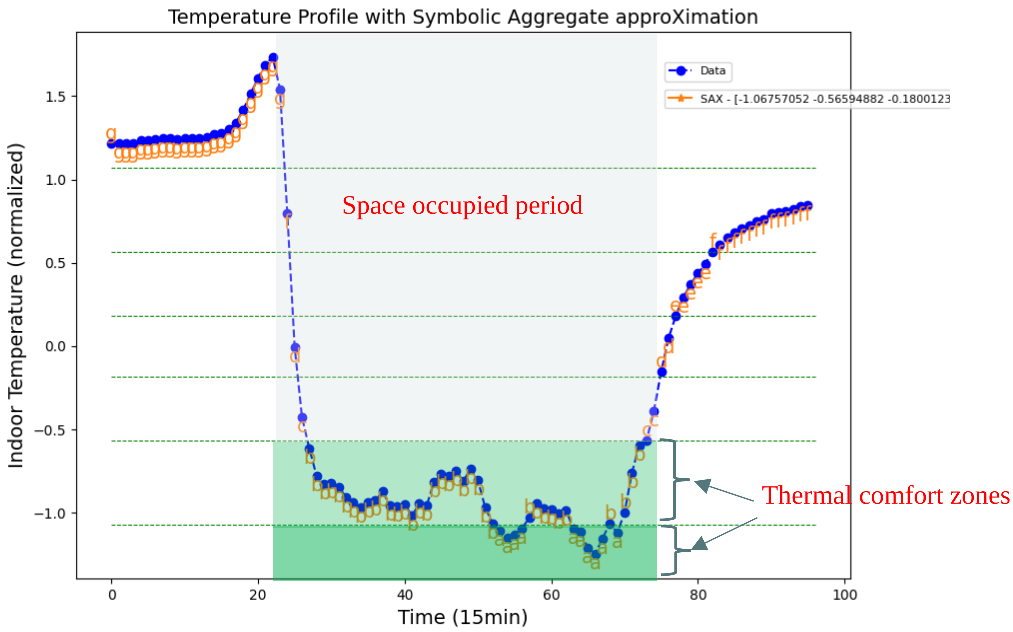


Figure 29: Thermal comfort profiling

3.7 Development & integration status

So far (M26 of the OPENTUNITY project), a fully functional proof of concept (PoC) of the BFMS has been developed. HYPERTECH's data has been used to demonstrate the feasibility and effectiveness of the proposed approach. Ongoing efforts are focused on integrating this PoC with other HEMS/BEMS, as well as with FSPs and the NODES platform. The progress and outcomes of these integrations will be documented in detail in the next version of this deliverable.

3.8 Requirements in Equipment & Infrastructure

The successful operation of the BFMS relies on the HEMS/BEMS installed at the consumer/prosumer premises, and more particularly on the installation of measurement and actuation equipment to retrieve the data necessary for the algorithms training and forecasting as well as for the dispatch of control signals based on the flexibility forecast. This usually involves the installation of the following equipment:

- temperature/humidity sensors,
- occupancy sensors,
- energy meters (total and asset ones, i.e., submetering),
- actuators (control devices)

Ambient and occupant sensors are used for the training of the Comfort Profiling and Building Thermal Modeling Modules, while energy related data are used for the Asset Profiling Module. The actuators are necessary for the successful execution of commands dictated by the incoming DR signals. Each OPENTUNITY pilot site is equipped with a HEMS/BEMs and shall be capable to satisfy theses needs. A detailed description of each HEMS/BEMs is provided in section 2.

3.9 Assumptions and restrictions

The correct BFMS operation highly depends on the reception of all necessary data from the pilot sites and the respective flexibility assets, a necessary step for the training of the flexibility algorithms of the component. Hence, any malfunctions in IoT equipment can have a significant impact on the BFMS functioning, as incomplete metering may lead to inadequately trained models. Similarly, failure to dispatch control commands to the building assets will affect the delivery of estimated flexibility at the pilot sites of the project.

In addition, the pilot sites of the project remain open to the inclusion of new buildings / dwellings and respective assets. In case new asset types are added to the project, the BFMS will incorporate the modeling of these assets to its operation, expanding in this way the capability of flexibility forecast and dispatch. Moreover, integration with the HEMS/BEMS systems is still ongoing and as such, changes might be necessary to accommodate the intricacies of the systems at hand.

Lastly, there are ongoing discussions with the partners implementing the FSPs so as to align BFMS with their specific context and requirements.

4. NILM and Behaviour Analytics

4.1 Component overview

As electricity prices continue to rise, optimizing electricity consumption at economic and energy level throughout the value chain is essential. Energy tariffs are constrained by the energy market, offering limited profit margins. On the other hand, while people are interested in appliance energy consumption, they lack tools to access energy-related information. This is where Non-Intrusive Load Monitoring (NILM) becomes relevant.

NILM is a methodology that detects appliance usage based on total power or energy consumption curves. It identifies patterns using exogenous variables such as active power, reactive power, current, voltage, and energy consumption, disaggregating the power usage of each appliance from the total consumption curves.

NILM has been extensively researched, with numerous algorithms proposed [9]. Initial approaches included linear optimizations based on maximum appliance power consumption and event detection [10] [11], Hidden Markov Models for pattern recognition approaches [12] and Python toolkits for developing and training various mathematical algorithms were also developed [13].

Recently, the AI boom has introduced machine learning and deep learning methodologies, yielding promising results. Supervised learning techniques, which use real data for model evaluation, have shown the best performance. Classification techniques predict appliance usage in binary terms (on/off) using models like Random Forest, Support Vector Machines, and Neural Networks [14]. Regression techniques predict numerical power consumption using models like XGBoostRegressor and Long Short-Term Memory neural networks [15]. However, supervised methods require labeled data, necessitating intrusive load monitoring or user data tagging. To prevent this, unsupervised and semi-supervised methods, such as k-means clustering [16] [17], have been developed to detect patterns and determine appliance usage with the cost of reduced model performance.

This project employs regression techniques to predict real-time individual appliance power consumption, creating generalized models applicable to any household with metering data. This approach clarifies electrical measurement information for end-users, indicating appliance performance more accurately. Additionally, these generalized models support demand response campaigns by providing consumption curves for flexible assets.

4.2 Component Prototype

The architecture defined in D2.3 with the SGAM framework can be seen in Figure 30. Component layer is shown, with the different components involved.

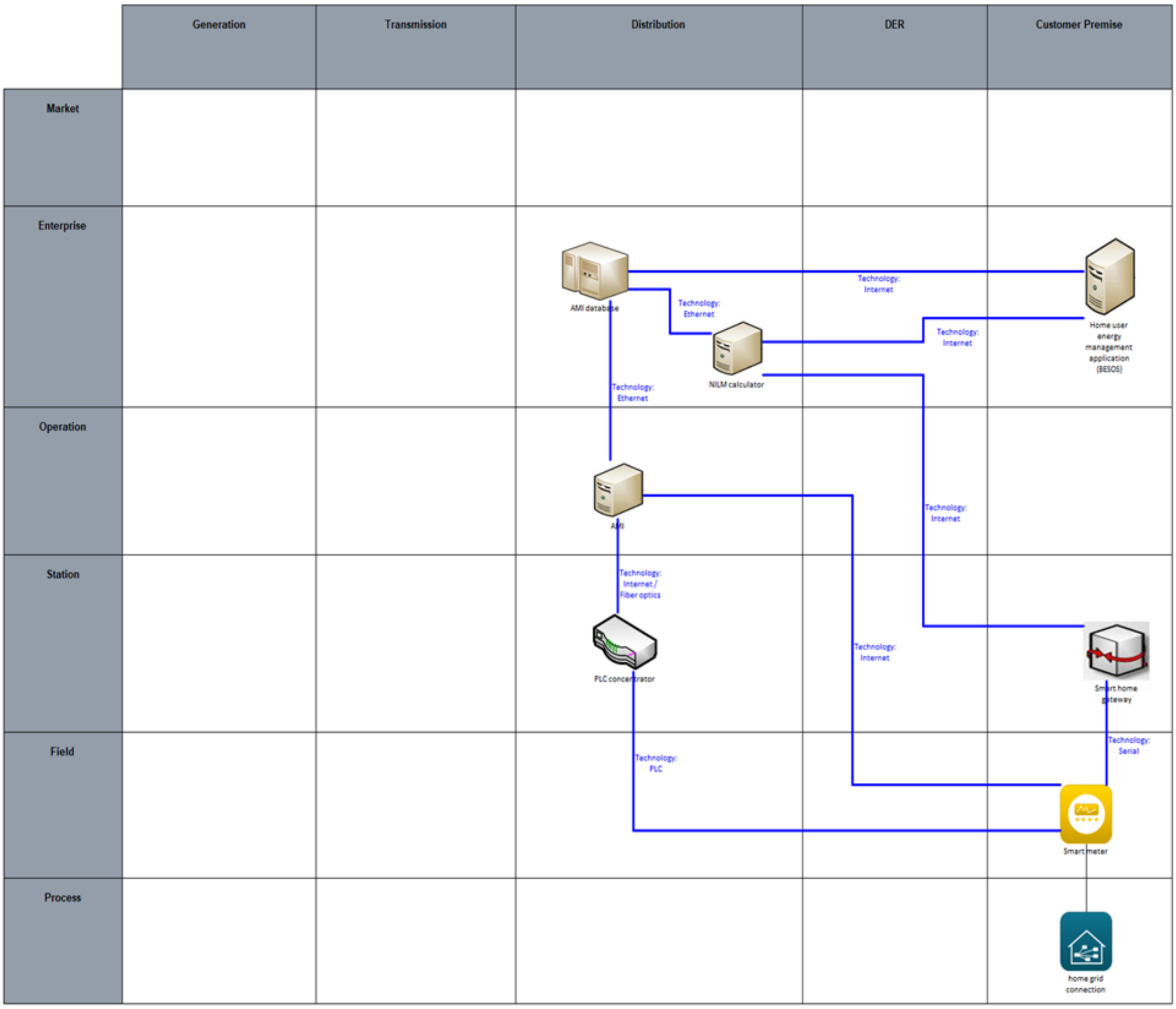


Figure 30: NILM and behavior analytics SGAM component layer

The central component of the NILM Use Case. is the NILM calculator. A subcomponent diagram can be seen in Figure 31 and is defined as follows:

