Master Thesis

Doble Máster Universitario en Ingeniería Industrial e Ingeniería de la Energía

Integration of a microgrid laboratory into an aggregation platform and analysis of the potential for flexibility

MEMORY

Author: Maite Etxandi Santolaya
Director: Cristina Corchero García

Call: June 2021

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Abstract

Due to the increase of renewable energy sources in the electricity grid, demand-side flexibility, lead by Demand Response (DR), is gaining momentum to counteract the uncertainties of the new electricity system and contribute to grid balancing. At the same time, consumers are becoming active participants by engaging in flexibility actions enabled by demand aggregators.

This project aims to integrate a microgrid laboratory with an aggregation platform in order to set up and configure the necessary tools to operate the laboratory as a platform to test flexibility. This way, the potential flexibility services that consumers can provide to the grid has been analysed. With the use of virtual, real and emulated elements from the laboratory, a more realistic impact and value of DR programs can be quantified. Two types of consumers have been defined: a residential one (Scenario 1), with a Heating Ventilation and Air Conditioning (HVAC) unit and a prosumer (Scenario 2), owning a battery in a solar Photovoltaic (PV) self-consumption system. For both scenarios, the effect of receiving flexibility activations from the aggregator has been analysed and compared with a base case in which no interaction with the aggregator occurs. In addition, OpenADR, a relevant protocol for DR has been implemented and tested in the laboratory (Scenario 3).

The experimental work developed shows that demand-side flexibility can play a big role in the current and future electricity grid, as customers, demand aggregators and grid operators can benefit from these actions.

From an operational point of view, the tests in the laboratory showed that the HVAC and the battery were able to follow the activations received, as long as the commands were properly calculated by the aggregator. For the Scenario 1, the aggregator modified the temperature setpoint of the HVAC, causing a shift of the consumption to different time periods. In the Scenario 2, the battery charge/discharge power setpoint was modified by the aggregator in order to reduce the electricity consumption from the grid or inject power into it when necessary.

From the customer's perspective, we saw that the DR actions can increase the energy cost in some cases, highlighting the importance of economic incentives to attract customer engagement. In addition, in the Scenario 1, involving the HVAC system, the thermal comfort of the users was not affected by the presence of the aggregator, as the indoor temperatures were maintained in an adequate range. Finally, allowing the aggregator to control the battery in Scenario 2 did not have any effect on the self consumption factor of the user.

Regarding OpenADR, a test was developed first virtually and then between the laboratory and the aggregator in Scenario 3. OpenADR proved to be a reliable and useful protocol that has the potential to be used in DR applications due to the different use cases it can cover.

Key words: demand-side flexibility, microgrid laboratory, Demand Response, OpenADR, aggregation platform

Table of Contents

ABSTRACT....2

1 INTRODUCTION....10
1.1 Aims and objectives....11
1.2 Document overview....12
1.3 Related work....12
1.4 Overview of demand-side flexibility....13
1.4.1 Flexibility potential of HVACs....16
1.4.2 Flexibility potential of second-life batteries....16
1.4.3 Demand Aggregators....16
1.4.4 Overview of OpenADR....17

2 IREC'S SMARTLAB DESCRIPTION....20
2.1 Facility description....20
2.2 Electrical scheme....20
2.2.1 Real elements....21
2.2.2 Emulation cabinets....23
2.3 Communication scheme....24
2.3.1 Emulator – LC communication....25
2.3.2 LC – SCADA Communication....26
2.3.3 Upper layer systems....26
2.4 SCADA description....26
2.4.1 Screens of the SCADA....27
2.4.1.1 Initial screen....27
2.4.1.2 Configurations....27
2.4.1.3 Profile configurations....28
2.4.1.4 Monitoring....28
2.4.1.5 Demo....29
2.4.1.6 Alarms....29
2.4.1.7 History....29
2.5 Aggregation platform....30

3 LABORATORY SET-UP....32
3.1 SCADA related work....32
3.1.1 ITME software description....32
3.1.2 Demo creation....33
3.1.2.1 Residential Demo....35
3.1.2.2 Prosumer Demo....36
3.1.2.3 OpenADR Demo....37
3.1.2.4 Demo simulations....38
3.2 Virtual elements: HVAC....38
3.2.1 Thermal zone model....39
3.2.2 HVAC control....39

3.2.3 Virtual HVAC diagram....41
3.3 Emulated elements....42
3.4 Real elements: Battery....43
3.4.1 Battery control....43
3.5 Integration with the aggregation platform....47
3.5.1 Microgrid definition....48
3.5.2 Sending data....48
3.5.3 Reading commands....49

4 SCENARIO DEVELOPMENT....50
4.1 Scenario 1: Residential consumer....52
4.1.1 Profile selection....53
4.1.2 Base case - summer....54
4.1.3 Base case - winter....57
4.1.4 Case with aggregator - summer....59
4.1.5 Case with aggregator - winter....62
4.1.6 Flexibility potential of the residential consumer....65
4.2 Scenario 2: Prosumer....66
4.2.1 Power profiles....67
4.2.2 Base case....67
4.2.3 Case with aggregator....70
4.2.4 Flexibility potential of the prosumer....75

5 OPENADR INTEGRATION....77
5.1 Communication between VTNs and VENs....77
5.2 Events in OpenADR....79
5.3 Implementation using openleadr....81
5.3.1 OpenADR Client....81
5.3.2 OpenADR Server....81
5.4 OpenADR communication tests....82
5.4.1 Simple events....84
5.4.2 CPP events....87
5.4.3 DLC events....90
5.5 OpenADR implementation in the laboratory....93
5.5.1 SmartLab VEN definition....93
5.5.2 OpenADR test in the laboratory....94
5.5.3 Aggregator API vs OpenADR comparison....96

6 TIME PLANNING....98

7 ENVIRONMENTAL AND ECONOMIC ANALYSIS....99
7.1 Environmental Analysis....99
7.2 Economic Analysis....100

8 CONCLUSIONS....101

9 BIBLIOGRAPHY....105

List of figures

Figure 1: Traditional vs new electricity grids [4]....12
Figure 2: Overview of the project....13
Figure 3: Implicit and explicit DR [18]....15
Figure 4: Status of DR in Europe [19]....15
Figure 5: Curtailable and shiftable loads [18]....16
Figure 6. Demand Aggregators for flexibility services [9]....18
Figure 7: OpenADR implementation overview [36]....19
Figure 8: SmartLab facilities....21
Figure 9. SmartLab electrical scheme....22
Figure 10: Converters of the SAFT battery / ultracaps....23
Figure 11: Second-life battery set-up....23
Figure 12: Second-life battery inverter....24
Figure 13: Emulation cabinet converters....24
Figure 14: Communication lines of the SmartLab....25
Figure 15: Three levels of communication of the SmartLab....26
Figure 16. Initial screen....28
Figure 17: Configuration screen....29
Figure 18: Monitoring screen....30
Figure 19. Alarms screen....30
Figure 20. History screen....31
Figure 21: Demo Screen....34
Figure 22: HVAC and building selection subscreens....35
Figure 23: Battery selection subscreen....35
Figure 24: Tasks of residential Demo....36
Figure 25: Tasks of prosumer Demo....37
Figure 26: Tasks of OpenADR Demo....38
Figure 27: One zone RC model....40
Figure 28. Flux diagram of the HVAC Control....41
Figure 29: HVAC control states....42
Figure 30: Virtual HVAC diagram....43
Figure 31: Battery reduction....45
Figure 32: Main battery control flux diagram....47
Figure 33: Auxiliary control flux diagram....48
Figure 34: Residential consumer diagram....54
Figure 35: Load profiles for the residential scenario (summer and winter)....55
Figure 36: Meteo profiles for the residential scenario (summer and winter)....55
Figure 37: HVAC Power and temperature evolution (residential base case, summer)....56
Figure 38: Meter power consumption (residential base case, summer)....57
Figure 39: HVAC power consumption (residential base case, winter)....58
Figure 40: Meter power consumption (residential base case, winter)....59

Figure 41: HVAC and indoor temperature evolution (residential summer, aggregator).....60
Figure 42: Difference in power consumption r/ base case (residential summer).....62
Figure 43: Reduction of power during events (residential, summer).....62
Figure 44: HVAC power consumption (residential winter, aggregator).....64
Figure 45: Difference in power consumption r/ base case (residential winter).....65
Figure 46: Reduction of power during events (residential winter).....65
Figure 47: Prosumer diagram.....67
Figure 48: PV Production and load profile for the prosumer scenario.....68
Figure 49: Prosumer base case results.....69
Figure 50: Battery setpoint vs real one.....69
Figure 51: Exported and imported electricity (prosumer, base case).....71
Figure 52: Prosumer battery power and SoC (aggregator case).....72
Figure 53: Prosumer Day 3 Zoom Morning.....73
Figure 54: Prosumer Day 3 Zoom midday.....74
Figure 55: Prosumer meter power comparison.....75
Figure 56: Reduction of power during events (prosumer).....75
Figure 57. OpenADR VTN-VEN interaction.....79
Figure 58: Times of an event in OpenADR [36].....81
Figure 59. OpenADR communication test.....83
Figure 60: Simple event timeline.....86
Figure 61. VTN Simple event output.....86
Figure 62. VEN Simple event output.....87
Figure 63. CPP event timeline.....89
Figure 64: VTN CPP test output.....89
Figure 65: VEN CPP test outputs.....90
Figure 66: DLC event timeline.....92
Figure 67: VEN DLC event output.....92
Figure 68: VTN DLC event output.....93
Figure 69: Laboratory OpenADR implementation.....94
Figure 70: OpenADR Lab test results.....96
Figure 71: Event logger for the OpenADR lab test.....97

List of tables

Table 1. Parameters of the storage devices....21
Table 2. Type codes....26
Table 3: Summary of the project scenarios....33
Table 4. Parameters of the HVAC model....38
Table 5: Emulation cabinet connection....42
Table 6: Main SCADA tags for LC 4 and 7 connection....42
Table 7: Battery connection....43
Table 8: Main SCADA tags for LC 9 connection....43
Table 9: Criteria for signs....43
Table 10: Battery parameter definition....44
Table 11: Variables of the battery control....45
Table 12: Configurable time variables of the battery control....45
Table 13: Possible states of the battery....47
Table 14. Overview of the developed scenarios....50
Table 15: Electricity tariff....51
Table 16: KPIs used for the scenarios....52
Table 17: Residential scenario characteristics....53
Table 18: Indoor temperature outputs (residential base case, summer)....56
Table 19: Residential scenario consumption (base case, summer)....57
Table 20: Indoor temperature output (residential base case, winter)....58
Table 21: Residential scenario results (base case, winter)....59
Table 22: Residential summer case events....59
Table 23: Indoor temperature output (residential summer, aggregator)....60
Table 24: Residential scenario results (summer, aggregator)....62
Table 25: Residential winter case events....62
Table 26: Indoor temperature output (residential winter, aggregator)....63
Table 27: Residential scenario results (winter, aggregator)....65
Table 28: KPIs for the residential scenario....65
Table 29: Prosumer elements (1)....66
Table 30: Prosumer elements (2)....66
Table 31: Prosumer exported and imported power (base case)....69
Table 32: Prosumer case events....71
Table 33: Prosumer electricity cost and consumption (aggregator)....75
Table 34: KPIs for the prosumer scenario....75
Table 35: OpenADR message list....77
Table 36. OpenADR standard event signals....80
Table 37. OpenADR test characteristics....83
Table 38. Simple event features....84
Table 39. Simple event test results....84
Table 40. Expected and obtained results of the simple test....87

Table 41. CPP event features....87
Table 42. CPP test results....88
Table 43. Expected and obtained results of the CPP test....89
Table 44. DLC event features....90
Table 45. DLC event test results....90
Table 46. Expected and obtained results of the DLC test....93
Table 47: OpenADR laboratory characteristics....94
Table 48: OpenADR lab test characteristics....94
Table 49: OpenADR lab test event description....95
Table 50: Task description and duration....98
Table 51: Project dedication according to ECTS credits....98
Table 52: Real grid consumption for the laboratory scenarios....99
Table 53: Total CO2 equivalent emissions of the project....99
Table 54: Electricity tariff considered for the project....100
Table 55: Total cost of execution of the project....100

Glossary

Abbreviation	Term
AC	Alternating Current
AFE	Active Front End
API	Application Programming Interfaces
CAN	Controller Area Network
COP	Coefficient of Performance
CPP	Critical Peak Pricing
CW	Command Word
DA	Demand Aggregator
DC	Direct Current
DER	Distributed Energy Resource
DLC	Direct Load Control
DR	Demand Response
DSO	Distribution System Operator
EMS	Energy Management System
EV	Electric vehicle
HMI	Human Machine Interface
HTTP	Hypertext Transfer Protocol
HVAC	Heating Ventilation and Air Conditioning
IP	Internet Protocol
IREC	Catalonia Institute for Energy Research

Abbreviation	Term
ITME	InTouch Machine Edition
TLS	Transport Layer Security
LC	Local Controller
TSO	Transmission System Operator
OSI	Open Systems Interconnection
OpenADR	Open Automated Demand Response
TOU	Time-of-use
PLL	Phase-Locked Loop
PV	Photovoltaic
RES	Renewable Energy Sources
SCADA	Supervisory Control and Data Acquisition
SoC	State of Charge
SW	Status Word
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UTC	Universal Time Coordinated
VEN	Virtual End Node
VTN	Virtual Top Node
XML	Extensible Markup Language
XMPP	Extensible Messaging and Presence Protocol

1 Introduction

Concerns about the environmental impact of the energy consumption are accelerating the transition towards a sustainable electricity system. The decarbonisation of the energy sector, aimed to achieve the climate goals, is closely linked to an increasing presence of Renewable Energy Sources (RES) and the electrification of the demand [1]. These actions, although necessary, can also generate reliability issues to the power system that need to be addressed for a successful energy transition.

The reliability of a power system depends on its ability to accommodate expected and unexpected changes (in generation and demand) while maintaining quality and security of service to the customers [2]. This ability is known as system *flexibility.* A lack of flexibility in a power system can lead to negative effects such as difficulty balancing demand and supply, renewable energy curtailments or price volatility [3].

Traditional electricity systems consist of generation, transmission and distribution, with customers at the end of the chain. In this structure, generation is in hands of large power plants who, by order of the system operators, are responsible to balance the changes in demand [4].

The increase of RES poses new challenges for Transmission System Operators (TSOs) and Distribution System Operators (DSOs). Although some RES provide flexibility, like geothermal, hydropower, bioenergy, and some concentrating solar power plants [5], this is not the case for all. Wind or Solar Photovoltaic (PV) have an inherently intermittent and stochastic character caused by their dependency on weather conditions. Due to this, they cannot be used as an element of flexibility by the system operator and other strategies must be explored for maintaining the demand-generation equilibrium, also known as grid balancing [6].

In addition to large renewable plants, the grid is also evolving towards one where microgrids, formed by distributed energy resources (DERs) such as rooftop solar photovoltaic (PV) installations, battery energy storage systems, electric vehicles (EVs) and smart home appliances, are growing and becoming active in the operation of the system [4]. Figure 1 captures the evolution of the grid from the traditional to the modern one.

In this new context, customers can take a central role in the electricity grid, which allows to increase the flexibility of the system by using demand-side management actions. Demand, that was often perceived as relatively inelastic, can be used to balance the uncertainty and variability introduced by RES [7]. Thanks to the Internet of Things (IoT) and new actors, like aggregators, demand can be controlled and customers can actively participate in the electricity grid [8] reducing their electricity bill and contributing towards a successful energy transition.

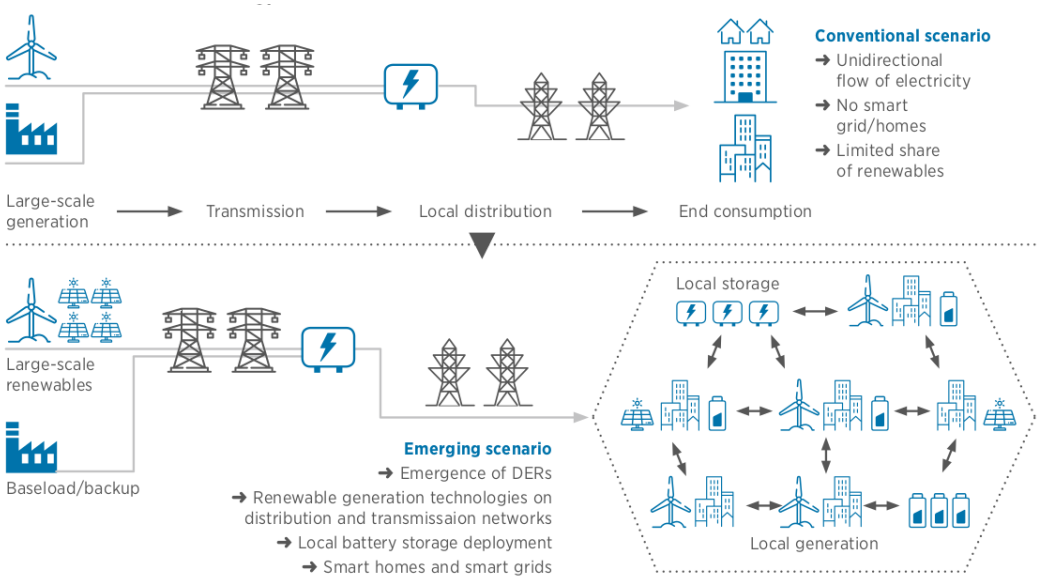

Figure 1: Traditional vs new electricity grids [4].

Thus, the use of demand aggregators for controlling the customer's consumption provides a tool that can benefit both customers and system operators [9]. Exploring these new alternatives and their impact on the users has been the main motivation for this project. The microgrid laboratory of the Catalonia Institute for Energy Research (IREC), Energy SmartLab, has been used to analyse the participation of different consumers in Demand Response (DR) events. Rather than a theoretical or simulated approach, by using the elements in the laboratory, more realistic scenarios have been carried out to study the flexibility potential of the defined customers.