- **Metering data:** This includes all information from smart meters and submetering in households selected from the pilots. Data is retrieved from their respective databases and Dataspace connectors defined in the project. It includes electrical measurements such as active and reactive power, current, voltage, etc.
- **API / Dataspace connector:** Gateways to retrieve information, including real-time measurements and historical data necessary to train the AI models. The primary aim is to use Dataspace connectors, but a database API can also be used to retrieve information directly.
- **NILM calculator:** This module retrieves real-time electrical measurement data to predict the individual power consumption of each appliance and sends this information back to the Dataspace connector or the corresponding database. The prediction models are stored and downloaded from the model storage software.
- **Model Storage:** Software designed to store AI models along with their evaluation metrics, datasets, and graphical content, including versioning and deployment information. Models

are trained offline using historical and publicly available data. Uploading and downloading of models are managed via the corresponding API.

- **NILM GUI:** This graphical user interface displays all electrical measurement information, including real-time, historical, and prediction data, using a dashboard. One page of the GUI is designed to show energy certificates. Users can input specific appliance information, and through an API, both theoretical and current energy certificates will be displayed.
- **NILM certificates:** This module calculates energy certificates based on the selected appliance and various input parameters. It computes a theoretical certificate and, using historical data, generates a certificate based on actual energy usage, allowing for comparison.

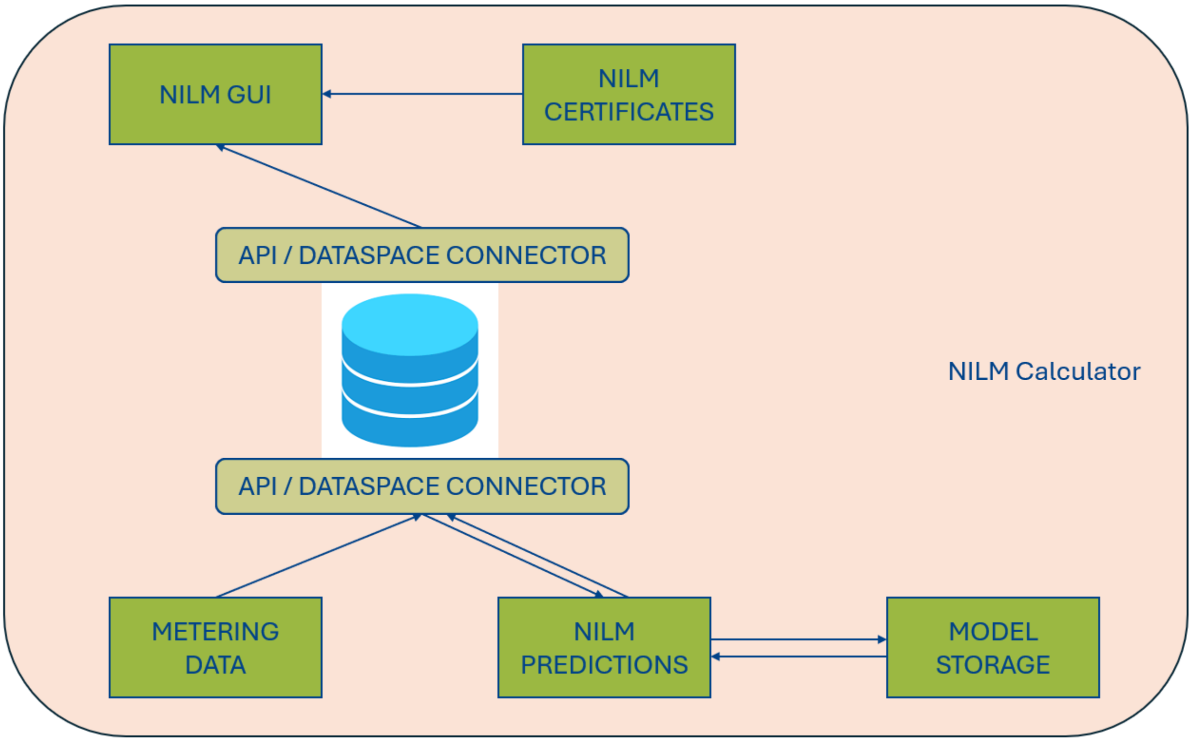


Figure 31: NILM calculator detailed subcomponent architecture.

The GUI is componentized, meaning it can be integrated into other software systems. Specifically, it is intended to be added to the HEMS software of ETRA called BESOS.

Components and subcomponents are designed for cloud deployment, allowing resource scaling as needed. The architecture utilizes microservices and Docker, enabling independent refinement of each subcomponent and simplifying deployments. Communication between subcomponents is managed via API, following the microservice architecture, while communication with pilot data is handled using Dataspace connectors.

4.3 Proposed Methodology

The methodology selected for NILM calculations involves supervised machine learning methods. The goal is to train generalized models for each appliance using historical data from publicly available datasets and the project's pilot data. This methodology requires Intrusive Load Monitoring for training, validation, and evaluation, but once trained, the models can be applied to any household.

The datasets available online consist of measurements from various households worldwide. Research papers have attempted to consolidate these datasets into a common repository [18]. These datasets vary in granularity, period, number of households, and collected electrical measurements, allowing for the creation of diverse machine learning models based on data needs. The primary benefit of this method is that generalized models can be applied to any household providing total energy or power consumption. Additional electrical measurements can enhance model performance, and submetering data can re-train models for increased accuracy.

Machine learning models are trained using collected information. The target variable is the individual power consumption of each appliance. Models have been trained using only active power consumption and both active and reactive power consumption as exogenous variables due to the lack of datasets with more than active and reactive power included and other electrical-related information such as voltage or current. Cross-validators and hyperparameter optimizers, such as BayesianSearchCV and RandomSearchCV, are used to determine optimal hyperparameters, minimizing prediction errors.

The optimizators need the numeric or class intervals to search for each hyperparameter, using a randomized search and trying to reduce the evaluation metric desired, the optimizator starts trying out some models with random hyperparameters in the defined grid for a defined number of iterations. Once the iterations are completed, the best performing model is served. The cross validators used are BayesianSearchCV and RandomSearchCV.

The main packages used for modeling are XGBoost and TensorFlow. XGBoost, short for Extreme Gradient Boosting, is a powerful and efficient implementation of gradient-boosted decision trees. It is popular for its fast learning and inference times, making it one of the best-performing models for structured/tabular data. The model works by sequentially adding decision trees, where each new tree corrects the errors of the previous ones. This boosting process helps to improve accuracy and reduce overfitting. XGBoost also includes advanced features like tree pruning, regularization, and parallel processing to enhance performance.

TensorFlow, on the other hand, is an open-source deep learning framework developed by Google. It is widely used for building and training neural networks. TensorFlow supports various types of neural network architectures, including Convolutional Neural Networks (CNNs) for image processing, Recurrent Neural Networks (RNNs) for sequential data, and fully connected networks for general-purpose tasks. The training process in TensorFlow involves optimizing the weights of the neural network using algorithms like stochastic gradient descent (SGD) and its variants (e.g., Adam, RMSprop). TensorFlow's flexibility and scalability make it suitable for both research and production environments.

MLflow open-source software is deployed to track the progress of each trained model. Evaluation metrics, datasets, trained models, and performance plots are available for each training run. MLflow also supports versioning and model downloading via API.

Regarding energy certificates, the European Commission defines a common procedure to calculate the label for each appliance [19]. These labels often involve annual or per-cycle energy consumption values, calculated during testing periods by manufacturers under specific conditions. These theoretical values can be compared with real values from submetering or NILM calculations, providing insights into appliance performance using energy labeling.

4.4 Technology stack and Implementation tools

The technologies selected can be divided between the different components:

Table 4: Libraries and Technologies used in NILM predictions

Library/Technology Name	Version	License
Python	3.10.12	PSF License Agreement and the Zero-Clause BSD license
Pandas	2.0.3	BSD 3-Clause "New" or "Revised" License
Numpy	1.24.4	Modified BSD license
Matplotlib	3.7.5	PSF license
xgboost	2.0.3	Apache License 2.0
tensorflow	2.10	Apache License 2.0
Scikit-learn	1.3.2	BSD 3-Clause "New" or "Revised" License
influxdb	5.3.2	MIT license
mlflow	2.11.1	Apache License 2.0
schedule	1.2.1	MIT license

Table 5: Libraries and Technologies used in Model Storage

Library/Technology Name	Version	License
MLflow	2.13.0	Apache License 2.0

Table 6: Libraries and Technologies used in NILM GUI

Library/Technology Name	Version	License
Zustand	4.1.1	MIT License
React-router-dom	6.22.2	MIT License

React-i18next	12.2.0	MIT License
React-favicon	2.0.3	MIT License
React-dom	18.2.0	MIT License
React	18.2.0	MIT License
Notistack	3.0.1	MIT License
Natsex	2.0.16	MIT License
Moment	2.29.4	MIT License
Meteor-node-stubs	1.2.10	MIT License
Lodash	4.17.21	MIT License
I18next-browser-languagedetector	6.1.5	MIT License
I18next	21.9.2	MIT License
Etra-table-material	0.0.21	MIT License
Etra-mui-theme	2.0.4	MIT License
Etra-mui-components	2.0.1	MIT License
Etra-metrics-components	4.0.147	MIT License
Citric-helper	1.5.17	MIT License
@mui/styles	6.1.2	MIT License
@mui/material	5.15.19	MIT License
@mui/icons-material	5.15.19	MIT License
@fontsource/roboto	5.0.8	MIT License
@emotion/styled	11.13.0	MIT License
@emotion/react	11.11.4	MIT License
@babel/runtime	7.24.0	MIT License

Table 7: Libraries and Technologies used in NILM certificates

Library/Technology Name	Version	License
fastapi	0.115.5	MIT license
pandas	2.2.3	BSD 3-Clause "New" or "Revised" License

numpy	2.2.1	Modified BSD license
requests	2.32.3	Apache License 2.0
influxdb	5.3.2	MIT license
Uvicorn	0.32.1	BSD 3-Clause "New" or "Revised" License
Python	3.10.12	PSF License Agreement and the Zero-Clause BSD license

4.5 Input/Output parameters

The inputs and outputs necessary to make use of the components are:

Table 8: Inputs required, and outputs delivered by NILM predictions

Inputs (* optional)	**Units**
Active power (Total from the household)	Watts
Reactive power (Total from the household) (*)	Volts-Ampere reactive
Current (*)	Ampere
Voltage (*)	Volts
List of appliances inside the household	Washing machine, dryer, air conditioner, water heater, fridge, oven, etc.
Outputs	Units
Active power (Individual consumption of each appliance monitored)	Watts

Table 9: Inputs required, and outputs delivered by Model Storage

Inputs	**Units**
Model name	-
Model version	-
Appliance	-
Outputs	Units
Machine-learning model class	-

Table 10: Inputs required, and outputs delivered by NILM GUI

Inputs (* optional)	Units
Active power (Total from the household)	Watts
Reactive power (Total from the household) (*)	Volts-Ampere reactive
Current (*)	Ampere
Voltage (*)	Volts
List of appliances inside the household	Washing machine, dryer, air conditioner, water heater, fridge, oven, etc.
Active power (Individual consumption of each appliance monitored)	Watts
Outputs	Units
-	-

Table 11: Inputs required, and outputs delivered by NILM Certificates disaggregated into the different appliances

Fridge and Freezer inputs	Units
Annual energy consumption	kWh/annum
Number of doors	-
Design type	Free-standing or built-in
Compartment type	Fresh food, wine storage, etc.
Compartment volume	dm^3 or litres
Compartment defrosting type	Auto-defrost or manual defrost
User	-
Fridge and Freezer outputs	Units
Theoretical energy efficiency index	-
Current energy efficiency index	-
Theoretical energy label	A to G
Current energy label	A to G
Washing machine inputs	Units
Rated capacity	Kg
Energy consumption per cycle, eco 40-60 programme	kWh

User	-
Theoretical energy efficiency index	-
Current energy efficiency index	-
Theoretical energy label	A to G
Current energy label	A to G
Washer-dryer inputs	Units
Rated capacity (complete cycle)	Kg
Energy consumption per cycle (complete cycle)	kWh
Rated capacity (Washing cycle)	Kg
Energy consumption per cycle (washing cycle)	kWh
User	-
Washer-dryer outputs	Units
Theoretical energy efficiency index (complete cycle)	-
Current energy efficiency index (complete cycle)	-
Theoretical energy label (complete cycle)	A to G
Current energy label (complete cycle)	A to G
Theoretical energy efficiency index (washing cycle)	-
Current energy efficiency index (washing cycle)	-
Theoretical energy label (washing cycle)	A to G
Current energy label (washing cycle)	A to G
Dishwasher inputs	Units
Assigned capacity	Liters
Width	Centimetres
Energy consumption (per cycle, based on the eco programme)	kWh

User	-
Theoretical energy efficiency index	-
Current energy efficiency index	-
Theoretical energy label	A to G
Current energy label	A to G
Dryer inputs	Units
Rated capacity	Kg
Weighted Annual Energy Consumption	kWh/annum
User	-
Dryer outputs	Units
Theoretical energy efficiency index	-
Current energy efficiency index	-
Theoretical energy label	A^{+++} to D
Current energy label	A^{+++} to D
Air conditioner inputs	Units
Seasonal Energy Efficiency Ratio (SEER)	-
Annual Electricity Consumption (Cooling mode)	kWh/annum
Seasonal Coefficient of Performance (SCOP)	-
Annual Electricity Consumption (Heating mode)	kWh/annum
User	-
Air conditioner outputs	Units
Theoretical energy efficiency index (SEER)	-
Current energy efficiency index (SEER)	-
Theoretical energy label (SEER)	A^{+++} to G
Current energy label (SEER)	A^{+++} to G
Theoretical energy efficiency index (SCOP)	-
Current energy efficiency index (SCOP)	-

Theoretical energy label (SCOP)	A^{+++} to G
Current energy label (SCOP)	A^{+++} to G
Water Heater inputs	Units
Load profile	3XS to XXL
Water heating energy efficiency	percentage
Annual electricity consumption	kWh
User	-
Water Heater outputs	Units
Theoretical energy efficiency index	-
Current energy efficiency index	-
Theoretical energy label	A^{+++} to G
Current energy label	A^{+++} to G
Television inputs	Units
On mode power demand in Standard Dynamic Range (SDR)	Watts
Visible screen area	dm^2
User	-
Television outputs	Units
Theoretical energy efficiency index	-
Current energy efficiency index	-
Theoretical energy label	A to G
Current energy label	A to G
Oven inputs	Units
Oven volume	Liters
Energy consumption per cycle (fan-forced convection mode)	kWh
User	-
Oven outputs	Units
Theoretical energy efficiency index	-
Current energy efficiency index	-

Theoretical energy label	A^{+++} to D
Current energy label	A^{+++} to D

4.6 Interfaces & API documentation

To clearly understand the data flows, refer to Figure 32. The goal is to have all information available in the OPENTUNITY Data Space using the Data Space Connector.

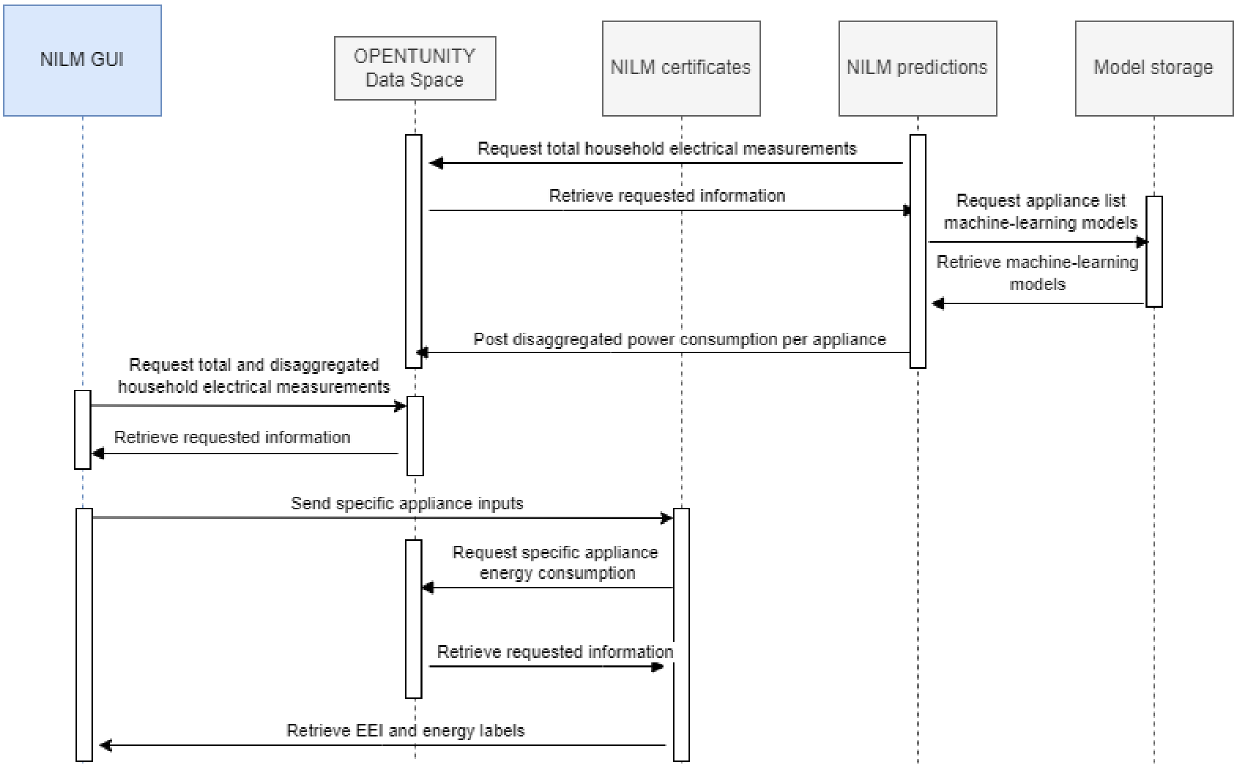


Figure 32: NILM calculator dataflows diagram.