1.1 Aims and objectives

The aim of this thesis is to integrate IREC's Energy SmartLab, from now on SmartLab, within the *Bamboo Energy* aggregation platform in order to develop a tool in the laboratory to test the potential of microgrids to offer flexibility services. The specific objectives of this project are:

- Adapt the Supervisory Control and Data Acquisition (SCADA) system of the SmartLab to allow the integration with the aggregation platform.
- Define and implement the communication between the aggregation platform and the SmartLab by using the aggregation platform API.
- Define assets in the laboratory that can provide flexibility, objective achieved by:
  - Defining a virtual model to represent a HVAC system for a thermal zone.
  - Connecting the emulation cabinets and one real element (the second-life battery) to the SCADA to use them for test development.
  - Defining a control model for the battery in a self-consumption system.
- Define, develop, and analyse several scenarios to test the response of the SmartLab when receiving DR events.
- Implement and test a key communication protocol in DR Programs, OpenADR.

An overview of the project schematic can be seen in Figure 2.

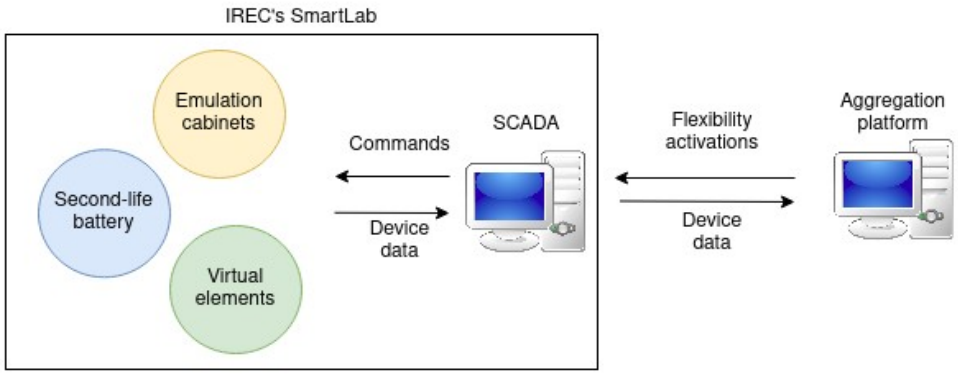

Figure 2: Overview of the project

1.2 Document overview

To meet the previously mentioned objectives, this document is divided as follows.

- The remaining of Chapter 1 includes a literature review of similar work developed in other projects as well as some theoretical explanations of demand-side flexibility, microgrids, demand aggregators and OpenADR.
- Chapter 2 includes the description of the SmartLab.
- In Chapter 3 the different tasks that were carried out to develop the scenarios are presented. This includes, the SCADA Demo screen definition, the virtual HVAC definition, the battery control definition, the connection of the laboratory elements and the integration with the aggregation platform.
- Chapter 4 presents all the scenarios developed in this project, including their results.
- The testing of OpenADR is presented in Chapter 5.
- Chapter 6 includes the temporal planning of the project.
- Chapter 7 presents the environmental and economic impact of the project.
- Finally, in Chapter 8 the conclusions extracted from the project are presented along with the lessons learnt and the potential future work.

1.3 Related work

The work developed in this project takes place in a microgrid laboratory. Microgrids, as defined by the US Department of Energy, are *"a group of interconnected loads and distributed energy resources with clearly defined electrical boundaries that acts as a single controllable entity with respect to the grid and can connect and disconnect from the grid to enable it to operate in both grid-connected or island modes" [10]*. Microgrids have the ability of self-supply and grid islanding and are able to locally generate, distribute, and regulate the electricity to customers. Thus, improved reliability, resiliency, energy efficiency and power quality are some of the most significant features of microgrids [11].

Besides the previously mentioned benefits, there is also growing interest in exploring the possibilities of demand-side management in microgrids. Extensive work has been performed to quantify their flexibility potential. These studies provide high value to understand how customers and different assets can provide flexibility. However, as most of them are based on simulations, they also represent ideal situations without considering factors of uncertainty that occur in real life (metering errors, communication delays…).

Microgrid laboratories allow to define more realistic scenarios as there is real power flowing during the tests, unlike during pure simulations. At the same time, the laboratory environment allows to create different scenarios without depending on external factors that are uncontrollable in real life, like weather conditions. Thus, flexibility tests in a laboratory represent a valuable step before the development of real-life applications.

Several projects have used experimental approaches to study flexibility. Most of these experiments are field or pilot trials and only a few take place in laboratories. As an example of a laboratory project, the authors in [12] developed several scenarios to analyse the economic impact of incentive-based Demand Response (DR) Programs. A hardware emulation approach was used to represent a typical residential consumer including a 4 kVA load and a 2 kW PV system. However, in this study, the loads were controlled by an Energy Management System (EMS) which has access to pricing information, while in our approach they are controlled by the aggregator.

After reviewing the existing literature, we can see that very few projects have used laboratory conditions to study DR actions controlled by an aggregator. Due to the difficulty of developing scenarios in real life, experiments in the laboratory offer a powerful tool to increase the knowledge in the field. Following a previous master thesis where the SCADA was implemented [13], by integrating IREC's laboratory with the Bamboo Energy aggregation platform, this project sets the basis for new studies. Using the work developed, further testing in the laboratory is expected. These experiments will allow to study the interaction between the aggregator and different customer profiles and DR programs in a controlled yet realistic environment.

1.4 Overview of demand-side flexibility

The IEA (International Energy Agency) considers demand-side flexibility as a key actor to arrive to meet the sustainable development goals. However, further efforts need to be made as less than 2% of the global potential for demand-side flexibility is currently in use [14]. As a tool for making use of flexibility, Demand Response (DR) refers to all the mechanisms which allow a modification of the customers energy consumption according to the power system needs [15].

DR programs are usually categorized into two groups: incentive-based or explicit DR and price-based or implicit DR. Through explicit DR, customers are paid in exchange of load reductions over a specific time. This type of DR usually includes a demand aggregator that then trades the

flexibility in different markets. In the other hand, through implicit DR customers can voluntarily reduce their load influenced by changing electricity prices [16]. The pricing can be based on a Time-of-use (TOU) tariff, in which, depending on the period of year and hour, different rates apply. In other cases, real time pricing can be used, where the price changes on short term to reflect the variability of the markets [17]. Figure 3 shows both types of DR programs [18].

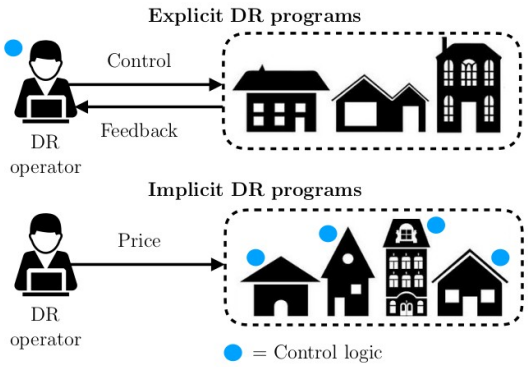

Figure 3: Implicit and explicit DR [18]

Europe is experiencing a growth in the presence of DR. The Energy Efficiency Directive (2012/27/EU) represents a clear step towards development of DR in Europe. The articles 15.4 and 15.8 of this Directive encourage demand-side resources to participate alongside supply in wholesale and retail markets, in a non-discriminatory manner. The Directive requires that regulators, TSOs and DSOs, define the technical modalities and requirements for DR providers, including aggregators, to participate in these markets [19]. While some countries are still behind in the incorporation of these new guidelines, some others have been working to enable and strengthen the role of DR. A summary of the status of each EU member, developed by [20] in 2020, can be seen in Figure 4.

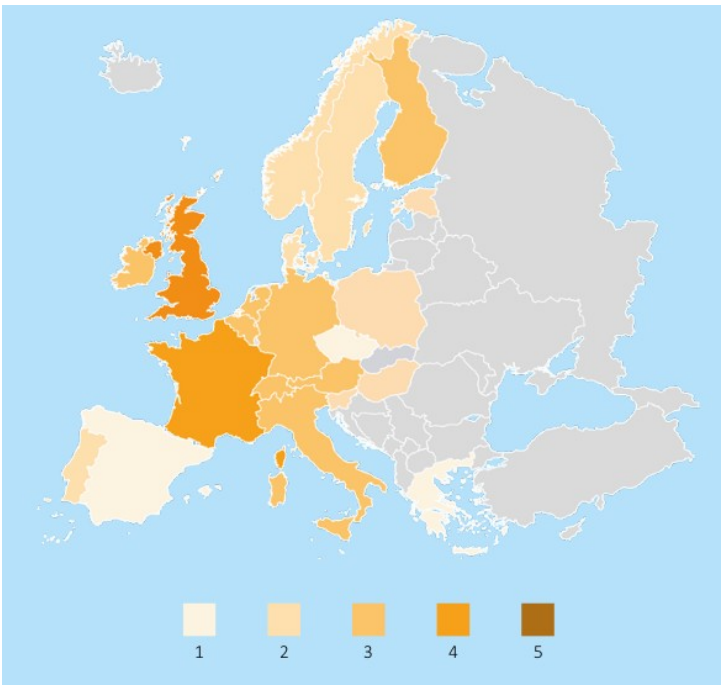

Figure 4: Status of DR in Europe [19]. With 1 being emerging markets and 5 the most matured ones.

According to the study, the most matured markets are France and Great Britain, followed by Ireland, Germany and the Netherlands. In these countries, flexibility markets are competitive and accessible for aggregated assets. Other countries like Spain are considered emerging as limited or no value stream has been defined for demand-side flexibility. In between these groups we find active markets which still present some important barriers to the adoption of demand-side flexibility. However, markets are rapidly changing. Spain, for example, has recently opened up balancing services to the demand, as included in the BOE (Boletín Oficial del Estado) [21].

Many customers can extract benefit from offering flexibility, from industrial, residential or commercial to EV owners [17]. The types of assets that can enrol into flexibility actions can be divided into two groups, as shown in Figure 5:

- Shiftable loads: devices that allow to shift their consumption to other periods of the day. Some of these loads can be modified without creating a loss of comfort in the user. This is the case of smart appliances like dishwashers, washing machines or dryers. In this group we also find electric batteries and other forms of thermal energy storage units. There are other shiftable loads that have the potential to create a loss of comfort for the user, like HVAC units or EVs [22]. For example, reducing the power consumed by a HVAC in the winter will reduce the indoor temperature and if not controlled adequately this temperature can fall below the comfort range.

- Curtailable or interruptible loads: in this group we find loads that can be curtailed or completely interrupted. In some cases industrial loads can be part of this category. When the activity allows to do so, non-critical load curtailment and temporary shutdown of industrial processes can provide a large modification of the demand [23]. Other loads under this category are also electric batteries [22].

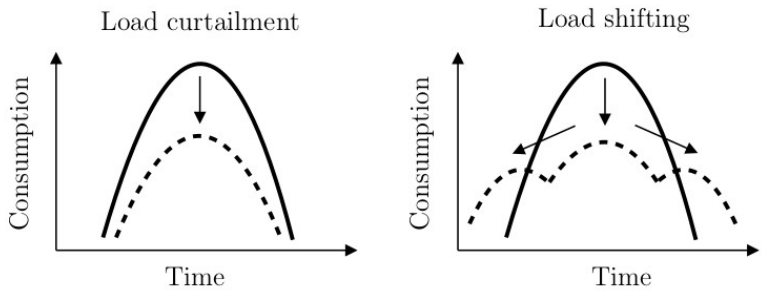

Figure 5: Curtailable and shiftable loads [18]

Each load type provides different features and levels of flexibility in DR actions. In order to explore different cases, the flexible loads available in the SmartLab and chosen for this project consist of two types: a HVAC unit and a second-life battery in a PV self consumption system.

1.4.1 Flexibility potential of HVACs

According to [24], heating and cooling accounts for over 60% of the total household consumption. In addition, these demands tend to occur when the overall electricity demand is already high [25]. For these reasons, the participation of heating and cooling systems in DR programs can have a high impact on the grid.

HVAC systems are a group of components working together to provide ventilation and to heat or cool a conditioned space. HVACs contain all the necessary elements for heating or cooling the air, to filter it and distribute it [26]. These systems are a form of thermostatically controllable load, meaning that their energy consumption is directly affected by the temperature setpoint at the thermostat. For DR applications, this means that by controlling the setpoint, the power output of the HVAC can be modified according to the needs of the grid. However, when changing the temperature setpoint, the indoor temperature of the building is modified, which may compromise the comfort of the occupants [22]. Nevertheless, with an adequate control scheme and by taking advantage of the thermal inertia of the building, the impact on the users can be minimized [25].

1.4.2 Flexibility potential of second-life batteries

Because of the high cost of new batteries, giving a second life to the retired ones from EVs, comes as an interesting option to reduce cost and improve the environmental impact of EVs [27]. Due to both cycling and calendar ageing, EV batteries reach End-of-life (EoL) when their capacity is reduced by 20-30%. At this moment it is considered that they are no longer suited for their original purpose in the EV [28]. However, as the batteries still hold an important amount of capacity at this point, there is a growing interest in exploring second-life alternatives for them. One of these options is their use for residential purposes along with a PV system. Nevertheless, this scenario, as [29] suggests, would only provide economic benefit when coupled with demand aggregators to participate in DR programs.

The use of electric batteries as elements of flexibility allows to change the demand profile, without any impact on user habits or comfort. When constraints arise in the grid, the battery can be discharged to reduce the electricity being consumed from the grid, or even act as a generator and inject electricity into it. The charging strategy can also be controlled to absorb power whenever there is a surplus of generation in the grid [22].

1.4.3 Demand Aggregators

Recently, a new figure has gained importance as an additional tool to unlock the demand-side flexibility potential: Demand Aggregators (DA). DAs act as mediators between the users and the utility operator. As the name suggests, the role of a DA is to aggregate all the loads belonging to the consumers with whom it has agreements [30]. DAs have opened the possibility to end-

consumers, such as residential households or industries, to successfully participate in DR Programs. Without this new agent, a single customer cannot offer flexibility services. This is caused, on one hand, due to the technical requirements, such as the minimum bid size, that exist in the electricity markets and on the other hand, because of the complexity of market participation (forecasting, bidding strategies…). By combining the demands, each aggregator represents a significant amount of total demand in the DR market and also possesses the required tools to participate in the market [31].

The DR market can be modelled in three levels. On the upper one, grid operators create programs and provide monetary rewards to aggregators that offer DR services. The aggregators are located in the middle level. They offer a total aggregated demand profile to grid operators which minimizes the cost of the operator to support it. Then, by offering incentives to customers they can achieve this demand profile, while trying to maximize their profit. On the lowest level, users provide the needed flexibility service and get an economic reward in exchange of it [32]. The DR flexibility chain is shown in Figure 6.

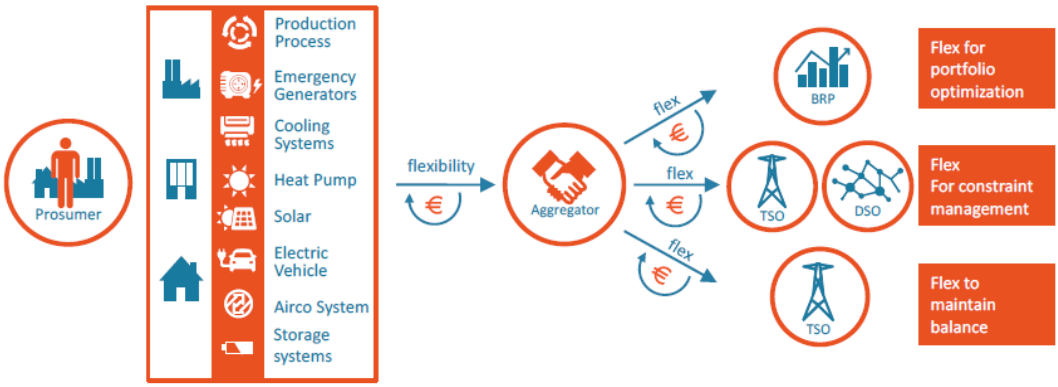

Figure 6. Demand Aggregators for flexibility services [9]

DAs face different challenges in their interaction with utilities and customers. In the bottom level, they seek to perform DR programs with minimum costs, while dealing with customer constraints and the uncertain response from them when offered incentives. In the upper-level, their goal is to optimize their trading options in the different markets [33]. To achieve all these targets, aggregators need tools like load forecasting and bidding optimization strategies [30].

When allowed by national regulations, aggregators can participate in different markets in order to make profit out of the demand-side management. One of the possibilities are national ancillary and flexibility markets, but also local ones created to manage grid congestions that may appear [34].

1.4.4 Overview of OpenADR

One of the goals of this project is to explore what is meant to be a dominant protocol in Demand Response: OpenADR. OpenADR, being an open protocol, allows to standardize the communications and ease the integration of DR. At the end of 2018, the OpenADR 2.0

standard was established as an IEC standard and has grown since then to over 150 members with certified products from all over the world [35]. It is therefore interesting to explore its use for the different stakeholders of DR. In this section, a brief overview of the protocol is presented. All the information on the OpenADR protocol has been extracted from the documentation (profile specification and program implementation guide) found on the official website [36]. The version considered in this section is OpenADR 2.0, as the previous 1.0 was created for validation.

Open Automated Demand Response (OpenADR) is an open communications data model, along with transport and security mechanisms, used to exchange information between two endpoints in Automated DR programs. This standard has been developed by the OpenADR Alliance, an entity created to standardize, automate, and simplify DR programs and help integration and management of DERs. OpenADR is designed to facilitate automated electric load shedding or shifting, as well as to provide continuous dynamic price signals. OpenADR has been gaining presence in Automated DR programs due to the following benefits:

- As its name suggests, it offers an open and non-proprietary standard with protocols that are flexible, platform-independent, interoperable, and transparent.
- It offers ease of integration as it facilitates integration of common Energy Management and Control Systems, centralized lighting, and other end-use devices that can receive Internet signals (such as XML).
- The standard describes a complete data model and architecture to communicate price, reliability and other flexibility activation signals.
- It ensures secure communication thanks to its security mechanism.

The protocol allows a two-way communication between a server or VTN (Virtual Top Node) and a client or VEN (Virtual End Node). In any communication there exists one VTN and one or more VENs, as shown in Figure 7.

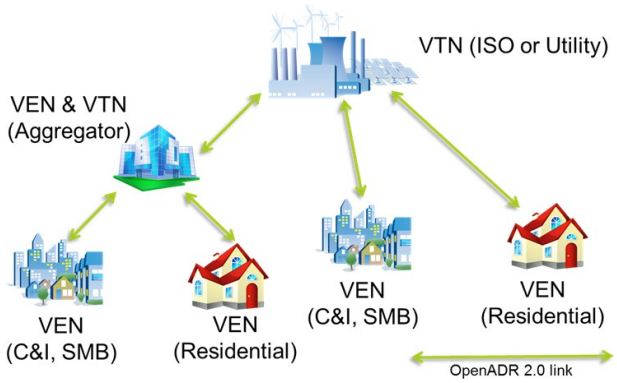

Figure 7: OpenADR implementation overview [36]

The VTN is the responsible for communicating grid conditions (prices, constraints…) to the VENs that control resources. The VEN receives these signals and communicates back with the VTN. Examples of VTNs include energy utilities, while VENs could be the Energy Management System (EMS) of a building with controllable loads. In some cases, the same entity can take the

role of a VTN and VEN. This is the case for aggregation platforms which act as VENs when receiving signals from utilities and as VTNs when communicating with the customers.

A “resource” is the entity enrolled in the DR Program that is able to provide flexibility in response to a DR signal coming from a VTN. A resource could be a single customer load behind a meter, a combination of various customers under the control of an aggregator or a specific element like a thermostat. Behind a VEN there might be one or more resources. An “asset” is a specific physical load that forms a resource and that is managed by the VEN or a control system behind the VEN. A VTN, such as a utility, cannot directly interact with the assets themselves. Instead, it communicates with the resource and might give further suggestions on the type of assets that could be used for the DR.

Currently, the OpenADR 2.0 data model offers the following services:

- Registration (*EiRegisterParty*): allows registration of the VENs and VTNs.
- Event (*EiEvent*): this is the core of the DR action and allows to call for performance under a transaction.
- Reporting or Feedback (*EiReport*): the state of a resource can be reported through periodic or one-time only information.
- Opt or Override (*EiOpt*): allows the VEN to communicate and define the availability to participate in events. After receiving an event the VEN can respond with an “optOut” message to show unavailability or “optIn” to engage in the event.

In relation to the transportation model defined by the protocol, VTNs must support both HTTP and XMPP. Depending on the VEN, simple HTTP or XMPP might be used. The service is defined by the XML payloads.

There are two patterns of interaction between the VTNs and VENs, PUSH or PULL. In the PUSH mode, the VEN must have a HTTP server, while this requirement is not needed in the PULL mode. When in PUSH mode, the VTN sends the messages to the VEN that may or may not respond. In the PULL functioning, the communication must be initiated by the VEN that constantly polls the VTN for information.

Two levels of security are defined in the protocol: standard, which is mandatory, and high level, which is optional. In the standard case, Transport Layer Security (TLS) is used to establish secure communication channels between VTNs and VENs. The high security adds XML signatures as an additional feature.

Finally, two types of profiles can be implemented: 2.0a and 2.0b. The A one is targeted to basic DR services that do not require complex mechanisms. The B profile suits most DR services as it includes flexible reporting (feedback) mechanism for past, current and future data reports.

As can be seen in the Certified Products of the website [36], the most common implementation of OpenADR is for 2.0b VENs, standard security, HTTP transport and PULL mode.

2 IREC's SmartLab description

2.1 Facility description

IREC's SmartLab, shown in Figure 8, is a flexible and innovative laboratory used for technology development and testing. By combining both real and emulated elements, the SmartLab, working up to 200 kVA, can be configured to provide a wide range of testing scenarios. The SmartLab is used for research in a large variety of areas, including the following [37]:

- Power Electronics for the integration and control of the elements within a building or community: Renewable Energy Sources (RES), Energy Storage Systems and Electric Vehicles (EVs).
- ICT Platform for smart communications and energy management of systems, building, networks and communities.

Figure 8: SmartLab facilities

To emulate different elements, the SmartLab has several emulation cabinets, each of which can be configured to work as an energy generator unit (e.g. Solar PV), an energy storage device or an energy consumption node. Besides the emulation cabinets, real storage systems, wind turbine emulators and EV charging points are present in the laboratory. Finally, there is a grid emulator allowing the study of different grid perturbations and configurations.

In terms of the control, each cabinet and storage system has a Local Controller (LC) and on a higher level, there is a Supervisory Control And Data Acquisition (SCADA). Finally, the aggregator that has been integrated in this project can act as the highest level of control.

2.2 Electrical scheme

A general scheme of the SmartLab is shown in Figure 9. The laboratory microgrid can be directly connected to the grid or to the grid emulator device. This emulator is a 200 kVA

configurable voltage source that allows to simulate different network conditions like weak grids or frequency alterations.

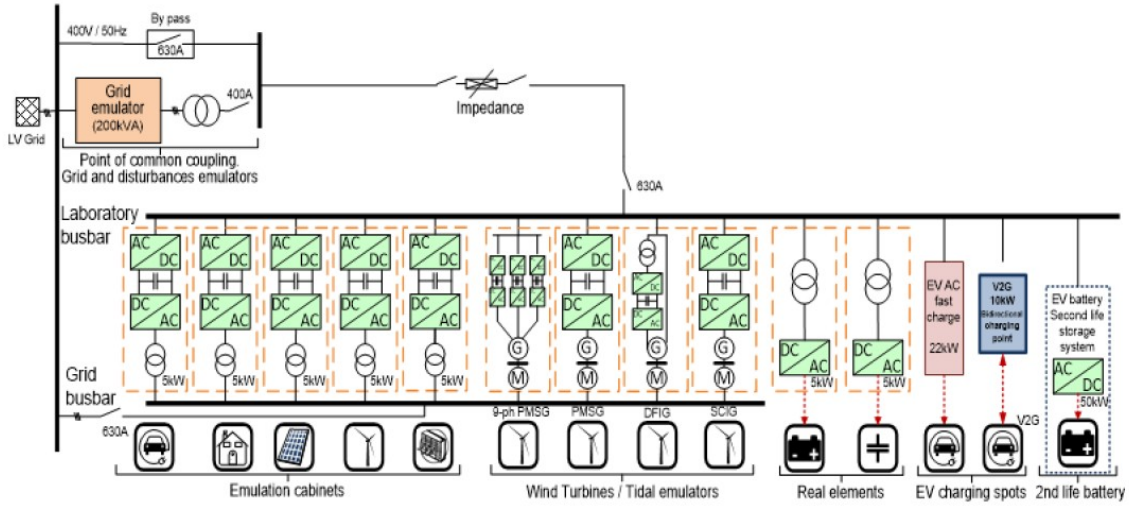

Figure 9. SmartLab electrical scheme

All the different elements available in the SmartLab are connected in parallel to the laboratory busbar. Out of these elements, the emulation cabinets and real elements, in particular the second-life battery, are of interest for the current project.

2.2.1 Real elements

The real elements currently in use in the laboratory are electrochemical energy storage devices, particularly, an ultracapacitor, two lithium-ion batteries and a second-life lithium-ion battery. To cycle these batteries, the laboratory also has a REGATRON battery tester. The main parameters of these elements are shown in Table 1. In addition, for self-consumption application testing, the laboratory has a setup that consists of several SMA inverters.

Element	Capacity	Power	Op. voltage
SAFT battery	20 000 Wh	37,5 kW (discharge)	189 – 278 V
2nd life battery	23 330 Wh	10 kW (charge)	240 – 398 V
Ultracaps	57 Wh	119 kW (discharge, 1s)	250 – 595 V
BYD battery	10 500 Wh	6,14 kW (charge)/ 9,22 kW (discharge)	40-59,2 V

Table 1. Parameters of the storage devices

The connection of the SAFT battery and the ultracapacitors to the grid is done through an AC/DC and a DC/DC converter. The DC/DC converter is used as a step-up converter up to DC 700 V which is the DC bus voltage. This DC voltage is then converted to the operating AC 400 V 3-phase line voltage of the microgrid through the AC/DC Voltage Source Converter. This set-up is shown in Figure 10. The rest of the elements have their own inverter.

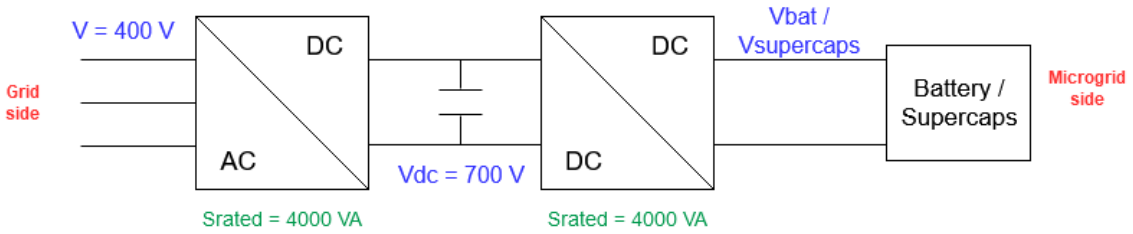

Figure 10: Converters of the SAFT battery / ultracaps

The second-life battery used for the current project belonged to a Renault Kangoo EV. After its service in the vehicle, the battery is now used in the laboratory as a static storage system. Due to the ageing of the battery, its current capacity is lower than the one in its beginning of life (shown in Table 1).

The battery pack is also equipped with a Battery Management System. This electronic system is aware of the internal state of the battery and is able to output that data via CAN Bus. In the same way, it is also able to receive information from the exterior through the same bus and communicate the maximum power that the battery is able to charge or discharge. The battery has a ABB 40 kVA inverter. The battery set-up and the schematic of the inverter are shown in Figure 11 and Figure 12 respectively.

Figure 11: Second-life battery set-up

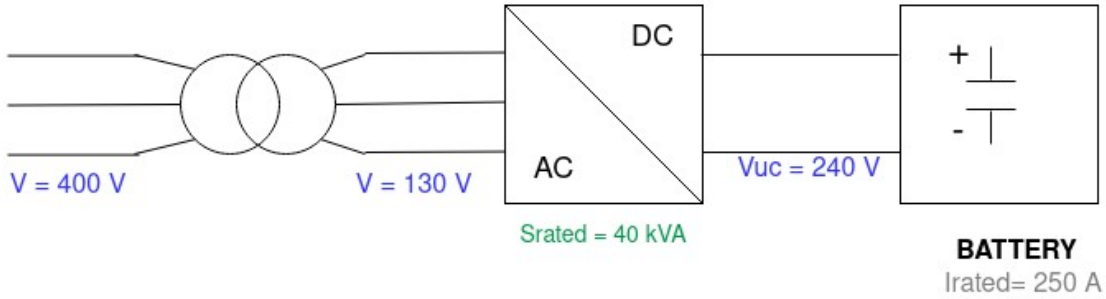

Figure 12: Second-life battery inverter

2.2.2 Emulation cabinets

Inside IREC's laboratory there are 5 B2B-uX-4kW emulation cabinets. The emulation is performed via hardware and allows to analyse scenarios that depend on phenomena that is uncontrollable in real life, like weather fluctuations [38]. Furthermore, with the emulation approach, there is real power flowing during an experiment, making it more realistic than a simulation.

Each unit is composed of two AC/DC converters in back-to-back configuration. These converters can work as active rectifiers or active inverters, allowing a bidirectional power flow. This way, the cabinets can act as generating or consuming nodes. Figure 13 shows both converters of the emulation cabinets in the back-to-back connection. The converters' operating voltage is 400V and they are connected through a DC bus of 700V.

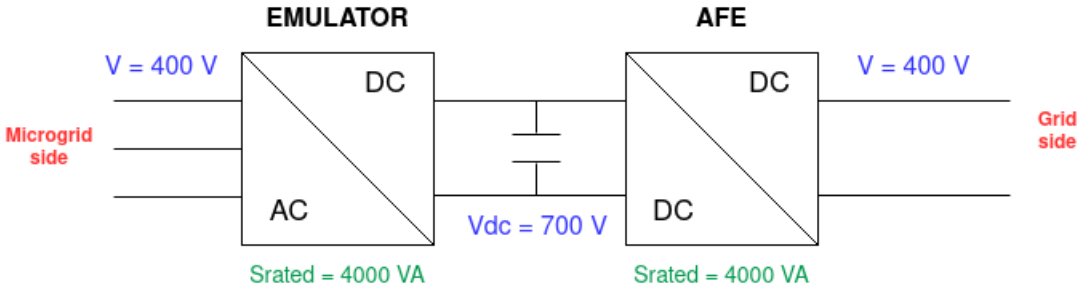

Figure 13: Emulation cabinet converters

The converter on the microgrid side, also known as emulator, is programmed to control the maximum active power by imposing a voltage on the DC bus [39]. The second converter is known as Active Front End (AFE) and is located in the grid side. This converter receives start/stop messages, active power and reactive power reference values, and it is requested information about status and measurements [38]. The AFE can inject or consume power from the grid, following the reference value received. When acting as a generator, the emulator drives power from the grid to the DC bus and when acting as a load, it drives power from the DC bus to the grid [40].

2.3 Communication scheme

The laboratory has four communication lines with different purposes as shown in Figure 14.

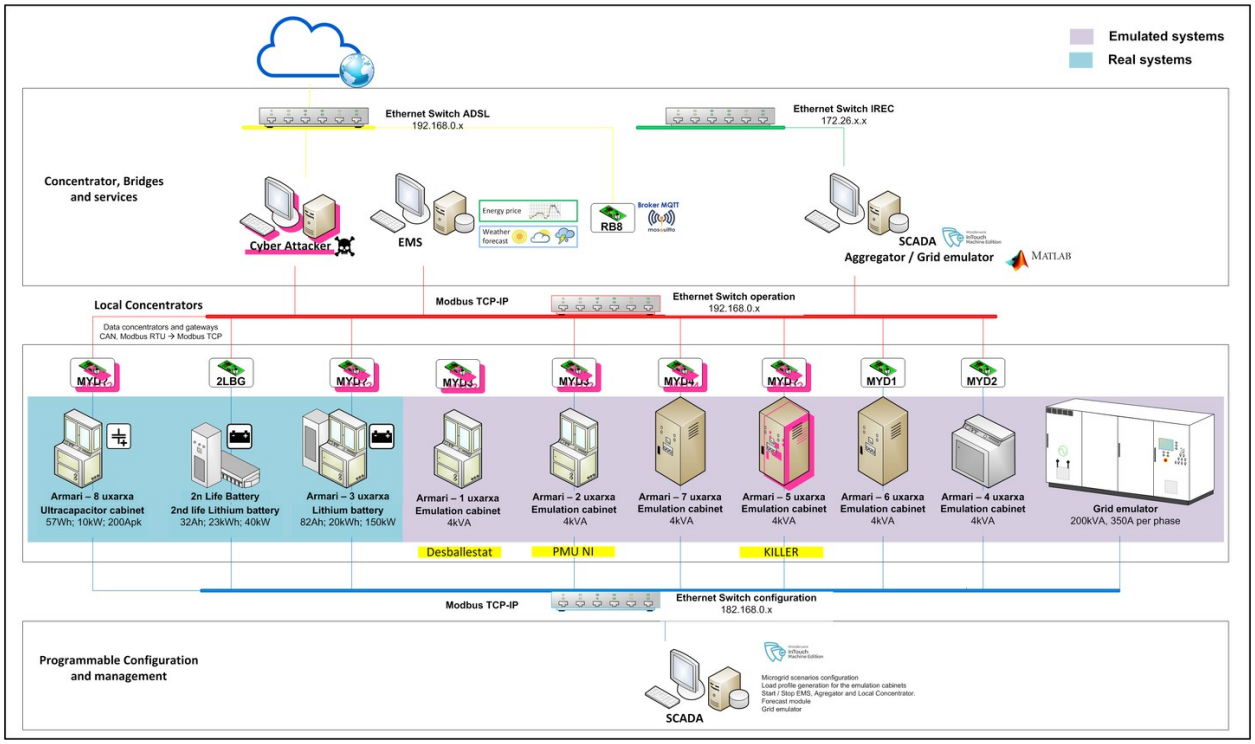

Figure 14: Communication lines of the SmartLab

- Configuration: this line is used to configure the laboratory elements from the SCADA.
- Operation: this line is used while the scenarios are running to control the elements, store information to the database, monitor the development of the scenarios etc.
- IREC: the SCADA is connected to IRECs communication line which is a corporative line where no laboratory equipment is connected and that has allowed to configure the remote desktop connection.
- External: only the SCADA computer and the EMS, are connected to an external line owing to the need of real-time information from the Internet. The line allows to access the laboratory elements externally.

The communication between the different elements in the SmartLab happens hierarchically in three levels, as can be seen in Figure 15.

- Bottom layer: the power converters.
- Middle layer: the LCs
- Upper layer: high-level control and management systems, such as the SCADA and the aggregation platform.

The communication between the upper and the middle layer are executed via Modbus TCP/IP over Ethernet cables. Whereas CAN is used between the middle and the bottom layer. The communication with the aggregator is done through the aggregator API which currently uses the InfluxDB database, further discussed in Chapter 2.5.

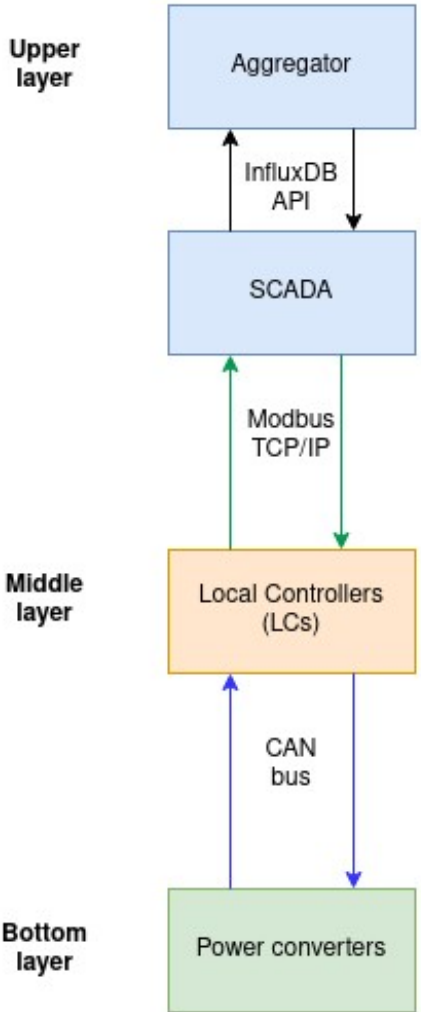

Figure 15: Three levels of communication of the SmartLab

2.3.1 Emulator – LC communication

The communication between the bottom layer and the middle layer (emulator and LC) is done via CAN bus. The emulation cabinets communicate with the LCs via CAN bus to send measurement data along with the state of the converter. This state is sent via the Status Word (SW). The LCs are Raspberry Pi 3 units, equipped with GNU/Linux and programmed in C++, which are normally placed on the top of the cabinet they are connected to. The LCs, via CAN bus, send the power setpoints to the converters and change their state through the Command Word (CW). The possible values for the SW and CW can be found in Annex B.