First, raw electrical measurements from households should be available to request total values and the list of connected appliances from the NILM predictions tool. AI models are then downloaded from the Model Storage using its Python API client, see [20]. This model storage is created using MLflow, which provides a GUI for performing actions that can also be done through the API, including run modifications, model registrations, results visualizations, etc. All machine-learning models are stored in "runs", see Figure 33. These runs save evaluation metrics, models, datasets used, and tags to track each model's performance. Once runs are completed and compared, the best-performing models can be registered and versioned using the UI or API for future download if necessary.

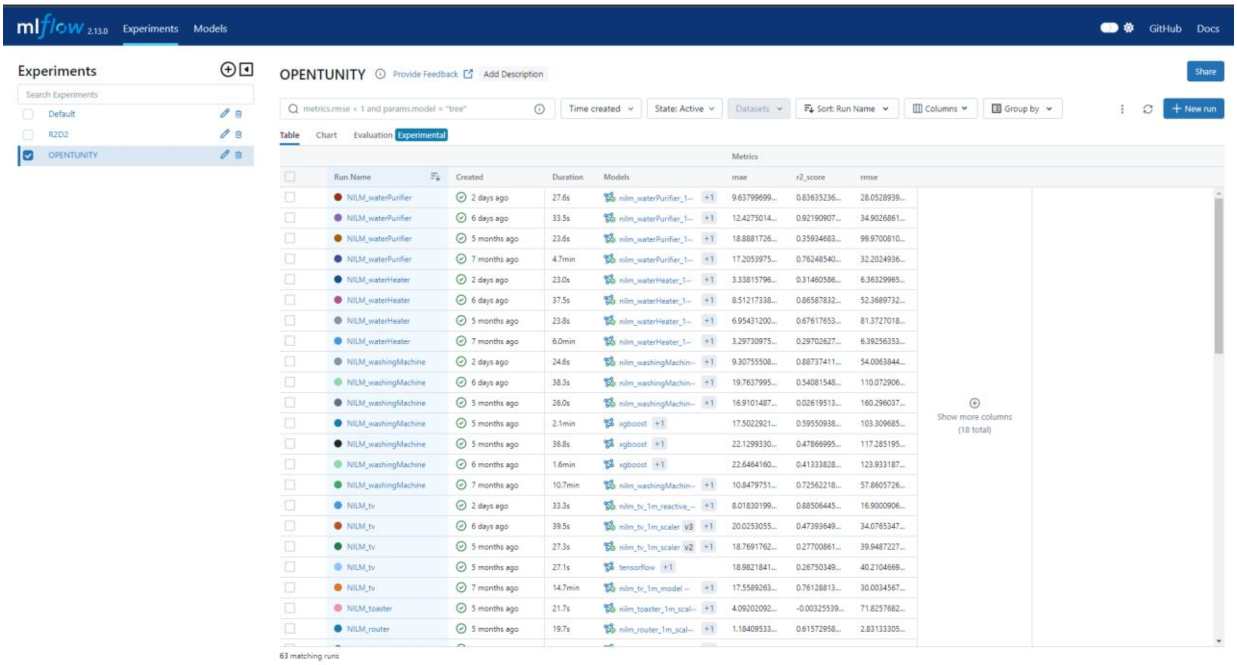


Figure 33: MLflow graphical user interface, runs page.

When models are downloaded, real-time predictions are calculated, and disaggregated power consumption results are updated into the Data Space. This process is scheduled to match the granularity of the raw smart meters. So far, no Data Space Connectors have been deployed; instead, cloud databases have been used for these actions.

At this point, total and disaggregated electrical measurements are collected from the cloud database, and the NILM GUI can display a series of KPIs and metrics using Grafana. For this purpose, the dashboard is included in the React component using an iframe, as shown in Figure 34.

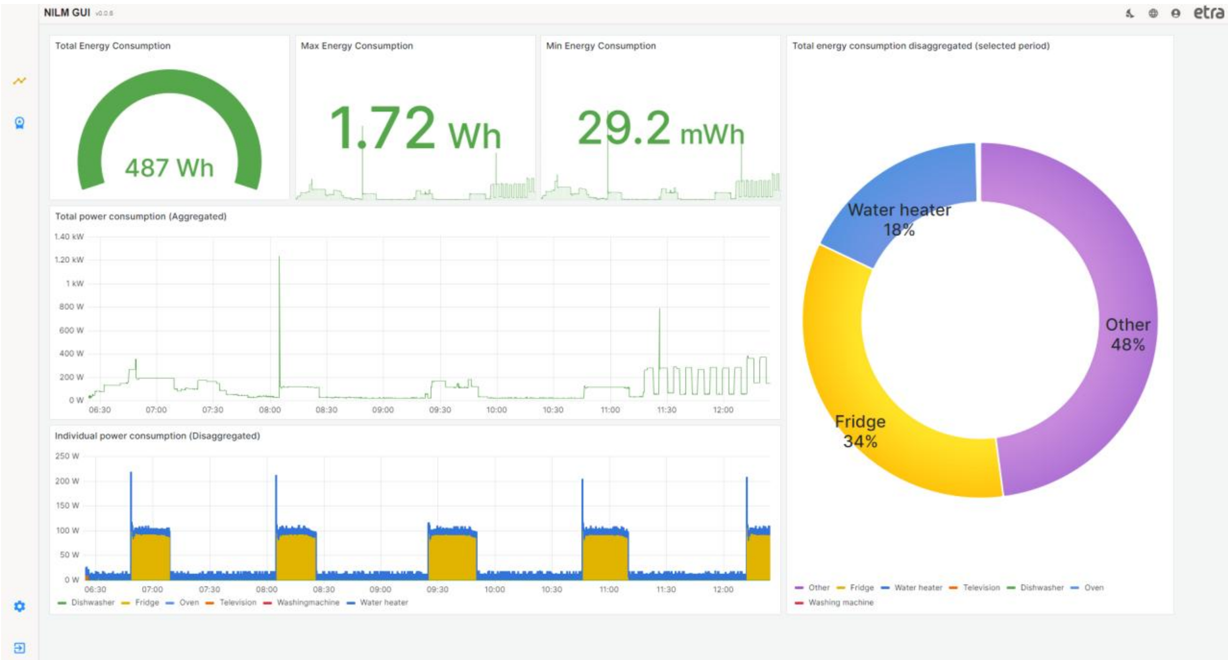


Figure 34: NILM graphical user interface, dashboard page.

Lastly, the energy certificates page is shown in Figure 35. Three energy certificates are already calculated, but to generate one, a series of inputs are needed from the end-user through the configuration tab and a link to the European Product Registry for Energy Labelling (EPREL) to find the corresponding appliance model and its characteristics. Once completed, the information is sent to the NILM Certificates API, which performs calculations using both theoretical data and real data from the household. Both theoretical and current energy labels, along with their corresponding Energy Efficiency Index (EEI), are displayed. The API uses the POST method with the appliance name, user ID, and corresponding characteristics in JSON format to create the certificate. The results are also sent in JSON format with the EEI and energy label, as seen in Figure 35.

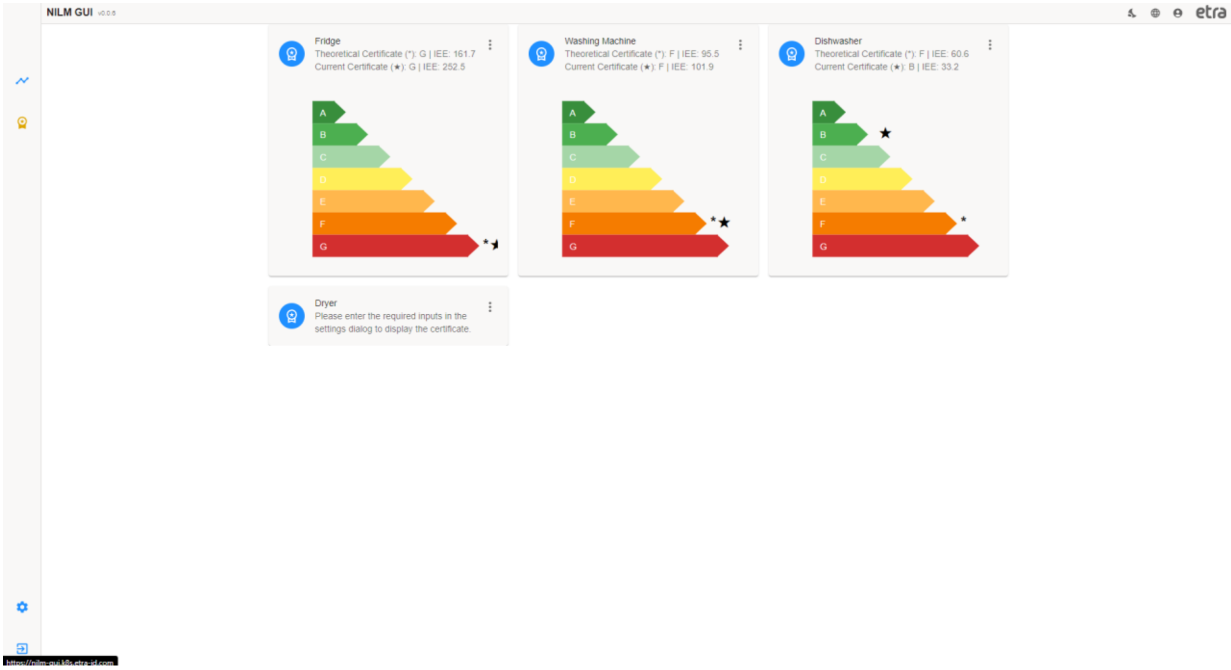


Figure 35: NILM graphical user interface, certificates page.

4.7 Application Example

This tool requires initial setup, including user creation and linking identifiers with data from smart meters. Once this is complete, credentials for the platform are generated, but initially, only total consumption data is displayed on the dashboard.

First, the user needs to select their household appliances in the settings menu, see Figure 36. This process can also be done during credential creation if the energy retailer provides the information.

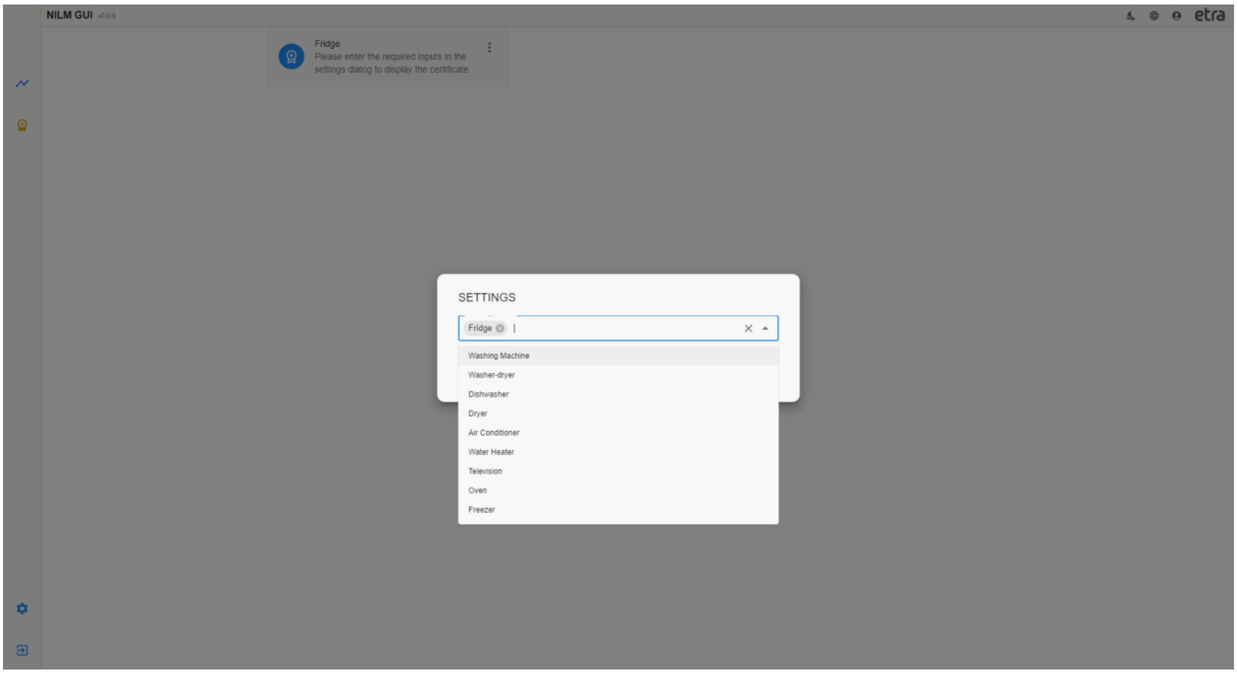


Figure 36: NILM graphical user interface, settings menu.

After listing all appliances, disaggregation begins, and the information appears on the dashboard, see Figure 34. The user can then navigate to the certificates page, see Figure 35, to create certificates by entering the requested information, see Figure 37. They can compare the theoretical value of the certificate with the actual one. Once a certificate is created, the information is stored and continuously updated. Appliance changes are considered, and the certificate can be recreated at any time.

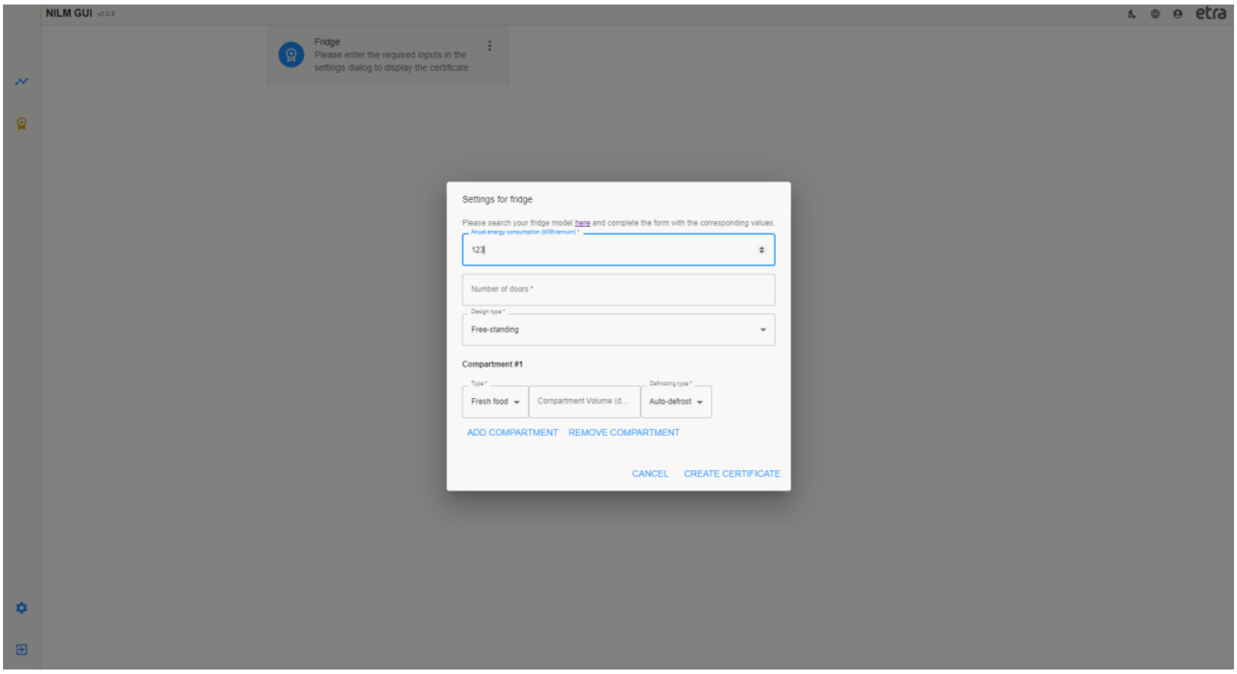


Figure 37: NILM graphical user interface, certificate settings.

4.8 Development & integration status

The development of the components is nearly complete, with everything functioning using simulated data. Refinement will occur once real-time data is integrated, but the components described are already operational. The main components that will undergo further refinement are the NILM GUI and the machine-learning models. Feedback from end users, gathered through Policy Labs conducted in other WPs, will be incorporated into the graphical interface, while real data from the pilots will be used to refine and re-train the machine-learning models for improved inference performance.

Specifically, for the second release of the tool in M34, the next roadmap, Figure 38, outlines the development and integration periods with pilots' data.