The LCs have two modes of functioning: in manual mode, they only switch the power setpoint

upon a specific command. If they are in auto mode, then setpoints are sent every second.

The type code described in Table 2 allows to identify the type of the element (both real and emulations) through an integer number. Type codes below 10 represent real elements and the ones above 20 represent emulated ones.

ELEMENT	TYPE	TYPE CODE
Battery	Real	1
Second-life battery	Real	2
Ultracapacitor	Real	3
Residence	Emulated	20
Solar Generator	Emulated	21
Wind Turbine	Emulated	22
Battery	Emulated	23
Tracking tester	Emulated	30

Table 2. Type codes

2.3.2 LC – SCADA Communication

The configuration of the LCs allows two modes of functioning: controllable or not controllable. When the emulation cabinets only follow the profile loaded in the LC, they are in not controllable mode. If external control can be applied from Modbus, they are in controllable mode. Only type codes 21, 22 and 23 allow the not controllable mode.

The full Modbus mapping (address, field, description and SCADA references) can be found in Annex A. All the addresses shown correspond to holding registers and thus start at address 4000.

2.3.3 Upper layer systems

The two main control systems of the laboratory are located in the upper layer: Firstly, the SCADA, explained in section 2.4, is responsible for directly communicating with the LCs. Then, the aggregator, integrated in the current project, acts as the highest level of control, communicating DR related information to the SCADA (more about it on section 2.5).

2.4 SCADA description

IREC implemented the current SCADA system in the SmartLab in the year 2018 through Benedek Paztor's Master's Thesis [13]. The main function of the SCADA is to centralize all the elements of the laboratory into a single computer from which different control and supervision tasks can be completed. The computer on which the SCADA runs can be accessed locally in

the SmartLab and from a remote desktop access that has been recently configured. With the use of the SCADA, the following tasks can be carried out:

- Generate and edit emulation scenarios by configuring the desired set-up and power profiles and by choosing between manual or auto mode.
- Running previously defined scenarios.
- Control and monitor all the laboratory elements, which include emulation cabinets and real elements. This means being able to stop or start the elements when running a scenario and change their setpoints.
- Access to historical data of the main variables of the scenario.
- Access to the alarm panel that shows a record of the incidents.

2.4.1 Screens of the SCADA

When accessing the SCADA different screens can be opened which are formed by a top bar, the navigator panel and a main screen that is characteristic of each screen. The following sections describe each of the screens that form the SCADA.

2.4.1.1 Initial screen

The initial screen (Figure 16) is configured to be the first to be opened when running the SCADA. The main screen simply shows a picture of the IREC facilities.

Figure 16. Initial screen

2.4.1.2 Configurations

The configurations screen (Figure 17) is used to define the laboratory set-up and shows the laboratory environment. Save and Load buttons in the top left part allow saving the current configuration and loading previously saved ones. On and Off buttons allow to choose which elements are going to be used. When an X sign appears on an element it means that the

connection has failed. In the lower part of the screen the alarm record is also accessible.

For all elements Manual or Auto mode can be defined as well as Control or No control mode. In the case of emulation cabinets, the type and profile can be selected. The maximum and minimum active and reactive power for each chosen element is also shown.

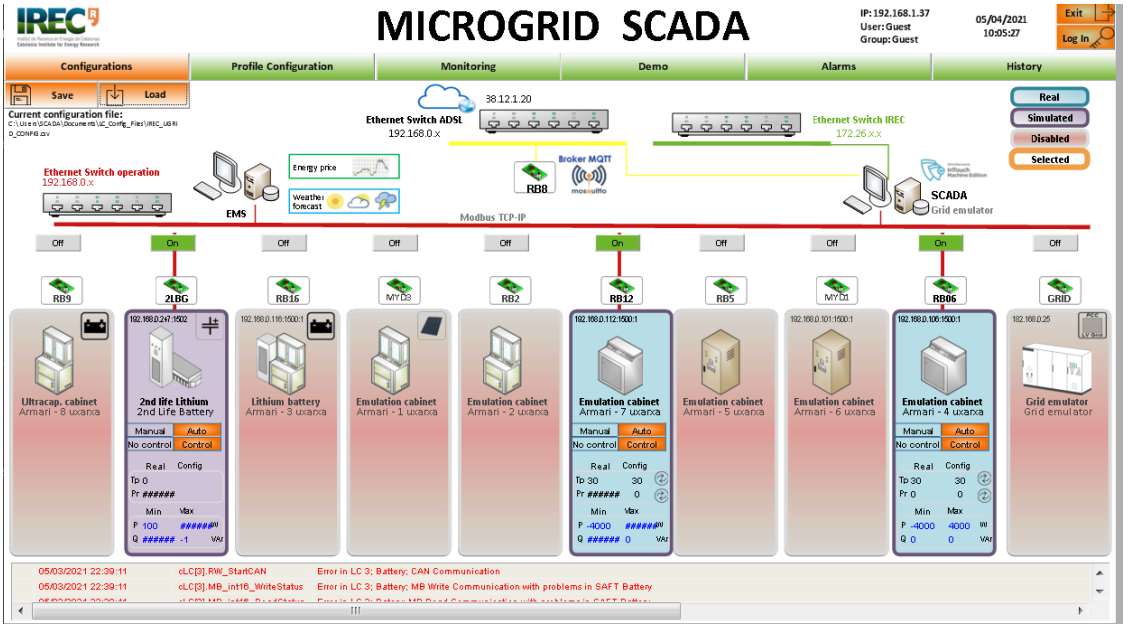

Figure 17: Configuration screen

2.4.1.3 Profile configurations

The profile configuration screen, allows the modification of the power profiles. The user first chooses the element type and then can check all the available profiles. If desired, these profiles can be modified on the excel file. At the time of the project, this option was not available and the power setpoints were sent periodically to the LCs.

2.4.1.4 Monitoring

This screen (Figure 18) allows general monitoring of the laboratory by sending messages to the LCs. For each element, the AC and DC voltage and current are shown along with the actual and set active and reactive powers. The current Status Word (SW) of each LC is also visible.

To control the LCs, the possible Command Words (CW) messages in the case of manual mode, and the start/stop buttons in case of the auto mode are accessible. There is also an emergency stop of the equipment that can be activated at any moment.

For more detailed information on the state of an element, the “Click for more” button can be pressed on top of each element. This button activates a pop-up window containing all the data of the element.

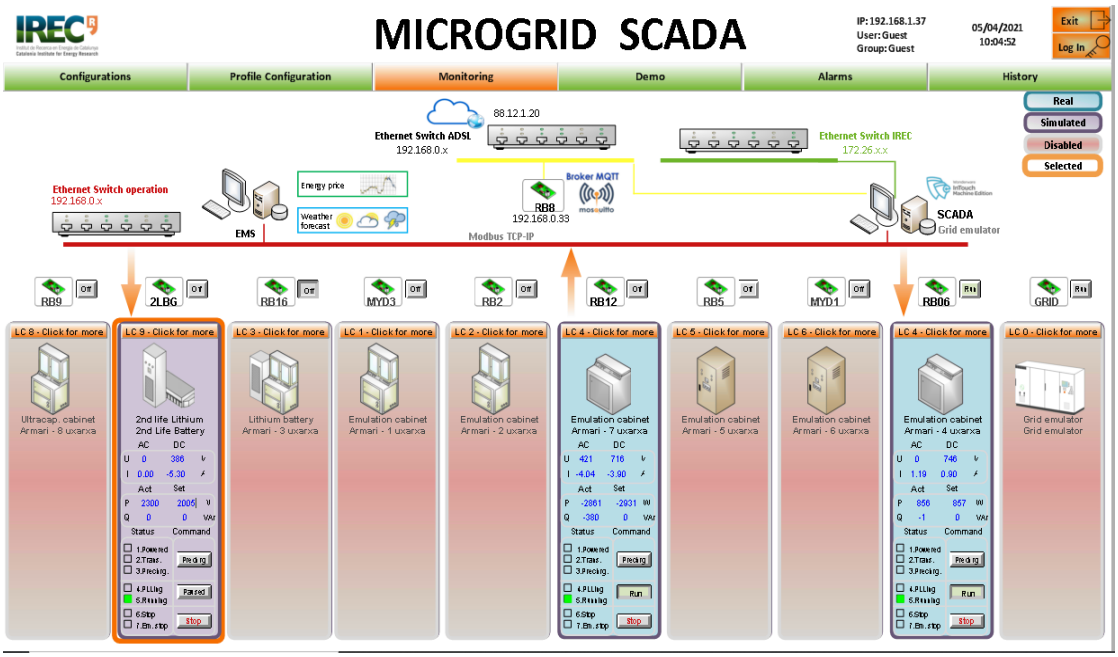

Figure 18: Monitoring screen

2.4.1.5 Demo

The Demo screen is project specific and is designed according to the scenario that is being developed. The Demo screen developed for the current work can be seen in Chapter 3.1.2.

2.4.1.6 Alarms

The alarms screen can be seen in Figure 19 and shows a history of all the incidents that have taken place. Currently, only disconnection and LC errors appear as alarms. However, this list could be further developed.

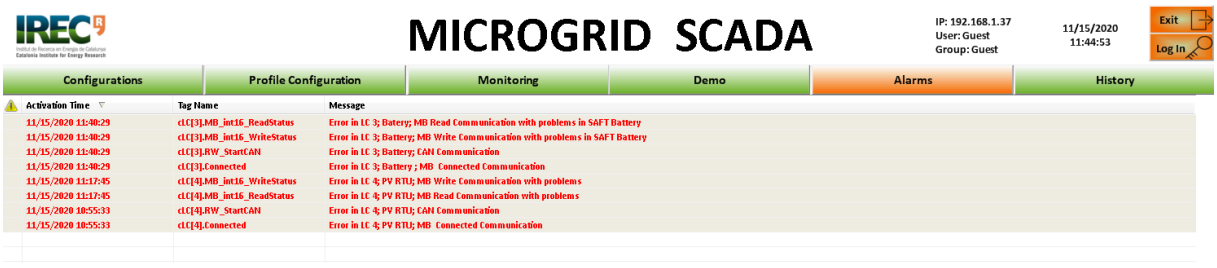

Figure 19. Alarms screen

2.4.1.7 History

This screen allows the users to choose the variables that they want to visualize which are then represented in the graph in the centre of the screen. The time period can be adjusted to show the desired one. The screenshot of this screen can be seen in Figure 20.

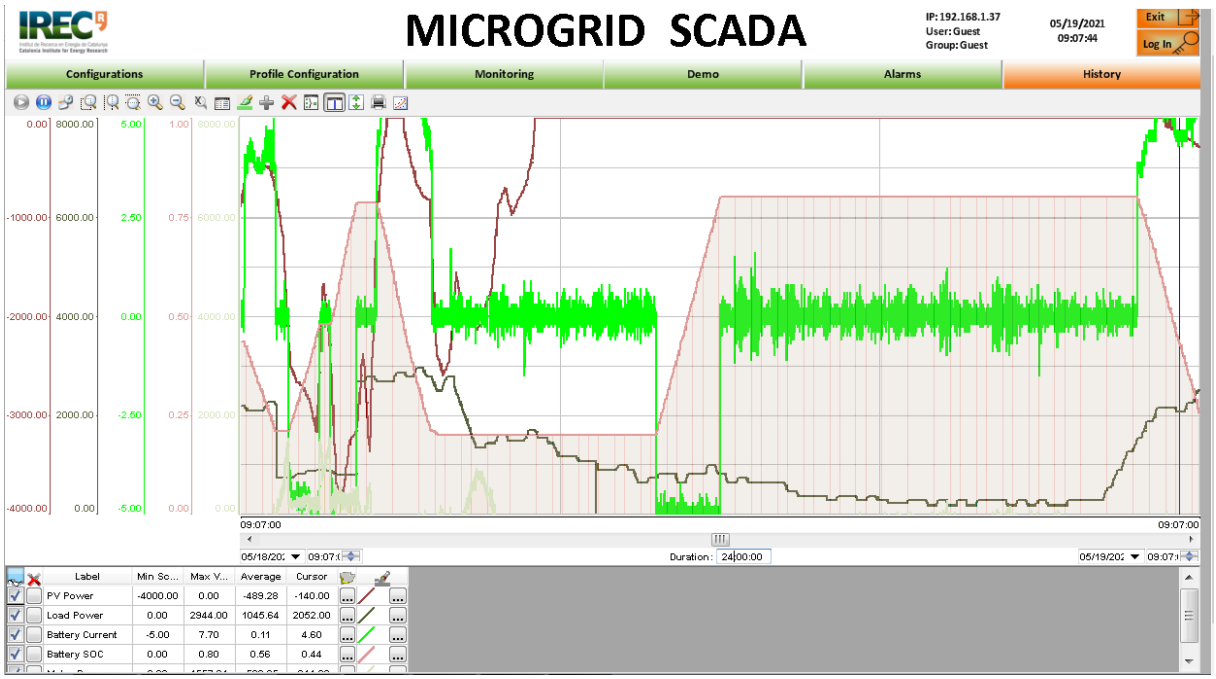

Figure 20. History screen

2.5 Aggregation platform

The aggregation platform, Bamboo Energy, integrated in the current project represents the highest level of control for the SmartLab. The platform was created as a spin-off at IREC in 2018 and its goal is to allow the users to actively participate in Demand Response actions by making use of their flexibility potential. The aggregator is able to send commands to the assets, when grid constraints arise, in order to modify their consumption. These constraints can be caused by a lack of generation capability or a high forecasted demand, which get reflected in high electricity prices or local grid congestions. More information on the platform can be found in the official website [41].

When the communication with the aggregator is enabled, real-time monitoring data of the SmartLab is sent to the platform. With this information and by using internal algorithms, the aggregator is able to calculate the flexibility that the assets can provide and send the corresponding commands to the laboratory. This data is then read from the laboratory to modify the consumption pattern of the different assets.

The communication between the SCADA and the platform is done, in a first instance, through the InfluxDB API. InfluxDB is a time series database used to writing and querying large amounts of timestamped data [42]. Some important concepts of InfluxDB are:

- Measurements: the part of the InfluxDB data structure that describes the data stored in the associated fields.
- Fields: they are required in InfluxDB data structures and they are not indexed. They are formed by a key-value pair. The key is the name of the field and the value is the actual

data.

- Tags: tags are an optional part of the data structure, but they are useful for storing commonly-queried metadata. They are also formed by a key-value pair but unlike fields they are indexed.
- Timestamps: the date and time associated with a point (in UTC).
- Points: a point is a single data record with a measurement, a collection of tags, timestamp and field.

The InfluxDB API allows a simple way to interact with the database to write and query information. For detailed explanations of the API refer to [42].

3 Laboratory set-up

This section contains a description of all the tasks that were carried out to define the scenarios and integrate the laboratory into the aggregation platform.

3.1 SCADA related work

3.1.1 ITME software description

Wonderware InTouch Machine Edition (ITME), currently knows as AVEVA Edge, is the HMI (Human-Machine Interface) used in the laboratory SCADA. Its most important elements are briefly described in this chapter. For further explanations, refer to the official website [43].

- Tags and classes: variables in ITME are called tags, which can be obtained through communication with external devices, calculations or user inputs. Tags are used to display information, manipulate screen objects or control runtime tasks. They can be basic tags (which hold a single value) or arrays, containing data in the form of Booleans, integers, real numbers or strings. In addition, classes allow to create customized structures of data that are formed by different members each being Booleans, integers, real numbers or strings.
- Screens: screens are the windows where the HMI is executed. A collection of screens can be put together forming the so-called screen group.
- Active objects: the objects that appear on the screens can be of several types (texts, buttons, check boxes etc.). ITME allows adding animations to these objects like defining visibility, defining colour or linking text data to them.
- Recipes: they allow to read and write data to external files while the program is running.
- Schedulers: these elements allow to define timers to the execution of the program. Schedulers are useful to run scripts or define the communication with external devices.
- Math: under the Math element, different calculations can be defined.
- Drivers: drivers allow the communication with connected devices. They contain information about the device and enable the implementation of a specific communication protocol. Through worksheets, the communication interface can be configured.

  Two types of driver sheets can be defined: Main Driver Sheet and Standard Driver Sheet. The first one is easier to configure and allows any header supported by the device. The Standard Driver Sheet gives more customizable triggering for write and read actions but only allows one header per sheet.
- Trends: they allow to link trend curves that show project tags with external databases.
- Alarms: alarms allow to create alerts when an unusual condition or error is detected.
- Scripts: to make a project more customized, ITME allows to create scripts in VBScript.

3.1.2 Demo creation

This section presents an explanation of the different elements implemented in the SCADA to carry out the Demo. The Demo has been created according to the scenarios defined. Three main scenarios have been developed, as summarized in Table 3.

Scenario	Name	Goal
1	Residential	Test the flexibility potential of a customer with a HVAC and an uncontrollable load (winter and summer)
2	Prosumer	Test the flexibility potential of a customer with a PV, battery and an uncontrollable load in a self-consumption system (summer)
3	OpenADR	Test the OpenADR protocol for sending events to a load and compare it with the current aggregator API

Table 3: Summary of the project scenarios

Figure 21 shows the screen developed for the Demo, which consists of different areas and elements (described in detail in Annex C), as explained below.