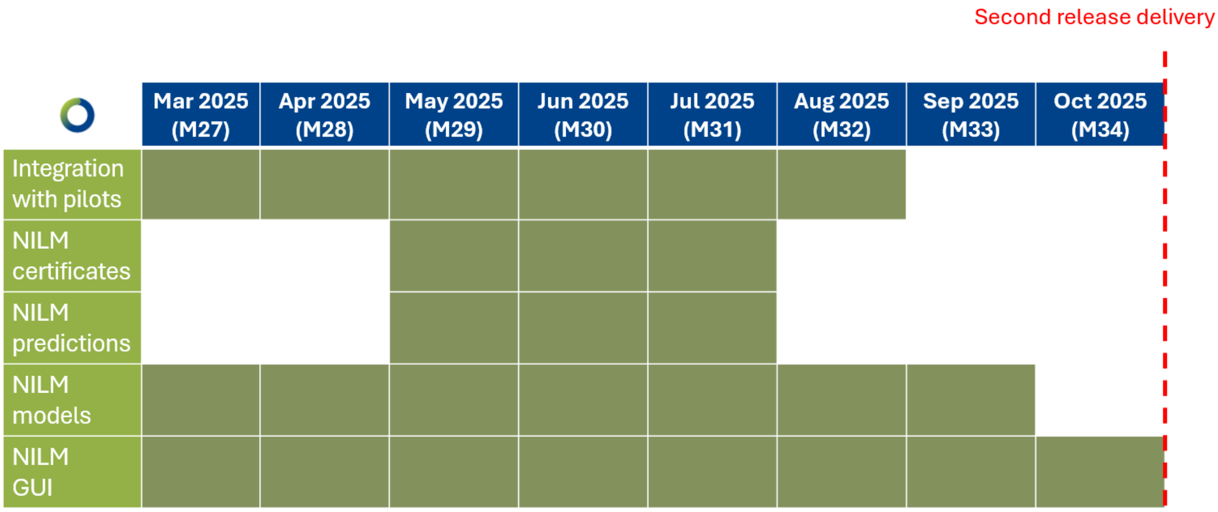


Figure 38: Development next steps roadmap.

Pilots' integration will occur in the initial months, necessitating the refinement of NILM AI models during and after this period. NILM certificates and predictions are nearly complete, with pending tasks such as the inclusion of new appliances and the creation of new models using different machine-learning libraries. Finally, the NILM GUI will be refined throughout the process, starting with transitioning visualizations from Grafana to the ETRA metrics library for enhanced visualizations, and continuously improving it based on feedback from pilots and end users.

4.9 Requirements in Equipment & Infrastructure

Intrusive Load Monitoring, as detailed in the methodology, is essential for evaluating and retraining machine-learning models. This process requires the installation of submeters within households to monitor appliances that contribute significantly to total energy consumption, such as dishwashers, washing machines, refrigerators, water heaters, and air conditioners. The main equipment requirements are:

- **Smart meters:** These devices monitor the total power consumption of the household with a minimum granularity of one minute, ensuring the recognition of typical appliance usage patterns.

- **Submeters:** These devices monitor the individual power consumption of selected appliances with the same granularity as smart meters, providing real data for retraining and evaluating machine-learning models.
- **Controllers:** No specific controllable devices are required for this task.

Regarding infrastructure, there are slight preferences for monitoring appliances that can be switched on or off at any time for flexibility market related use cases.

4.10 Assumptions and restrictions

The main restriction in applying NILM prediction components and the models developed so far is data granularity, as each appliance has a characteristic curve that defines it. If the granularity is insufficient, model performance will be compromised. Currently, the minimum granularity provided for the models is one minute, but increased granularity enhances AI model performance. As can be seen in Figure 39, the difference between both granularities lies in the amount of information given to the model explaining the consumption data, particularly visible at peaks. Therefore, the AI model is able to distinguish the appliances patterns.

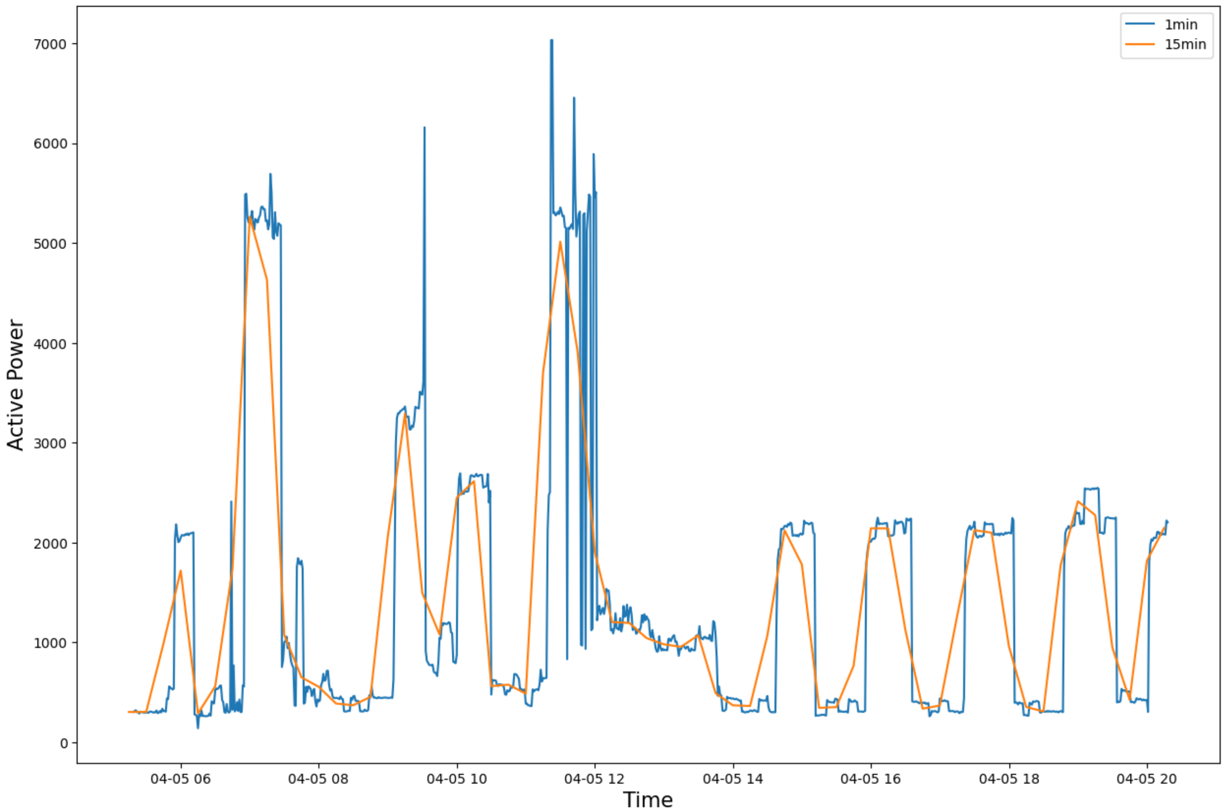


Figure 39. one minute vs fifteen-minute granularity example

4.11 Pre-validation activities & Preliminary tests

Validation activities and preliminary tests have been conducted, focusing on NILM machine-learning models to assess their predictive capabilities. Currently, no pilot data is available for use in evaluation metrics to validate the models, so publicly available data from the internet has been utilized for these purposes. Main results can be seen in Table 12 and Table 13.

Several public datasets have been examined, but only those with large samples exceeding one month of data, high granularity with a minimum of one-minute intervals, and minimum active power in their exogenous variables were selected. The chosen datasets include UK-DALE [21], the Almanac of Minutely Power dataset (AMP) [22], Electricity Consumption and Occupancy (ECO) [23], REFIT [24], ENERTALK [25], and DEDDIAG [26].

4.11.1 Data Processing & Training methods

Datasets have undergone a series of steps to ensure clean and clear information:

- **Resampling:** The datasets vary in granularity. Once all variables are ordered in the same pandas DataFrame, resampling functions are used to match the target granularity, which is one minute. The entire dataset is then divided into as many subsets as there are appliances and households.
- **Data pre-preprocessing and cleaning:** For each dataset, records with NaN and None values are removed. This process is applied to exogenous variables, primarily active and reactive power. The target variable, which is the appliance's individual active power, must also match these records and consider its own NaN and None values. An example is shown in Figure 40, using the ECO dataset.

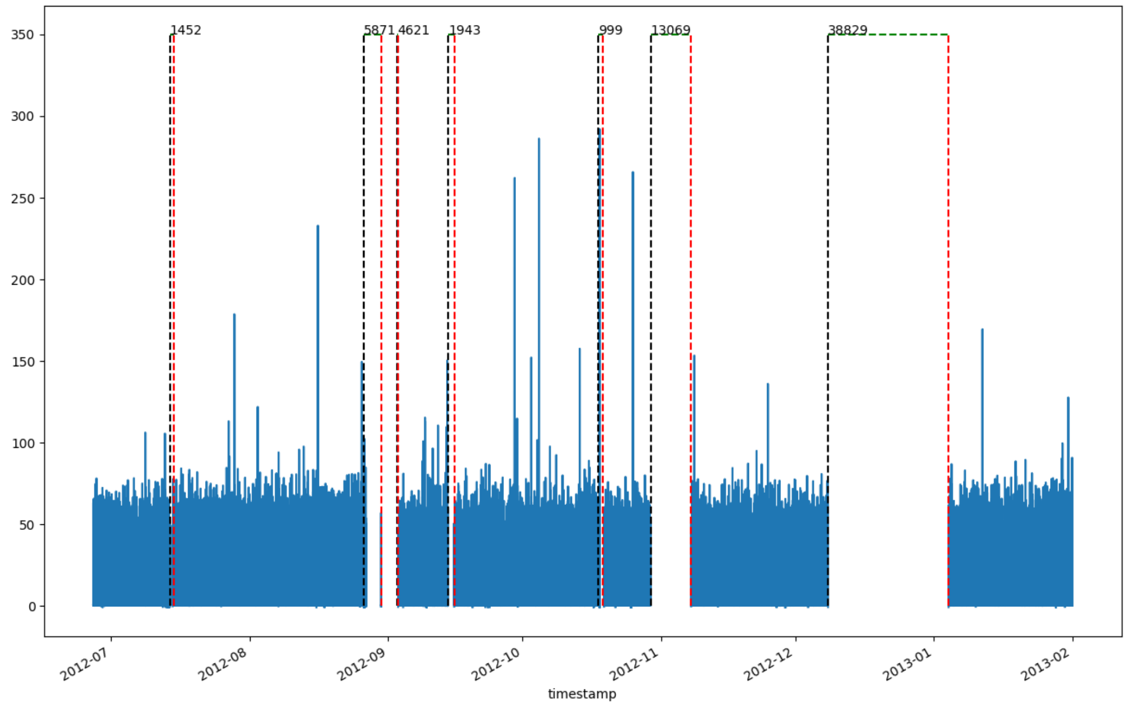


Figure 40: Example of ECO dataset NaN cleaning.