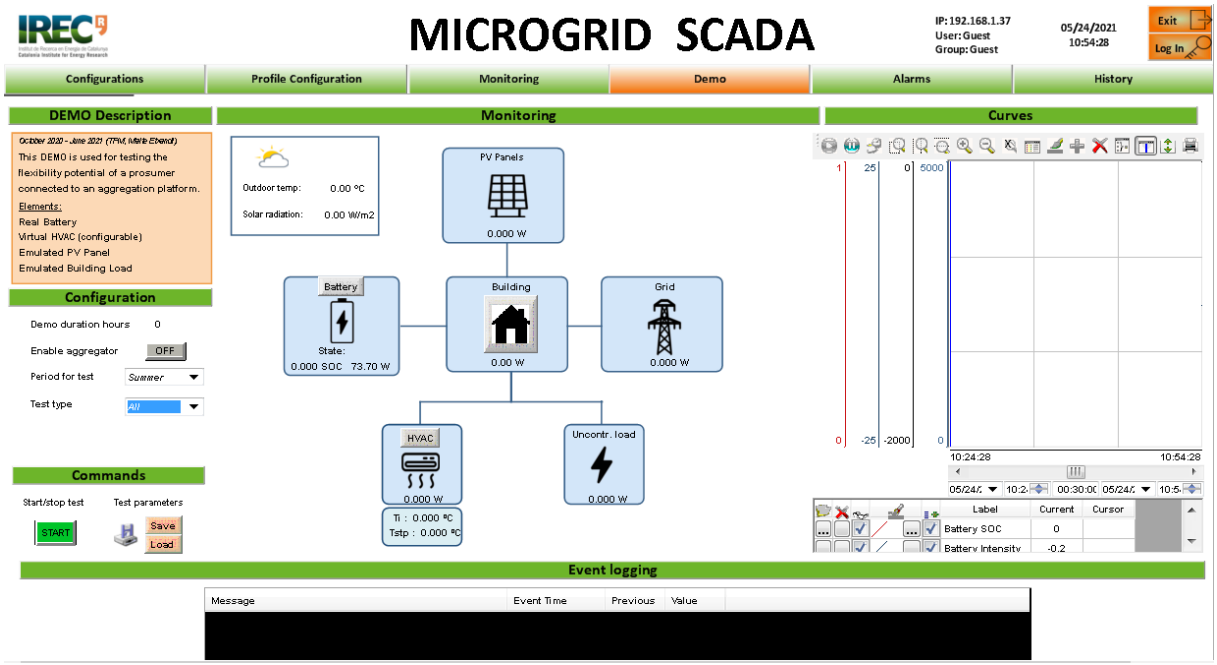

Figure 21: Demo Screen

- Configuration: the user can input the Demo duration hours, the period for the test (summer or winter) and the type of test (residential, prosumer or OpenADR). In addition, there is a push button to activate or deactivate the communication with the aggregator (*Aggregator_on* variable).
- Commands: two main actions can be performed. The user can save the current configuration to a recipe or load the existing one. Also, the START button allows to start running a test (*DEMO_on* variable).

- Monitoring: in this part the user can see the elements that are present in the Demo, the current meteo values, and the current variables of the test for each asset. In addition, by clicking on the HVAC button, the user can configure the properties of the HVAC, and when clicking on the building, the thermal characteristics can be defined (see section 3.2 for a detailed explanation of the virtual HVAC parameters). The two subscreens opened are shown in Figure 22. Similarly, when clicking on the battery name, the battery parameters can be selected through the subscreen shown in Figure 23 (see section 3.4.1 for explanation on the battery parameters).

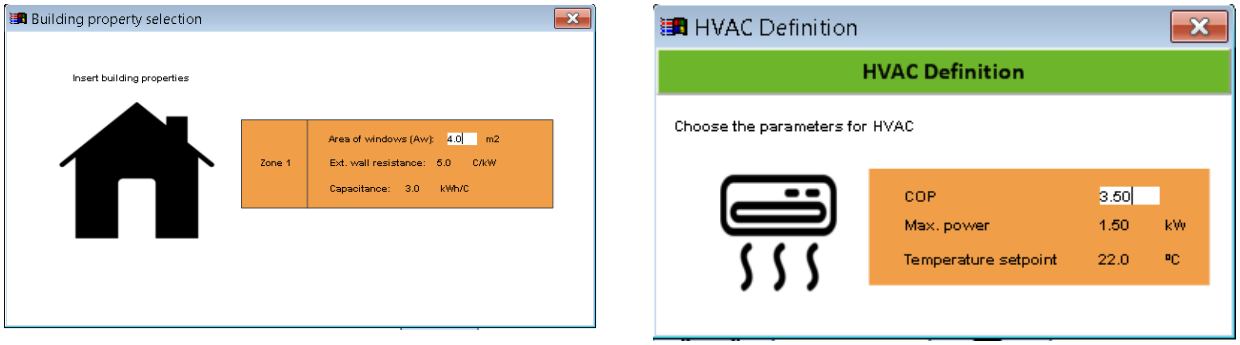

Figure 22: HVAC and building selection subscreens

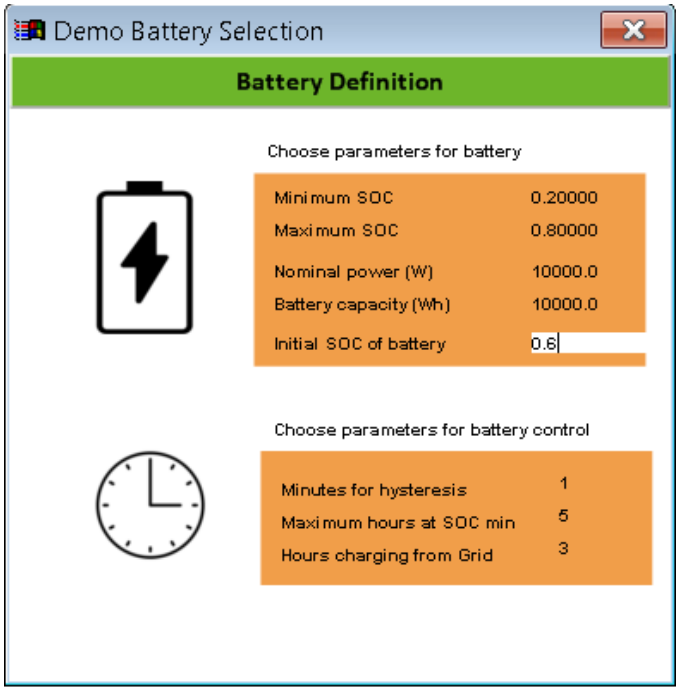

Figure 23: Battery selection subscreen

When the Demo is running, the screen also shows the runtime of the scenario and the remaining time till the end of it, as calculated in the Demo script.

- Event logging: in this section the user gets notifications from commands received from the aggregator.

- Curves: the graph shows the evolution of the variables of interest for each test.

In the following sections, each of the three scenarios developed (see Table 3) is explained. The scripts, schedulers and recipes referenced can be seen in detail in Annex C.

3.1.2.1 Residential Demo

The residential consumer consists of a virtual HVAC and a virtual load. The Demo runs with the use of some variables that trigger tasks. These tasks take place every 15 minutes, however, since some of them depend on others, it is important to make sure that they run in order and synchronously. This means that each task is finished before jumping to the following one.

For the residential case, the Tasks script gets executed every 15 minutes (variable *Task_Script* is set to 1) and variables *t0-t7* (Figure 24) get triggered activating the corresponding schedulers, each defined as a synchronous execution. If an error is detected it gets logged. Finally, after task 7 is executed T*ask_Script* is set to 0.

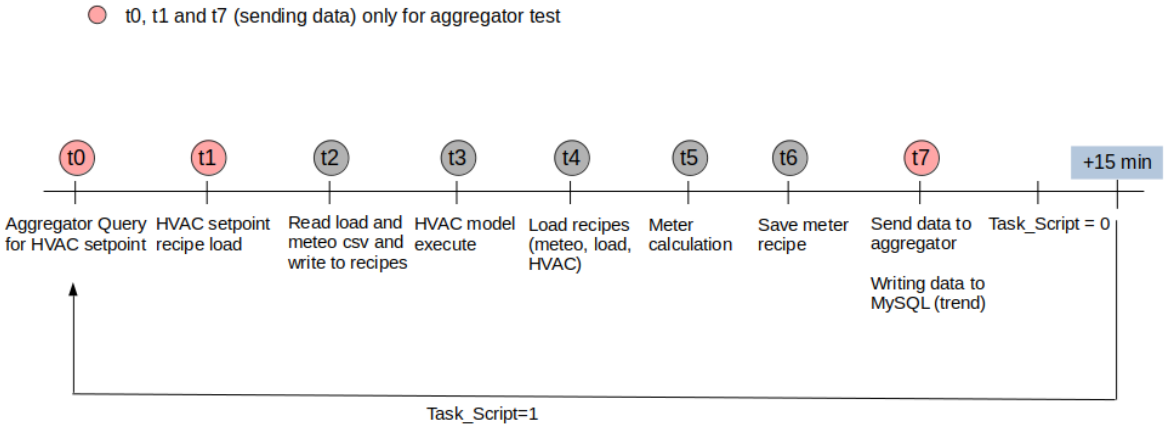

Figure 24: Tasks of residential Demo

- Task 0: when the communication is allowed, the HVAC setpoint is queried to the aggregator through the *InfluxDB_Query.py* script. This setpoint is written into a recipe (*ACTIVATIONS_RES.DAT*).
- Task 1: the previous recipe is loaded into the SCADA.
- Task 2: The corresponding value for the meteo and load are read from the csv files. These values are written to recipes (*METEO.DAT* and *LOAD.DAT*).
- Task 3: the HVAC model script is executed (*HVAC_model.py*) and the values are written to a recipe (*HVAC.DAT*).
- Task 4: all the previous recipes (HVAC, LOAD and METEO) are loaded into the SCADA.
- Task 5: the meter is calculated.
- Task 6: the recipe *METER.DAT* is saved.
- Task 7: all the data from the current interval is written to the SCADA database mySQL using the *Test results residential trend*. Also, when communicating with the aggregator, the current data from the load, HVAC and meter are sent to the aggregator (the script

used for this task is *InfluxDB_ReadDAT_WriteDB.py*).

3.1.2.2 Prosumer Demo

The prosumer consists of an emulated PV system, an emulated load and the real battery from the SmartLab. In this case the tasks are divided into two groups as shown in Figure 25.

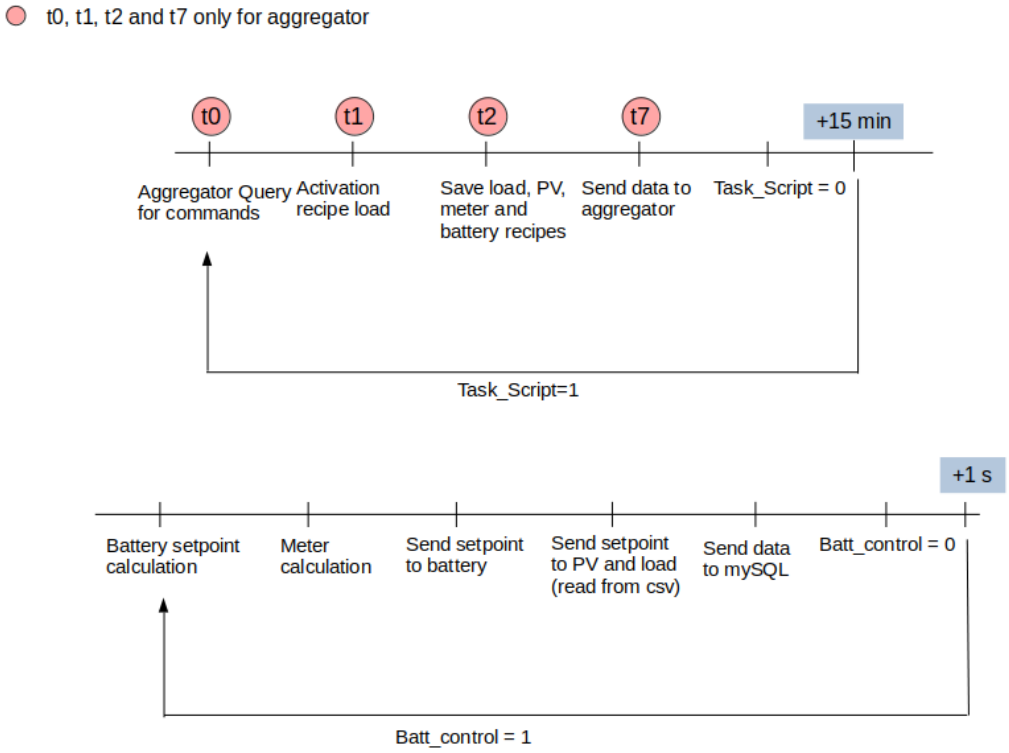

Figure 25: Tasks of prosumer Demo

The tasks that run every 15 minutes are controlled by the triggers *t0*, *t1*, *t2* and *t7* from the *Tasks script* and are only used for the aggregator case (when *Aggregator_on* is set to 1 by the user by pushing the *Enable Aggregator* button).

- Task 0: the activation from the aggregator is queried using the *InfluxDB_Query.py* script and saved into a recipe (*ACTIVATIONS_PRO.DAT*).
- Task 1: the previous recipe with the activation is loaded to the SCADA.
- Task 2: all recipes are saved (*LOAD.DAT*, *PV.DAT*, *METER.DAT* and *BATTERY.DAT*).
- Task 7: the current data from the load, PV, battery and meter are sent to the aggregator (the script used for this task is *InfluxDB_ReadDAT_WriteDB.py*).

The tasks that run every second are executed through the *Battery control* script (runs when *Batt_control* is set to 1), which consists of:

- Battery setpoint calculation, according to the control defined in Chapter 3.4.1.
- Meter calculation
- The setpoint of the battery calculated previously is written to the LC
- The setpoint of the load and PV is read from the csv profiles and written to the LC
- Data is uploaded to the SCADA database (mySQL)

3.1.2.3 OpenADR Demo

The OpenADR Demo uses one of the emulation cabinets as a controllable load. For this Demo, when *Aggregator_on* is set to 1 in the configuration, the communication is done using the current aggregator API. When this variable is set to 0, it means that the communication is done using OpenADR. The script O*penADR test* organizes the main tasks to be executed for this test and gets executed every minute (when *oadr_trigger* is set to 1). These tasks (Figure 26) are:

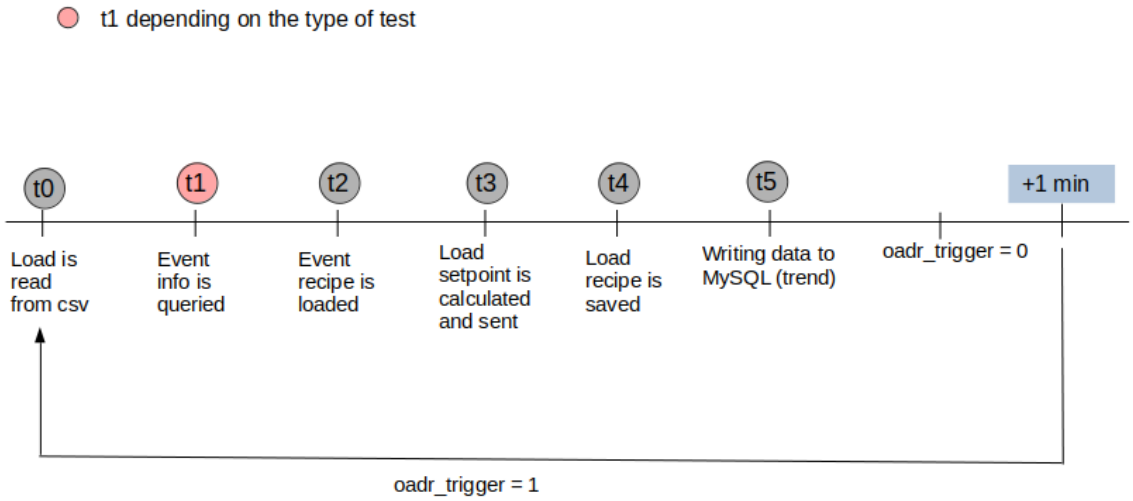

Figure 26: Tasks of OpenADR Demo

- Task 0: Read the base load setpoint from the csv file containing the profile.
- Task 1: flexibility activations are written into the *ACTIVATIONS_OPENADR* recipe. Task 1 depends on the value of *Aggregator_on* (see below)
- Task 2: The event info is read from the *ACTIVATIONS_OPENADR* recipe.
- Task 3: If an event is is active the modified setpoint is calculated and sent to the LC.
- Task 4: The *LOAD.DAT* recipe is saved.
- Task 5: The variables are sent to the SCADA database (mySQL).

For the OpenADR communication, the OpenADR client must be executed first (using the run_openADR_client.bat batch file which executes the *openADR_client.py* script). The main elements that it contains are summarized below, for a detailed understanding of this OpenADR functions refer to Chapter 5.5.

- Requests registration in the beginning of the test and defines the reports to be sent.
- When an event notification or update is received, an “optIn” response is sent to indicate that the client will participate in the event.
- When an event notification or update is received, all the event info is written into the *ACTIVATIONS_OPENADR.DAT* recipe. The second line of the recipe contains the

status, which is “far” for events defined but not active, “active” once the event starts and “completed” once it is over. The first line contains the event signal value and other values in the recipe include event type, duration, number of intervals or client response.

- Every minute the load information is sent to the server.

For the aggregator API communication, the tasks developed are similar to the previous scenarios:

- The activation from the aggregator is queried using the *InfluxDB_Query.py* script and saved into a recipe (*ACTIVATIONS_OPENADR.DAT*) every minute.
- The current data from the load are sent to the aggregator (the script used for this task is *InfluxDB_ReadDAT_WriteDB.py*) every minute.

3.1.2.4 Demo simulations

Before running each test, simulations for the Scenarios 1 and 2 have been carried out in Python to make a validation on the expected results. In particular, to test the battery control and HVAC model developed with the selected meteo and load profiles. Also, example events that the aggregator could send have been defined. Those scripts can be found in Annex F.

3.2 Virtual elements: HVAC

The laboratory cabinets allow to emulate only electrical behaviours. Since the thermal part is necessary to evaluate the indoor temperature of the building, a simulation has been built. With this purpose, a model has been created for one thermal zone. The model allows to calculate the indoor temperature depending on the building characteristics and real time parameters. In addition, the power output of the HVAC is also defined through a control model that follows the temperature setpoint. The parameters used in the model are summarized in Table 4.

TYPE	PARAMETER	NAME	UNIT	DESCRIPTION
Building	Resistance	R1	ºC / kW	Represents the thermal transmission of a wall
	Capacitance	Ci	kWh / ºC	Represents the inertia of the building
	Area of windows	Aw	m^2	Area of windows
Weather	Ambient temperature	Ta	ºC	Outdoor temperature
	Solar radiation	Ps	kW/m^2	Solar radiation per unit surface
HVAC	Coefficient of performance	COP	-	Relation between the thermal power delivered and the electric power consumed
	HVAC nominal power	Pmax	kW	Maximum power of the HVAC
	HVAC power	P	kW	Power delivered by the HVAC
	Setpoint temperature	Tstp	ºC	Temperature to be achieved in the zone
Zone	Indoor temperature	Ti	ºC	Indoor temperature measurement

Table 4. Parameters of the HVAC model

3.2.1 Thermal zone model

A RC model (Figure 27) has been defined to simulate the temperature variations in a thermal zone of a building. These type of models are grey-box models, which are used for building modelling due to the good trade-off between computational speed and accuracy they offer [44]. The model uses a RC network to represent the heat transfer mechanisms that occur inside a building. For the model developed, the change in temperature is due to the following effects:

- Solar radiation through the windows.
- Heat conduction through the walls due to a temperature difference between interior and exterior.
- Effect of the existing HVAC in the zone.
- A noise function (represented as c in the equations) that adds variability to the model.

The model assumes several simplifications to allow an easier definition of the HVAC. These include:

- The internal gains caused by the appliances and people are not considered.
- The COP of the HVAC is considered constant, regardless the temperature setpoint.
- The heat conduction over the walls to adjacent zones, like other floors or rooms, are not considered.

The equation that defines the model of one zone, as explained in reference [45], is:

$$Ti = Ti_0 + \frac{Ta_0 - Ti_0}{R_a \cdot C_i} \cdot \Delta t + \frac{Aw \cdot Ps_0}{C_i} \cdot \Delta t + \frac{COP \cdot P}{C_i} \cdot \Delta t + c \qquad (1)$$

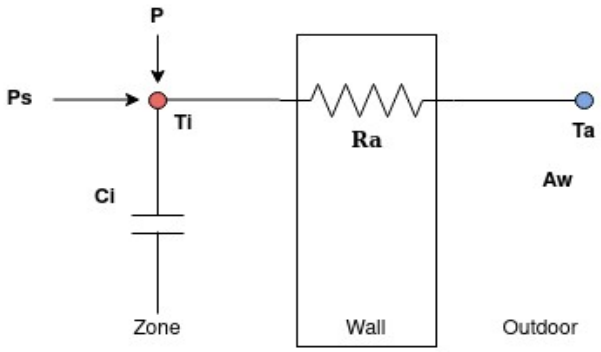

Figure 27: One zone RC model

3.2.2 HVAC control

The control model allows to calculate the necessary power that the HVAC needs to supply in order to achieve the desired indoor temperature, which is defined by a temperature setpoint. The flux diagram of the control developed can be seen in the Figure 28.

The control of the HVAC is defined so that the machine follows a hysteresis, between a maximum and minimum temperature delta over the setpoint, to avoid constant jumps of on/off states in the HVAC.

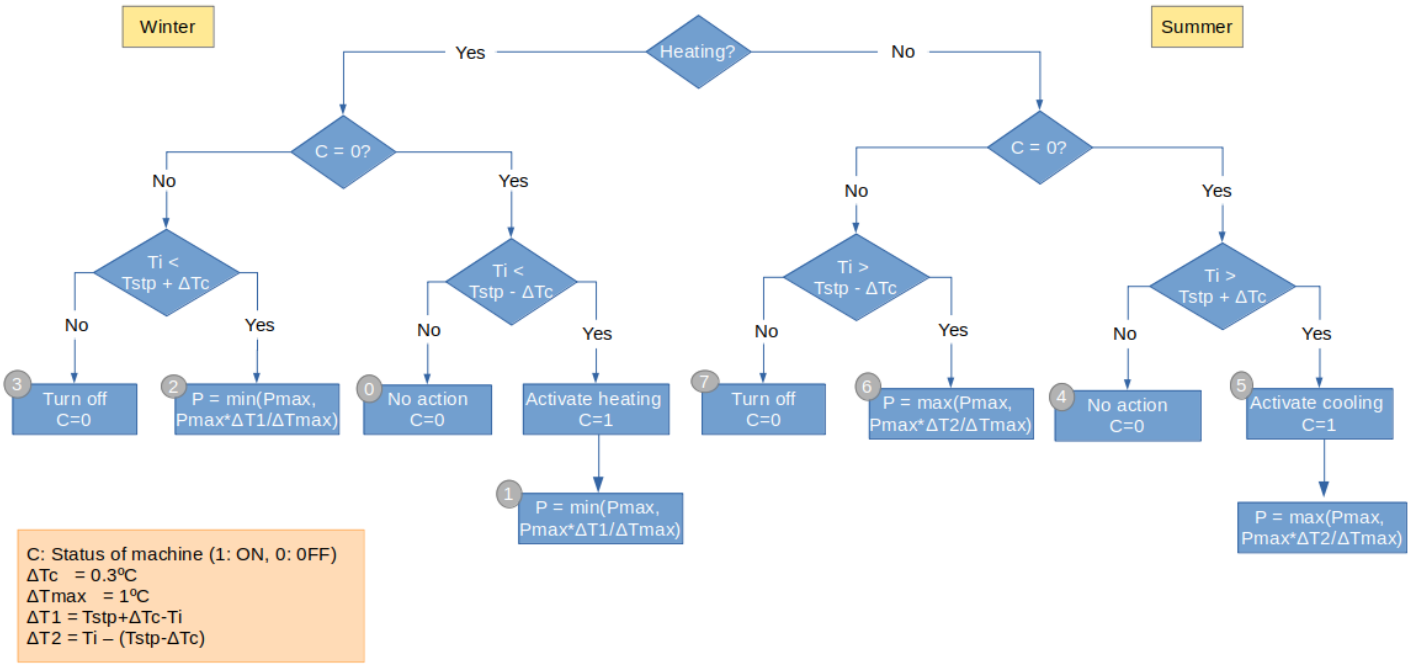

Figure 28. Flux diagram of the HVAC Control

If the machine was stopped at the beginning of the period (C=0), the HVAC turns on only if the temperature is lower than 0.3ºC below the setpoint (heating mode) or more than 0.3ºC above it (cooling). Otherwise, the temperature is in an acceptable range and the HVAC stays off.

If the machine was already turned on (C=1), then the control guarantees that the temperature arrives to a minimum temperature below the setpoint (when cooling) or a maximum one over the setpoint (when heating). This is done to avoid the machine turning on and off constantly.

Whenever the machine is turned on, the power output is calculated depending on the delta of temperature between the desired one and current indoor temperature. Considering that the HVAC works at maximum power when the ΔT is 1ºC, the power output is calculated as shown in equations 2 and 3. This output is always below the maximum power of the HVAC and the minimum working one, which is set at 15% of the maximum one (to avoid the machine working at very low outputs, as it is unrealistic).

The following equations show the HVAC power for cooling (P<0) in (2) and heating (P>0) in (3):

$$P = min\left(P_{min}, max\left(P_{max}, P_{max} \cdot \frac{Ti - (Tstp - 0.3)}{deltaT\,max}\right)\right) \quad (2)$$

$$P = max\left(P_{min}, min\left(P_{max}, P_{max} \cdot \frac{Tstp + 0.3 - T_i}{deltaT\ max}\right)\right) \quad (3)$$

Figure 29 shows the power of the HVAC and the indoor temperature for all the possible states, following the same number notation (states 1-7) as in the flux diagram (Figure 28):

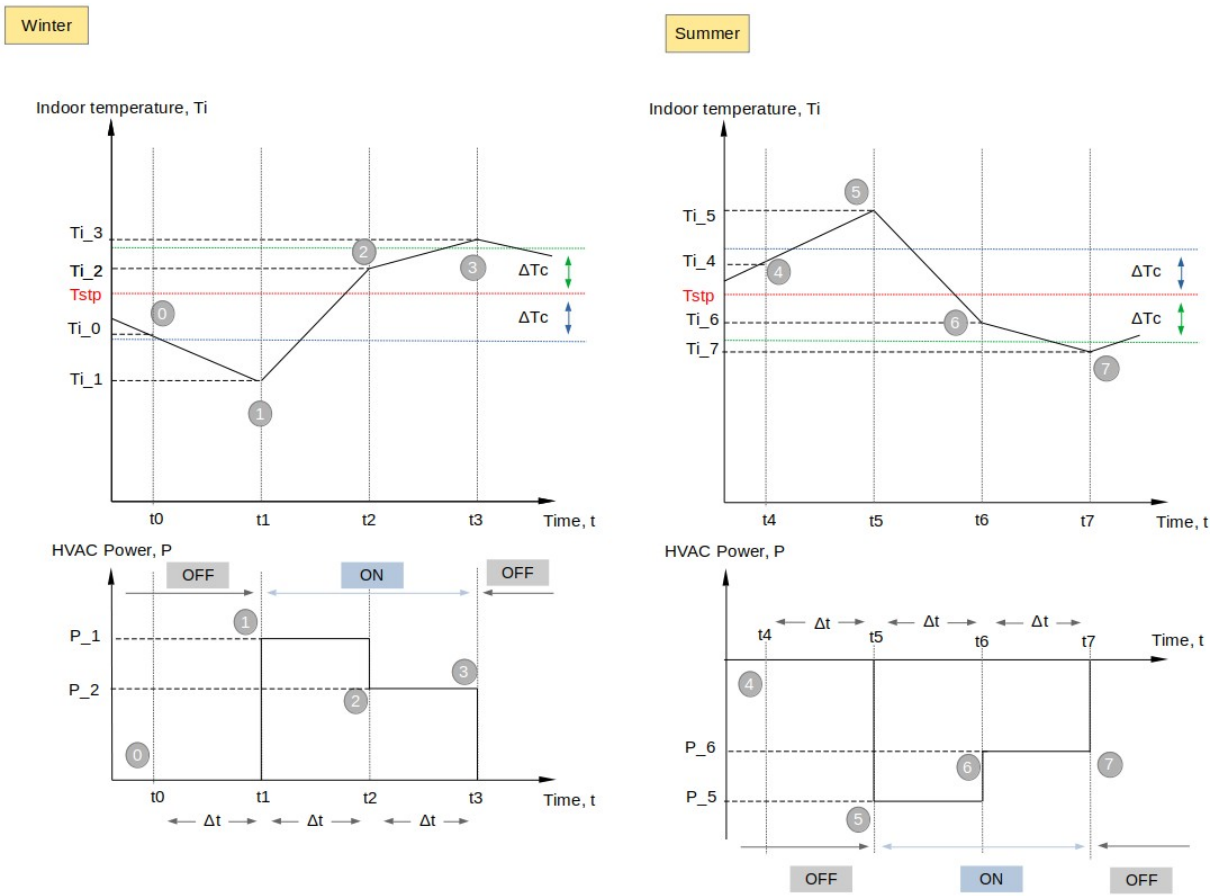

Figure 29: HVAC control states

3.2.3 Virtual HVAC diagram

Figure 30 shows how the desired parameters (Power and indoor temperature) are calculated based on the real time information and the building and HVAC properties. The steps needed to output the indoor temperature are described below:

1- The user inputs the model characteristics at the beginning of a test (COP, Pmax, Ra, Ci, Aw).
2- The solar radiation and outdoor temperature are obtained for the period.
3- The temperature setpoint of the machine is defined. This can be a fixed setpoint chosen by the user or a command sent by the aggregator.
4- With all this data the control model (section 3.2.2) is applied to obtain the required HVAC power to arrive to the temperature setpoint.
5- The thermal zone model (section 3.2.1) is used to calculate the final indoor temperature after the delta of time due to the effects of the solar radiation and the HVAC power.

Steps 2-5 are repeated until the test ends. The Python scripts developed can be found in Annex D. The model has been validated using the data from an existing building provided by IREC.

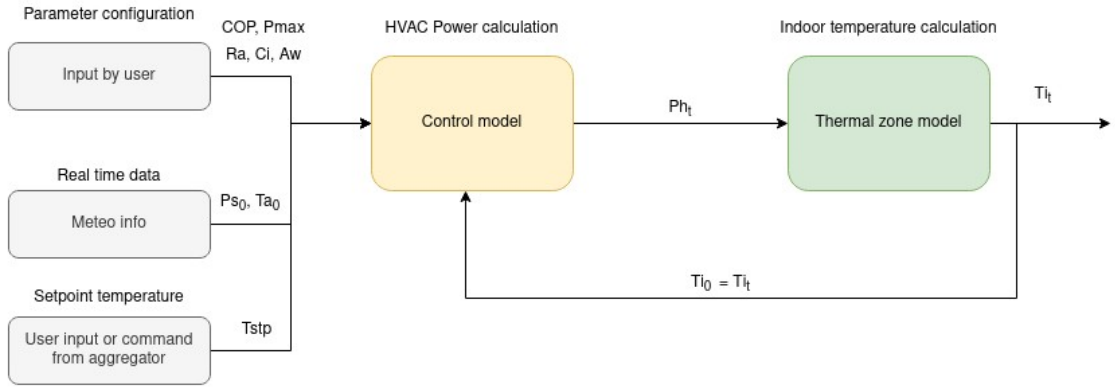

Figure 30: Virtual HVAC diagram

3.3 Emulated elements

To use the emulation cabinets in the laboratory, Modbus drivers were configured from the SCADA. The emulation cabinets used in the current project are shown in Table 5:

Cabinet	IP Address	Port number	PLC ID	Emulation code type
LC 4 - RB12	192.168.0.106	1500	1	20 - Load
LC 7 - 2LBG	192.168.0.112	1500	1	21 - PV

Table 5: Emulation cabinet connection

For each of these elements, a standard driver sheet has been configured, defining the link between the SCADA variables and the Modbus addresses. These variables allow writing and reading data from the LCs. Out of all these variables (see Annex A for the full mapping), the most important ones are shown in Table 6, where “x” is either LC 4 or 7. The full detail of the drivers can be found in Annex C.

Tag name	Modbus Address (header 4X:0)	Description
cLC[x].RW_PsetPoint	7	Active power setpoint to the cabinets (W)
cLC[x].R_P	18	Current active power of the cabinets (W)
cLC[x].R_Error	13	Contains the error code

Table 6: Main SCADA tags for LC 4 and 7 connection

3.4 Real elements: Battery

The cabinet and IP address corresponding to the second-life battery are shown in Table 7:

Cabinet	IP Address	Port number	Emulation code type
LC 9 - 2LBG	192.168.0.247	1502	2

Table 7: Battery connection

From the SCADA point of view, the same methodology as the one shown for the emulation cabinets is applied to connect with the battery. This means defining the driver sheets in the SCADA. For the battery, two different drivers have been configured, one for reading (starting at address 3000) and the other one for writing data (starting at address 4000). The detail of both drivers can be found in Annex C. For the battery the main tags used are shown in Table 8.

Tag name	Header	Modbus Address	Description
cLC[9].RW_PSetPoint	4X:0	5	Active power setpoint to the LC (W)
cLC[9].R_Vdc	3X:0	39	Voltage of the battery (V)
cLC[9].R_Idc	3X:0	38	Intensity of the battery (A)
cLC[9].R_P	3X:0	36	Current active power of the battery (W)

Table 8: Main SCADA tags for LC 9 connection

3.4.1 Battery control

The battery was used in combination with an emulated PV system and a load. For this reason, it was necessary to define the way the battery is be charged and discharged depending on the value of the solar generation and the electricity consumption. The battery control defined constitutes a simple form of control that tries to maximize the users self-consumption and that limits the battery usage range to reduce degradation. The sign criteria is shown in Table 9

Battery	Charging	> 0
	Discharging	< 0
Grid	Consuming from grid	< 0
	Injecting to grid	> 0
Load	Consumption	> 0
PV	Generation	< 0

Table 9: Criteria for signs

The State of Charge (SoC) of a battery gives a measure of the current energy stored. The SoC is defined as the current capacity (Qbat) divided by the maximum capacity of the battery (Qmax). Therefore 0% of SoC means that the battery is empty and 100% that it is full. As both

high and low SoCs increase battery degradation [46], it is common to define a more conservative SoC that the battery will not exceed when discharging or charging.

$$SoC = \frac{Q_{bat}}{Q_{max}} \quad (4)$$

The battery used for the experiments belonged to an EV and had a rated capacity of 23 kWh. Even if it has aged and the current capacity is lower, for the scenarios, this capacity was still too large and only a fraction of the real battery was used. Therefore, the capacity, power and starting SoC of the battery had to be defined. The values of the sized and original battery are presented in Table 10 and a schematic is shown in Figure 31.

Name	Variable	Sized Value	Original value
Battery capacity	Enom	5 000 Wh	23 000 kWh
Starting SoC	SOCstart	60%	-
Minimum SoC	SOCmin	20%	20%
Maximum SoC	SOCmax	80%	95%
Nominal power	Pbat_nom	10 000 W	10 000 W

Table 10: Battery parameter definition

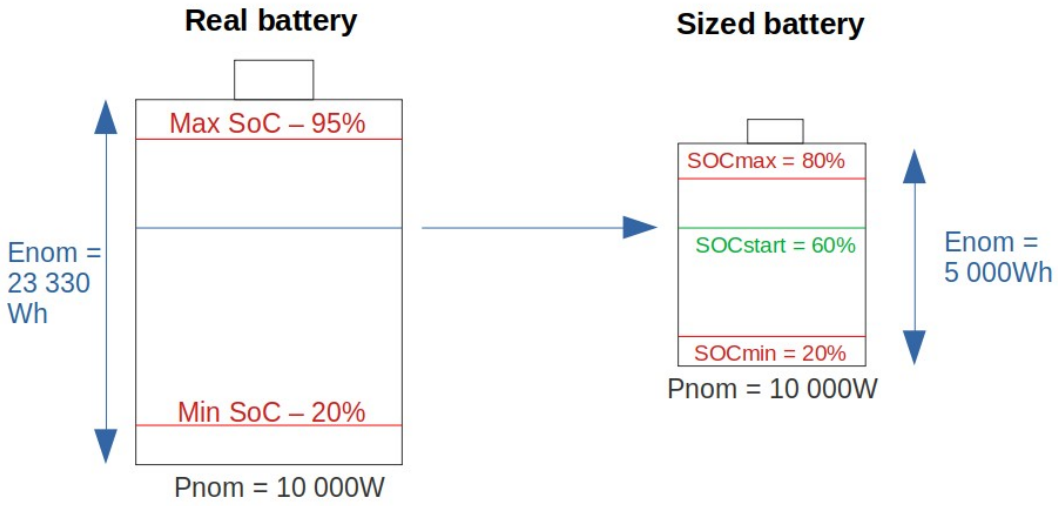

Figure 31: Battery reduction

The variables used in the battery control are described in Table 11.