- **Feature extraction:** New variables are created to provide more immediate information to the models. Seasonal time variables such as day, hour, minute, weekday, month, and season are generated. Additionally, up to 60 lags are created for each record to provide the model with past values of total active power and recognize typical curves.

- **Train and test dataset creation:** At this stage, a large dataset for each appliance is created by collecting all the relevant household datasets. Each dataset is divided into 90% training and 10% testing. The years of data per appliance for datasets containing only active power and those with both active and reactive power are shown in Figure 41 and Figure 42.

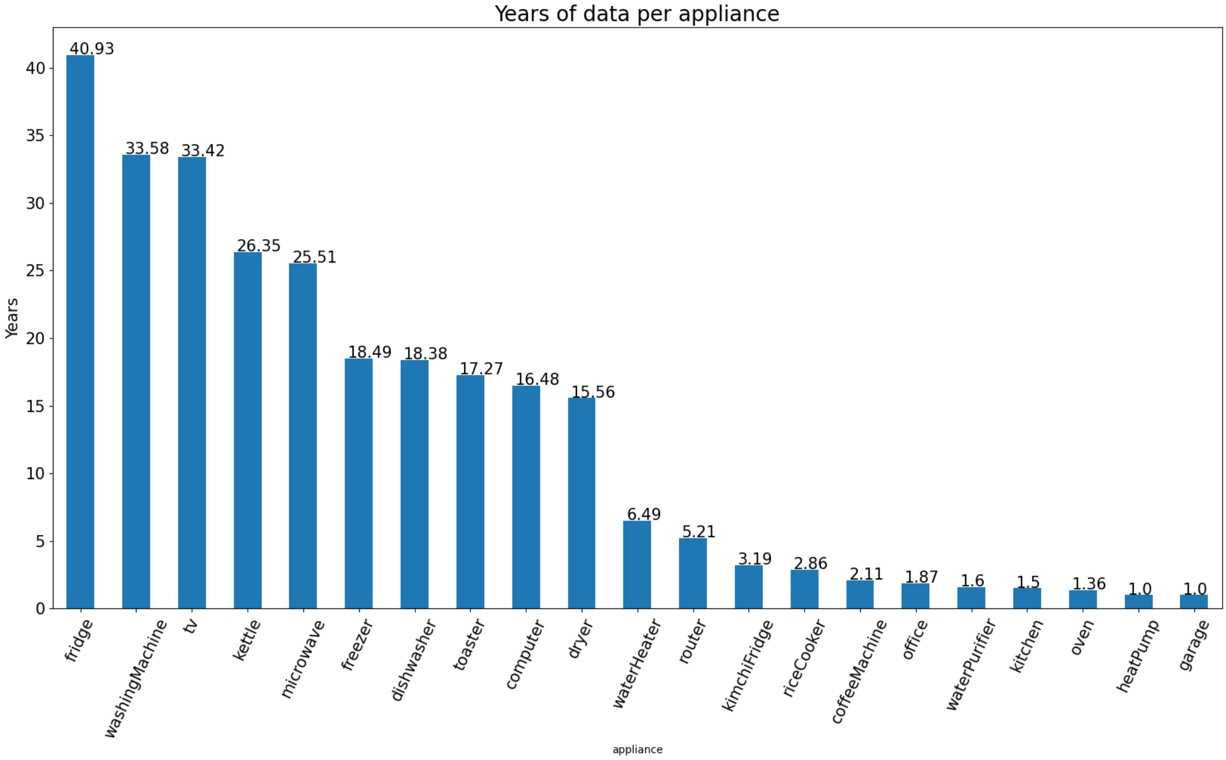


Figure 41: Years of data per appliance with only active power datasets

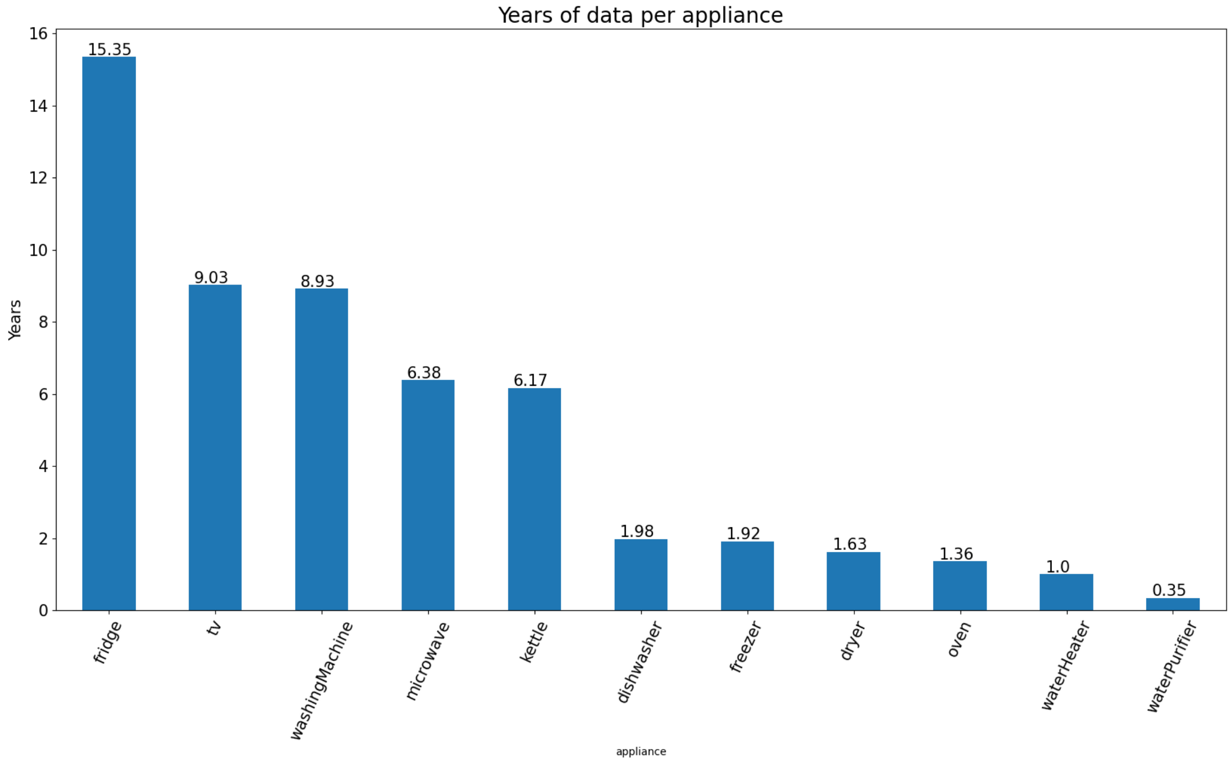


Figure 42: Years of data per appliance with active and reactive power datasets

Training methods can be divided into the following steps:

- **Data normalization:** To reduce complex model training calculations, datasets are normalized and scaled using the Standard Scaler:

$$z = \frac{(x - u)}{s}$$

  where *x* is the value, *u* is the mean of the column, and *s* is the standard deviation. Other scalers like MinMax Scaler were also tested, but performance did not change significantly.
- **Model training:** Using the scaled training dataset, machine learning models are created with Cross Validators, specifically BayesianSearchCV and RandomizedSearchCV. Cross Validation involves dividing the training dataset into several folds, training the model on some folds, and testing on others. The model with the best evaluation metric on each fold is considered to generalize best to different data. Grid Search methodologies are used to find the best-performing model hyperparameters. The desired intervals for hyperparameter tuning are selected, and optimizers find the best-evaluated hyperparameters through Random searches or more complex techniques like BayesianSearchCV. After sufficient training sessions, the best-performing parameters are identified. This process was tested with XGBoostRegressor models and TensorFlow neural network architectures.
- **Model evaluation:** The proposed hyperparameter search optimizers are evaluated on each fold using metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R^2. MAE was the primary metric used. These metrics are also applied to the test dataset. Once the models are created, their performance is tested against the test datasets, yielding the following results:

Table 12: NILM best models evaluation metrics results for each appliance using active power only

Appliance	R^2 (%)	MAE (W)	RMSE (W)
Washing Machine	0.54	19.76	110.07
Dishwasher	0.75	23.35	123.57
Dryer	0.73	33.83	144.93
Fridge	0.39	26.48	38.47
Freezer	0.50	27.40	41.61
Kettle	0.59	16.66	121.58
Oven	0.82	6.29	66.92
Microwave	0.23	11.60	76.27
Television	0.47	20.03	34.08
Water Heater	0.87	8.51	52.37
Router	0.62	1.18	2.83

Table 13: NILM best models evaluation metrics results for each appliance using active and reactive power

Appliance	R^2 (%)	MAE (W)	RMSE (W)
Washing Machine	0.89	9.31	54.00
Dishwasher	0.87	8.80	60.28
Dryer	0.98	3.70	48.69
Fridge	0.70	16.32	26.89
Freezer	0.91	13.03	25.58
Kettle	0.78	8.36	75.45
Oven	0.83	5.21	68.01
Microwave	0.78	5.12	40.90
Television	0.89	8.02	16.90
Water Heater	0.31	3.34	6.36

- **Model uploading:** The final step is to upload the trained and evaluated models to the Model serving platform MLflow, as shown in Figure 33. The metrics and exogenous variables are also registered with the model.