Variable	Units	Description	Calculation
Pload	W	Power consumed by the load	Read from LC
Ppv	W	Power generated by the PV	Read from LC
Pgrid	W	Power consumed or injected to the grid	$-(Pload + Ppv + Pbat)$
Pbat	W	Power charged or discharged by the battery	Calculated by the algorithm

SoC	-	State of Charge of the battery. Calculated using the intensity and voltage measured every second	$SoC_t = SoC_{t-1} + \frac{(V \cdot I)}{E_{nom} \cdot 3600}$
Esoc	Wh	Energy stored in the battery at a defined SoC	$SoC \cdot E_{nom}$
η_d or η_c	-	Discharge and charge efficiencies respectively	Constant (assumed 98%)
Δt	hours	Frequency of control	Defined as 1 second
t	min	Consecutive minutes with the same battery state (charging, discharging etc.)	Calculated by the algorithm
t1	min	Auxiliary time variable defined to guarantee that discharging or charging are maintained a minimum period of time	Calculated by the algorithm
Pbat_max	W	Maximum power at which the battery can be charged or discharged to avoid overcharge or overdischarge.	$min(Pbat_{nom}, (E_{SOC_{max}} - E_{SOC})/(\Delta t * \eta_c$ $min(Pbat_{nom}, (E_{SOC} - E_{SOC_{min}})/(\Delta t * \eta_d$

Table 11: Variables of the battery control

Other variables used, that can be changed in the SCADA Demo screen (see section 3.1.2) are the ones shown in Table 12:

Variable	Units	Description	Value
t_hist	min	Related to the hysteresis of the battery (see below for explanation)	1 min
t_Grid	hour	Maximum allowed hours at minimum state before charging from the grid (see below for explanation)	5 h
t_Grid_maintain	hour	Hours charging from the grid (see below for explanation)	3 h

Table 12: Configurable time variables of the battery control

The battery control flux diagram is shown in Figure 32 and described in the following lines, where the colours match the ones used in the flux diagram. The possible battery states are summarized in Table 13. Whenever there is electricity production from the PV panels, this is used to fully or partially supply the load. If the PV production is lower than the load consumption, the battery SoC is checked. If it is over the minimum SoC, then the battery is discharged (state D). The condition set here is that the battery never discharges below the minimum SoC. If the battery is at minimum SoC or below then all the electricity needed is taken from the grid (state MIN).

On the other hand when there is a surplus of PV and the battery is below the maximum SoC, it gets charged (state C). The battery never gets charged over the maximum SoC. In case there is still a surplus after charging the battery, it is injected to the grid. If the battery is at maximum SoC or over, all the surplus goes to the electricity grid (state MAX). If the PV generation is equal to the load, the battery does not charge or discharge (state S).

Two other aspects have been controlled on the battery. First, if the battery spends a long period

of time at the minimum SoC (more than *t_Grid*, 5h) without being charged with solar power, it gets charged from the grid for *t_Grid_maintain*, 3h (state G) or until maximum SoC. This is done to maintain the health of the battery. If the battery is at minimum SoC and does not get charged for a while, it wiould start decreasing its capacity due to the self-discharge and thus the degradation would be increased.

Second, to avoid constant change of charging direction of the battery from charge to discharge or vice versa (states C to D) a new condition has been defined. This could happen, for example, if the PV is producing electricity but suddenly a cloud appears for a short period of time, or due to communication delays. The PV output would go from high values to low/zero values and back to high values again in a matter of seconds. To avoid this erratic functioning of the battery, if the battery control goes from charging to discharging, the control checks how long it was in the previous mode and if it was lower than 1 minute, the battery maintains the new state for 1 minute (*t_hist*). The time spent in this forced charge or discharge is measured with the variable t1. This last condition is described in the flux diagram from Figure 33.

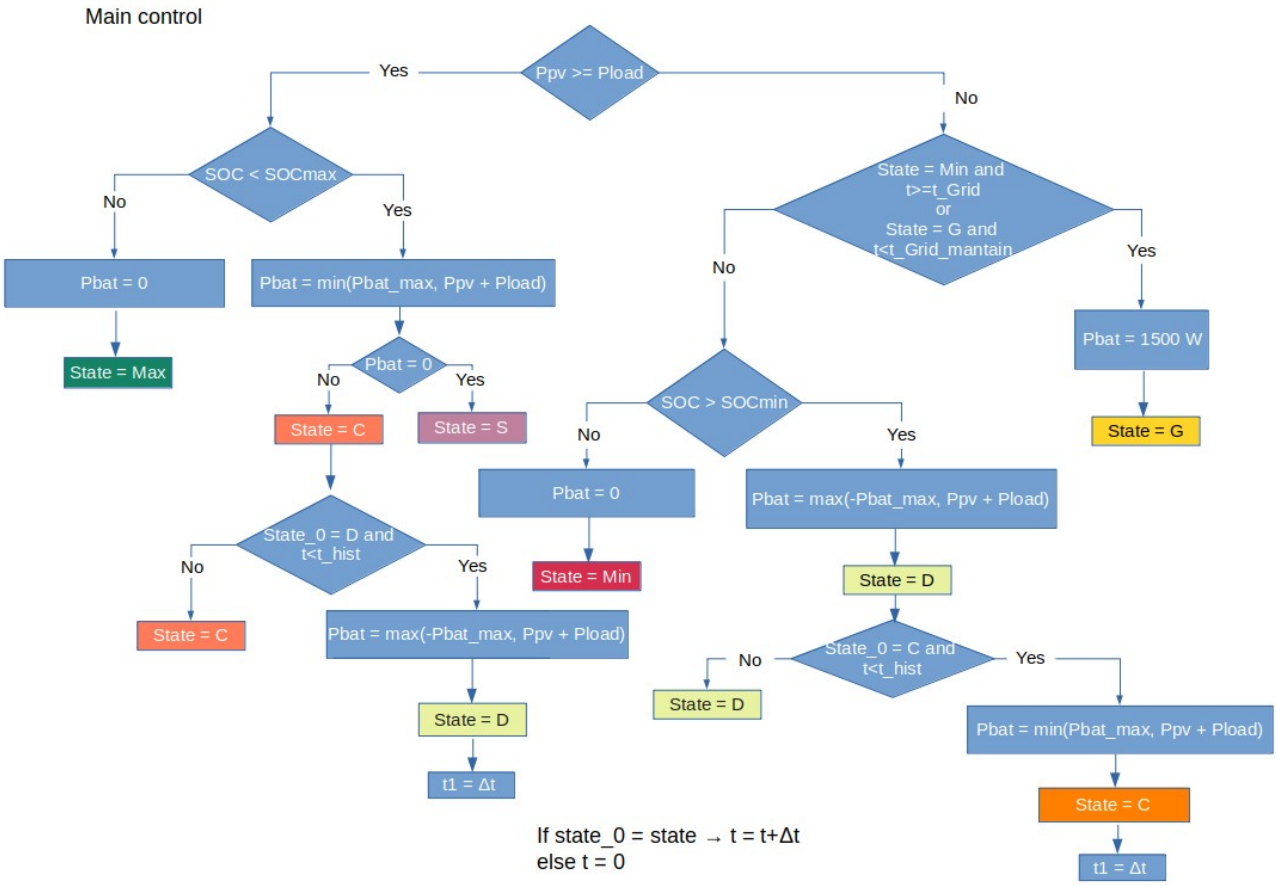

Figure 32: Main battery control flux diagram

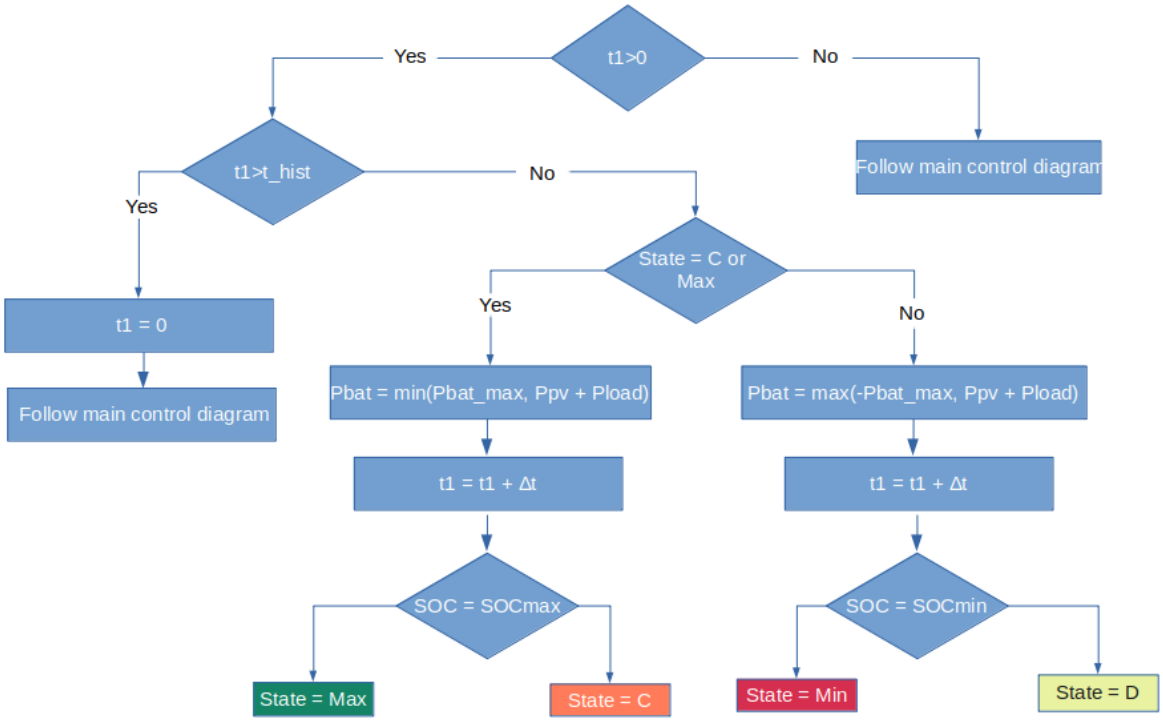

Figure 33: Auxiliary control flux diagram

STATE	DESCRIPTION	Battery Power
C: Charging from PV	Battery charging from PV	Pbat > 0
D: Discharging	Battery discharging to grid	Pbat < 0
G: Charging from grid	Battery charging from grid	Pbat > 0
Min: Minimum	Battery at SoC minimum	Pbat = 0
Max: Maximum	Battery at SoC maximum	Pbat = 0
S: between max and min	Battery between SoC min and max	Pbat = 0

Table 13: Possible states of the battery

The code for the battery has been first implemented in Python for testing and then directly in the SCADA (Annex C).

3.5 Integration with the aggregation platform

There are three main actions to be carried out in order to integrate the SCADA with the aggregation platform:

1. Define the elements that form the microgrid
2. Allow to send data in real time from the SCADA to the aggregator.
3. Read the activations commands sent by the aggregator.

All these actions are done through the InfluxDB API (see Chapter 2.5). For the SCADA integration, the scripts developed have been made using Python and they can be found in Annex E.

3.5.1 Microgrid definition

To calculate the flexibility of the different assets, the aggregator needs information about the elements that form the microgrid. Depending on the asset, different parameters must be defined:

- HVACs: maximum and minimum power, number of thermal zones and comfort temperature sensors for each zone.
- Meters, load or PV: maximum and minimum power.
- Battery: maximum and minimum SoC and maximum and minimum power.

All this information must be sent in a JSON file to the aggregator.

3.5.2 Sending data

The first step is to define, how the communication with the aggregator will take place. In this case, the communication happens through the InfluxDB API. For each asset (HVAC, PV, Battery, Load and Meter) the data sent to the aggregator must contain the following information:

- Measurement: “measures”
- Tags:
  - “Asset”: depending on the asset (“hvac”, “battery”, “PV”, “load” or “meter”)
  - “ID”: “smartlab”
  - “ID_prosumer”: “IREC”
- Timestamp
- Fields:
  - “power”: active power (W)
  - “SOC”: State of Charge (0-100), only when the asset is the battery
  - “temperature”: corresponds to the indoor temperature of the thermal zone (ºC), only when the asset is the HVAC.
  - “status”: this field is 0 when the asset is turned off and 1 when it is on.
  - “quality”: 1 when the quality is good, 0 when there is an error.

Using the Influxdb Python library, the data can be defined using a JSON that contains all the previous data. For each test different assets take part and thus the data sent differs. The script developed to send data to InfluxDB can be found in Annex E. The type of test to be carried out and the data sent to the database is read from the SCADA through recipes (as explained in the section 3.1.2).

3.5.3 Reading commands

A query must be made per each controllable asset (HVAC, battery or load, depending on the scenario) to read commands from the aggregator. The aggregator writes the commands for each asset in the database under the measurement “commands”. To read this data, the query must contain the corresponding tags (asset name, ID and ID_prosumer).

For the HVAC (Scenario 1), the field “temperature” in the database contains the setpoint for the HVAC at all moments. Thus, this value is constantly read from the SCADA through the next query:

"select last(temperature) from caseDB.autogen.commands where Asset='hvac' and ID_prosumer='IREC' and ID='smartlab'"

For the the battery (Scenario 2), two fields have been defined: “activation” which contains a 1 if the aggregator is sending a setpoint to the battery and 0 otherwise. The second field is “power” which contains the power setpoint for the battery.

"select last(activation) from caseDB.autogen.commands where Asset='battery' and ID_prosumer='IREC' and ID='smartlab'"

"select last(power) from caseDB.autogen.commands where Asset='battery' and ID_prosumer='IREC' and ID='smartlab'"

For the the load (Scenario 3), two fields have been defined: “activation” which contains a 1 if the aggregator is sending a setpoint to the load and 0 otherwise. The second field is “power” which contains the percentage at which the load should be working, in respect to the base case.

"select last(activation) from caseDB.autogen.commands where Asset='Load' and ID_prosumer='IREC' and ID='smartlab'"

"select last(power) from caseDB.autogen.commands where Asset='Load' and ID_prosumer='IREC' and ID='smartlab'"

The script developed for the queries can be found in Annex E. The test to be developed is read from a configuration recipe. Then, the query is made for each asset that takes part in the test. Finally, the activations are written into recipes that are read from the SCADA.

4 Scenario development

To study the potential flexibility that different consumers can provide, two customers have been created. For each one, the base case has been first developed, with no interaction with the aggregator. Then, the communication with the aggregator has been allowed in order to see how these customers would be affected by the commands received. Each test lasts 3 days.

1 Residential: this customer has a HVAC that can be used for flexibility and an uncontrollable load that represents other consumptions (lighting, appliances…). In the base case two temperature setpoints are defined (one for the night and the other for the day). For the aggregator case, during peak demands, different temperature setpoints are received for the HVAC system to change the level of consumption. Both summer and winter scenarios have been considered.

2 Prosumer: this customer has a self-consumption system formed by a PV, a second-life EV battery and an uncontrollable load. The goal of this scenario is to analyse how the battery charge and discharge power can be modified by the aggregator. The base case only uses the battery control defined in section 3.4.1, that tries to maximize self-consumption. If there is a generation surplus it is sold to the grid. In the aggregator case, the setpoints calculated by the battery control can be overwritten by the activations received.

An overview of the developed tests and cases can be seen in Table 14:

Nº	NAME	ELEMENTS			CASES TESTED
		Name	Type	Controllable	
1	Residential consumer	HVAC	Virtual	Yes	1 Summer base case 2 Winter base case 3 Summer with aggregator 4 Winter with aggregator
		Load	Virtual	No	
2	Prosumer	PV	Emulated	Yes	1 Base case 2 Case with aggregator
		Battery	Real	Yes	
		Load	Virtual	No	

Table 14. Overview of the developed scenarios

To evaluate the cost of the electricity bill, the average of the Spanish Voluntary price for the small consumer (PVPC) with two periods, during the days of test (year 2020) [47] has been considered. For the prosumer scenario, the surplus energy is sold at the price specified in the third row of Table 15, obtained from also from the regulated values [48].

	Summer (17-19 July)		Winter (10-13 December)	
	Hours	**Price**	**Hours**	**Price**
Peak	13-22h	0.1109 €/kWh	12-21h	0.1299 €/kWh
Off-peak	23-12h	0.0372 €/kWh	22-11h	0.0537 €/kWh
PV surplus	00-23h	0.0310 €/kWh	-	-

Table 15: Electricity tariff

Finally, to quantify the effect of the aggregator in each scenario, different KPIs (Key Performance Indicators) have been extracted from literature [49][50] and from the ones used by the aggregator:

- **KPI1 - Total energy consumption change:** measures the reduction or increase of the energy imported from the grid over the three days of the test.
- **KPI2 - Total energy cost change:** measures the reduction or increase of the cost over the three days of the test.
- **KPI3 - Peak load reduction**: in order to check if the demand has been reduced during the DR events, this KPI measures the difference in power during the events defined by the aggregator.
- **KPI4 - Rebound effect:** following a DR event, it is common to experience what is known as the “rebound effect”, which reflects the additional energy consumed due to the flexibility activation. This KPI measures this effect by comparing the energy consumed during the hour after the event is finished.
- **KPI5 - Reliability index:** this KPI is used to measure whether the user has successfully received the activation. In this case, the index is calculated by the number of time intervals in which the user has read the correct setpoint from the aggregator.
- **KPI6 - Thermal comfort level:** this KPI has been defined as the amount of time intervals that the indoor temperature exceeds the comfort of the users (19ºC to 24ºC) during the event and rebound periods. Only used for the residential scenario.
- **KPI7 - Self consumption factor change:** measures the change in the self consumption factor of the aggregator case and the base case. This factor calculates the percentage of load that has been covered by the PV system, directly from it or from the battery. Only used for the prosumer scenario.

Nº	KPI	Equation	Scenarios
1	Total energy consumption change	$\frac{ET_{aggregator} - ET_{base}}{ET_{base}} \cdot 100$	Both
2	Total energy cost change	$\frac{CT_{aggregator} - CT_{base}}{CT_{base}} \cdot 100$	Both
3	Peak load reduction	$\frac{PDR_{aggregator} - PDR_{base}}{PDR_{base}} \cdot 100$	Both
4	Rebound effect	$\frac{Erebound_{aggregator} - Erebound_{base}}{Erebound_{base}} \cdot 100$	Both
5	Reliability index	$\frac{Intervals\ with\ correct\ Tstp}{Total\ number\ of\ intervals}$	Both
6	Thermal comfort level	$\frac{Intervals\ out\ of\ confort_{aggregator} - Intervals\ out\ of\ confort_{base}}{Intervals\ out\ of\ confort_{base}}$	Residential
7	Self-consumption factor change	$\frac{SC_{aggregator} - SC_{base}}{SC_{base}}$	Prosumer

Table 16: KPIs used for the scenarios

Where:

- ET: energy imported from the grid during the three days
- CT: cost of the energy consumed during the three days. The cost refers to the electricity bill and does not consider the incentives offered by the aggregator.
- PDR: power measured during all the DR events sent by the aggregator
- `Erebound`: energy consumed the hours after all the DR events sent by the aggregator
- SC: self-consumption factor, calculated as $SC = 1 - \frac{Electricity\ imported}{Electricity\ consumed}$

4.1 Scenario 1: Residential consumer

The first scenario, shown in Figure 34, represents a regular household with a HVAC system and a load. The HVAC has a nominal power of 1.5 kW and the rest of the consumer's load are 1.5 kW. The main building and HVAC characteristics chosen are presented in Table 17.