4.11.2 Tests & Results

Model testing and evaluation metrics are presented in Table 12 and Table 13. The main results indicate that reactive power is a useful variable for determining appliance behaviour, showing superior overall results. However, datasets containing both variables are limited, as shown in Figure 41 and Figure 42.

Preliminary conclusions suggest the necessity of high granularity and exogenous variables to incrementally enhance the model's performance. Generalization remains challenging in machine learning projects, requiring substantial amounts of detailed training and testing data on appliances. One potential opportunity for improving model performance is to re-train generalized models with specific household appliance electrical measurements, providing them occupant needs and behaviours. Alternatively, training specific machine-learning models for individual households during the sub-metering process, followed by continued monitoring without sub-metering using NILM, could be explored.

5. CONCLUSIONS

The deliverable presented the development of the BFMS and NILM components, as envisaged in Tasks 4.1-4.3 of the OPENTUNITY project. The two tools mark a significant advancement in the identification of the usage patterns for individual loads and energy flexibility calculation and dispatch.

At first, the HEMS/BEMS located at the pilot sites of the project were described, as the indispensable tools that provide the necessary data for the BFMS and NILM. The HEMS/BEMS are responsible for the monitoring and control of flexibility assets, the streaming of sensorial data and the overall recording of the energy consumption of a dwelling.

BFMS provides a robust mechanism for unlocking flexibility potential, enabling market interactions, and facilitating energy monetization. Based on power consumption data, it calculates day-ahead baseline and flexibility power consumption timeseries. The inclusion of DR Initialization Service facilitates DR participation for occupants, reducing barriers to engagement while ensuring that user comfort and constraints are respected.

In parallel, the NILM component brings state-of-the-art load identification and disaggregation capabilities to the OPENTUNITY project. By leveraging advanced AI-driven methodologies, NILM is able to provide precise energy usage insights that can be used to feed the BFMS, but also to identify usage patterns and optimize energy consumption.

Both tools, have been tested and demonstrated using either available data from the project (BFMS), or publicly available dataset (NILM). Ongoing and future efforts focus on the integration with the pilot sites of the project, as well as with other tools and services, such as FSPs and the NODES platform. As the pilot sites are in the process of getting connected to the two tools, more data will be made available in the coming months, providing more input for the use, validation and further exploitation of the BFMS and NILM.

6. References and acronyms

6.1 References

[1] "ACCEPT Project," [Online]. Available: https://www.accept-project.eu/.

[2] "ACCEPT D4.8 Building Digital Twin v2".

[3] M. Ester, H. P. Kriegel, J. Sander and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Second International Conference on Knowledge Discovery in Databases and Data Mining*, 1996.

[4] D. Arthur and S. Vassilvitski, "K-means++: The advantages of a careful seeding," in *18th annual ACM-SIAM symposium on discrete algorithms*, 2007.

[5] C. Yuan and H. Yang, "Research on K-Value Selection Method of K-Means Clustering Algorithm," *Multidisciplinary Scientific Journal,* 2019.

[6] T. John, I. Sarantakos and T. T. Teo, "Stacking different services of an energy storage system in a grid-connected microgrid," *Renewable Energy,* 2022.

[7] N. Zografou-Barredo, C. Patsios, I. Sarantakos, P. Davison, S. Walker and P. Taylor, "MicroGrid Resilience-Oriented Scheduling: A Robust MISOCP Model," *IEEE Transactions on Smart Grid,* 2021.

[8] T. Sayfutdinov, C. Patsios, D. Greenwood, M. Peker and I. Sarantakos, "Optimization-based modelling and game-theoretic framework for techno-economic analysis of demand-side flexibility: A real case study," *Applied Energy,* 2022.

[9] J. Revuelta Herrero, Á. Lozano Murciego, A. Barriuso, D. Hernández de la Iglesia, G. Villarrubia, J. Corchado Rodríguez and R. Carreira, "Non Intrusive Load Monitoring (NILM): A State of the Art.," pp. 125-138, 2018.

[10] M. Basseville, "Detecting changes in signals and systems—A survey," *Automatica,* vol. 24, pp. 309-326, 1988.

[11] S. M. a. I. V. B. M. Z. A. Bhotto, "Load Disaggregation Based on Aided Linear Integer Programming," *IEEE Transactions on Circuits and Systems II: Express Briefs,* vol. 64, no. 7, pp. 792-796, 2017.

[12] W. S. N. a. A. C. L. D. Srinivasan, "Neural-network-based signature recognition for harmonic source identification," *IEEE Transactions on Power Delivery,* vol. 21, no. 1, pp. 398-405, 2006.

[13] J. K. O. P. H. D. W. K. A. R. A. S. M. S. Nipun Batra, "NILMTK: An Open Source Toolkit for Non-intrusive Load Monitoring," *5th International Conference on Future Energy Systems (ACM e-Energy),* 2014.

[14] T. &. C. H. S. &. L. D. &. T. T. &. Y. T. Kato, "Appliance Recognition from Electric Current Signals for Information-Energy Integrated Network in Home Environments," vol. 5597, pp. 150-157, 2009.

[15] D. Kelly, "Disaggregation of domestic smart meter energy data," 2016.

[16] G. J. a. P. Henneaux, "Unsupervised learning procedure for NILM applications," *IEEE 20th Mediterranean Electrotechnical Conference (MELECON),* pp. 559-564, 2020.

[17] Y. a. W. J. a. D. J. a. S. W. a. T. P. Liu, "Non-Intrusive Load Monitoring Based on Unsupervised Optimization Enhanced Neural Network Deep Learning," *Frontiers in Energy Research,* vol. 9, pp. 718-916, 2021.

[18] F. H. M. A. M. M. A. Q. M. N. A. A. R. C. Hafiz Khurram Iqbal, "A critical review of state-of-the-art non-intrusive load monitoring datasets," *Electric Power Systems Research,* vol. 192, pp. 0378-7796, 2021.

[19] E. -. E. P. R. f. E. Labelling, "https://eprel.ec.europa.eu/screen/home," [Online].

[20] MLflow, "https://mlflow.org/docs/2.20.0rc0/python_api/index.html," [Online].

[21] U.-D.-2. dataset, "https://data.ukedc.rl.ac.uk/browse/edc/efficiency/residential/EnergyConsumption/Domestic/UK-DALE-2017/UK-DALE-FULL-disaggregated," [Online].

[22] A. dataset, "http://ampds.org/," [Online].

[23] E. C. &. O. dataset, "https://vs.inf.ethz.ch/res/show.html?what=eco-data," [Online].

[24] R. E. L. M. (Cleaned), "https://pureportal.strath.ac.uk/en/datasets/refit-electrical-load-measurements-cleaned," [Online].

[25] E. dataset, "https://github.com/ch-shin/ENERTALK-dataset," [Online].

[26] D. dataset, "https://figshare.com/articles/dataset/DEDDIAG_a_domestic_electricity_demand_dataset_of_individual_appliances_in_Germany/13615073," [Online].

6.2 Acronyms

Table 14. Acronyms

Acronym	Explanation
AC	Air condition
API	Automatic Programming Interface

BEMS	Building Energy Management System
BFMS	Building Flexibility Management System
DHW	Domestic Hot water
DR	Demand Response
ESS	energy storage system
EV	Electric Vehicle
EU	European Union
FSP	Flexibility Service Provider
GP	Gaussian Processes
HEMS	Home Energy Management System
HVAC	Heating Ventilation and Air Conditioning
IoT	Internet of Things
LV	Low Voltage
NILM	Non-Intrusive Load Monitoring
OCPP	Open Charge Point Protocol
PV	Photovoltaic
R&D	Research and Development
RF	Random Forest (RF)
SAX	Symbolic Aggregate Approximation (SAX
SO	System Operator
UI	User Interface
UC	Use Case
WP	Work Package
WSN	Wireless Sensor Network