The interest of this scenario lies in its application to several customers. HVACs are a common asset in households and as explained in Section 1.4.1, they are an important source of flexibility. A screenshot of the SCADA Demo for this scenario is presented in Annex G.

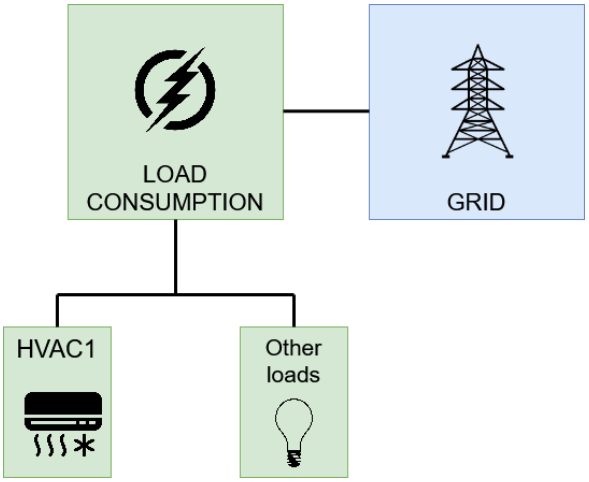

Figure 34: Residential consumer diagram

ELEMENT	MAXIMUM POWER (W)	TYPE
HVAC	1500	Virtual
Load	1500	Virtual

VARIABLE	VALUE
COP	3.5
Area of windows	4 m^2
Resistance of wall	5 ºC / kW
Capacitance	3 kWh / ºC

Table 17: Residential scenario characteristics

The goals of this first scenario are the following:

- Validate the virtual HVAC model developed
- Test the different elements defined in the SCADA (schedulers, scripts etc.)
- Test the correct integration of the SCADA into the aggregation platform through the Aggregator API (sending data and reading commands).
- Analyse the effect of the aggregator in the HVAC consumption both for winter and summer and determine the flexibility services that a residential user can provide.

4.1.1 Profile selection

All the profiles used were provided by IREC and have a frequency of 15 minutes. The original data corresponded to the a whole hotel for the year 2019 in Barcelona. The profiles for two periods, summer (17/07 – 19/07) and winter (10/12 – 12/12), were extracted and then multiplied by a reduction factor to represent a usual residential house. The load in this scenario represents all the consumption from the customer that is not controlled by the aggregator (lightning, washing machines, TV, oven...). To execute the HVAC model the environmental conditions must be defined (outdoor temperature and radiation). In Figure 35 and Figure 36, the load and meteo profiles can be seen, respectively.

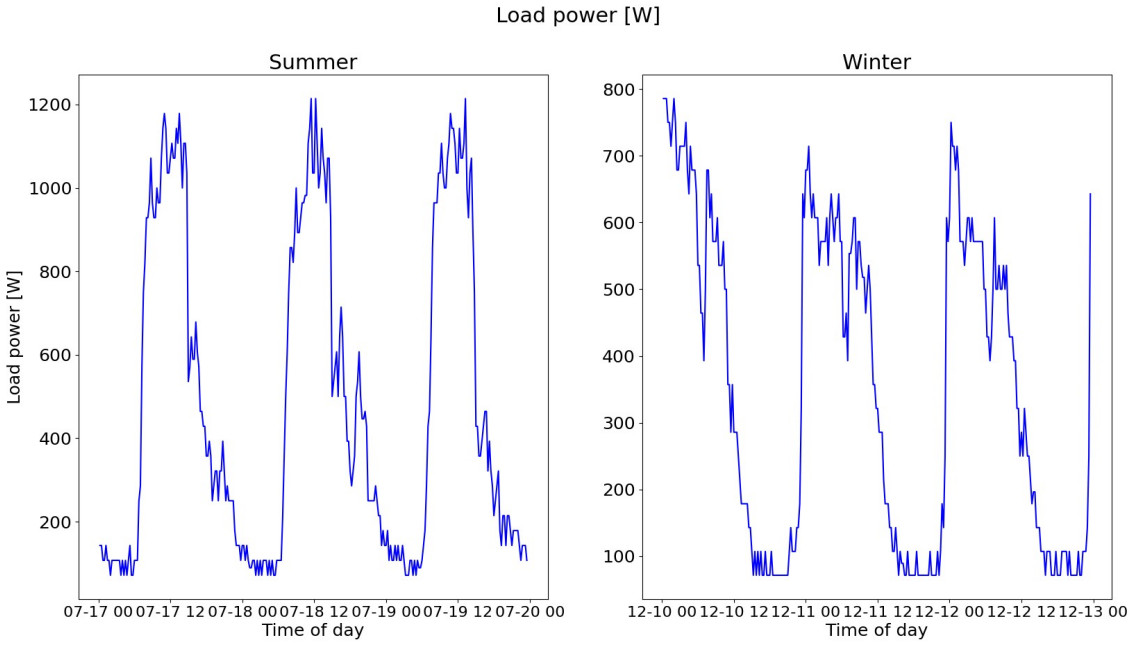

Figure 35: Load profiles for the residential scenario (summer and winter)

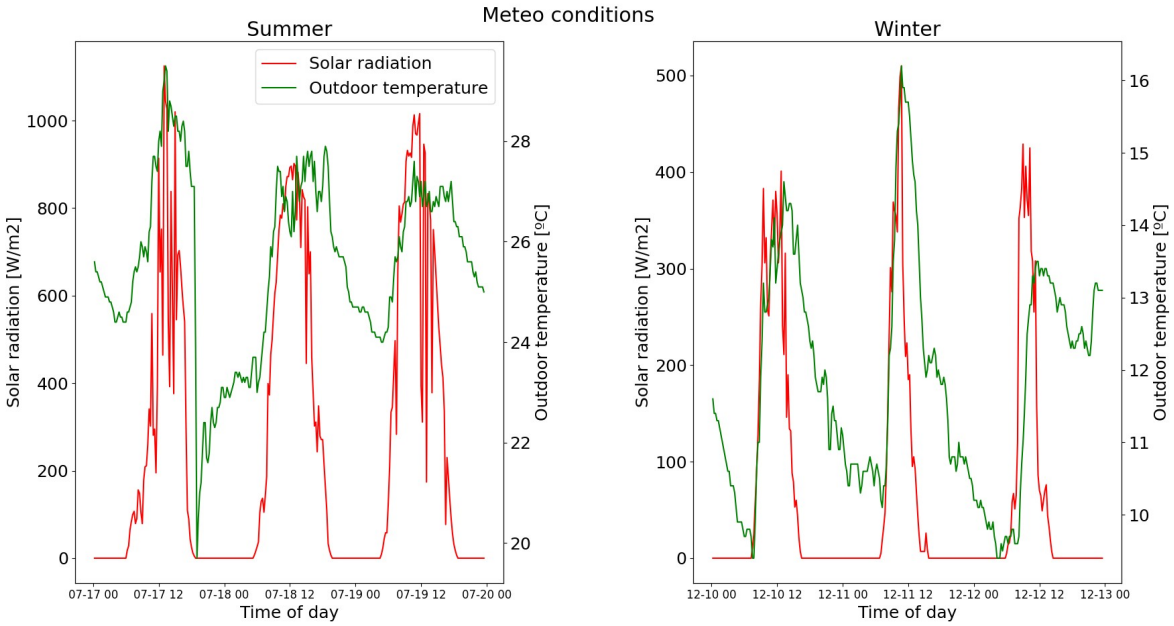

Figure 36: Meteo profiles for the residential scenario (summer and winter)

4.1.2 Base case - summer

This case represents a standard consumption of a residential consumer during three days in summer. Since no commands from the aggregator are received, the HVAC works to achieve the comfort temperature set by the user. During the daytime, the user seeks to achieve thermal comfort, while at night it is common to increase the setpoint temperature to reduce the HVAC consumption. Thus, two temperature setpoints have been defined:

- Night period (from 12pm to 8am): 22ºC
- Daytime (the rest of the day): 24ºC

The HVAC works according to the control defined in Chapter 3.2.2 to achieve these indoor temperatures. The resulting consumption of the HVAC is shown in the top of Figure 37, in the middle, the indoor temperature versus the setpoint and on the bottom, the outdoor temperature.

From the power evolution we can determine a pattern in the HVAC. In general, in the morning (at 8am), the machine starts cooling at high power values, arriving in some cases to work at the maximum power, 1.5 kW. The high powers are due to the sudden setpoint change, from 24ºC at night to 22ºC during the day. Late in the afternoon (around 8-9 pm), when the solar radiation and the indoor temperatures are lower, the HVAC is turned off, while maintaining the desired indoor temperature. The HVAC stays off during the night, as the setpoint increases to 24ºC.

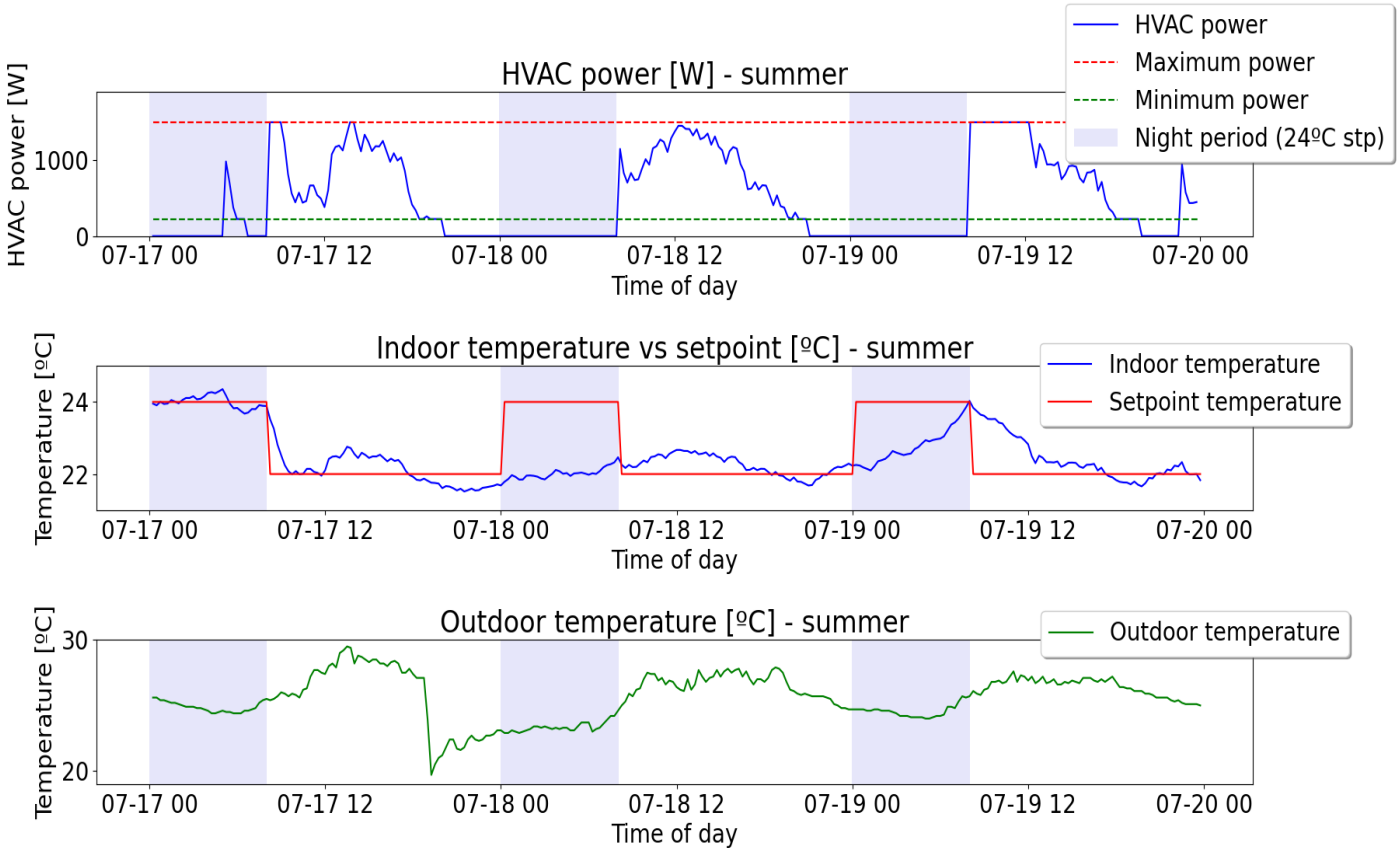

Figure 37: HVAC Power and temperature evolution (residential base case, summer)

There are some exceptions to this pattern, depending on the meteo conditions. For example, during the first night, we see that the indoor temperature starts increasing over the setpoint and, as a consequence, the HVAC starts working. We can also see that, because the ambient temperatures are lower in the first morning, the HVAC goes from working at maximum power to lower ones during some hours of that period.

Regarding the indoor temperature, we see how it is maintained in a reasonable range (between

21.5ºC and 24.5ºC approximately). During the day, the indoor temperature is close to the defined setpoint (22ºC), while at night it changes depending on the outdoor conditions. The first night, there is a temperature increase, but it is maintained around the setpoint (24ºC). During the second night however, we see that the temperature decreases. The outdoor temperature at this period is lower and, as the machine is not allowed to provide heating during the summer, the temperatures drop. Finally, in the third night, the temperature increases over night until the HVAC is turned on in the morning.

The maximum temperature difference between the setpoint and the indoor temperature during the day is almost 2ºC (during the last morning, right when the temperature setpoint changes). Table 18 shows, the minimum, maximum and mean temperatures achieved each day and night.

	Night 1	Day 1	Night 2	Day 2	Night 3	Day 3
Minimum Ti	23.68 ºC	21.52 ºC	21.79 ºC	21.68 ºC	22.10 ºC	21.66 ºC
Maximum TI	24.36 ºC	23.52 ºC	22.47 ºC	22.67 ºC	24.03 ºC	23.84 ºC
Average Ti	24.0 ºC	22.11 ºC	22.02 ºC	22.25 ºC	22.79 ºC	22.49 ºC

Table 18: Indoor temperature outputs (residential base case, summer)

The total consumption of the building, including the load and the HVAC can be seen in Figure 38. The consumption during the day is clearly higher, as both the uncontrollable load and the HVAC are consuming electricity.

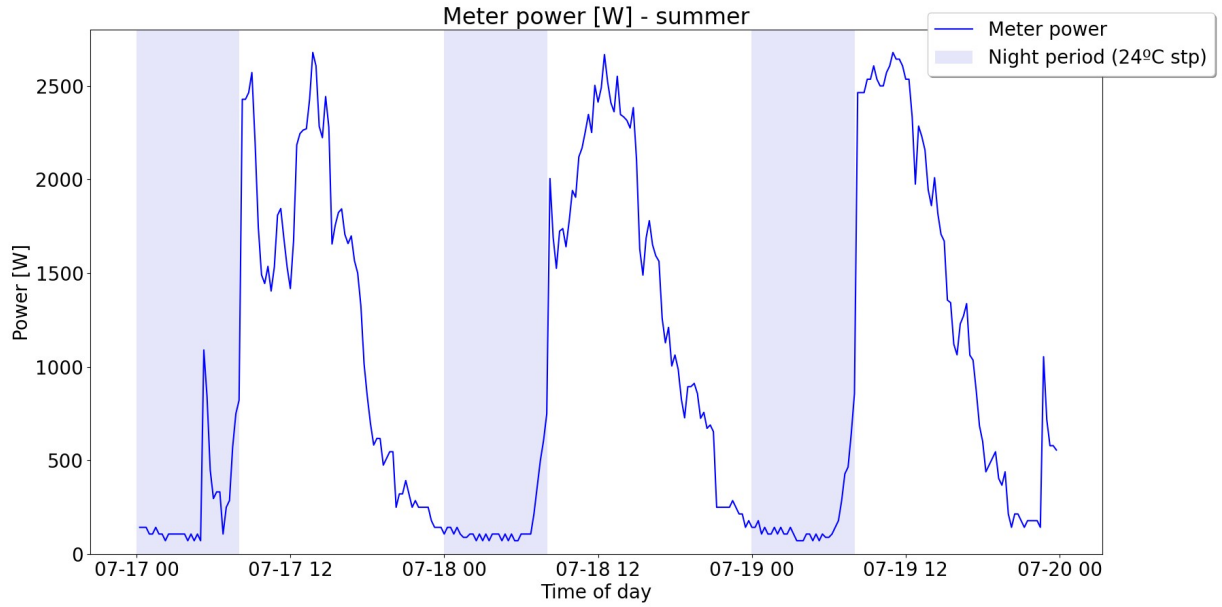

Figure 38: Meter power consumption (residential base case, summer)

The total energy consumed and the mean powers during the three days are shown in Table 19:

	Day 1	Day 2	Day 3	Total
Load Mean Power	499.63 W	497.40 W	457.22 W	**484.75 W**
HVAC Mean Power	461.09 W	494.82 W	510.02 W	**488.64 W**
Meter Mean Power	960.72 W	992.21 W	966.86 W	**973.27 W**
Electricity consumption	23.06 kWh	23.81 kWh	23.20 kWh	**70.08 kWh**
Cost of electricity	1.66 €	1.72 €	1.48 €	**4.86 €**

Table 19: Residential scenario consumption (base case, summer)

4.1.3 Base case - winter

This case is equal to the one presented in Chapter 4.1.2 but for winter. Similarly to the summer case, two temperature setpoints have been defined, one for the night and another for the day.

- Night period (from 12pm to 8am): 19ºC
- Daytime (the rest of the day): 22ºC

The power consumed by the HVAC is shown in the top of Figure 39.

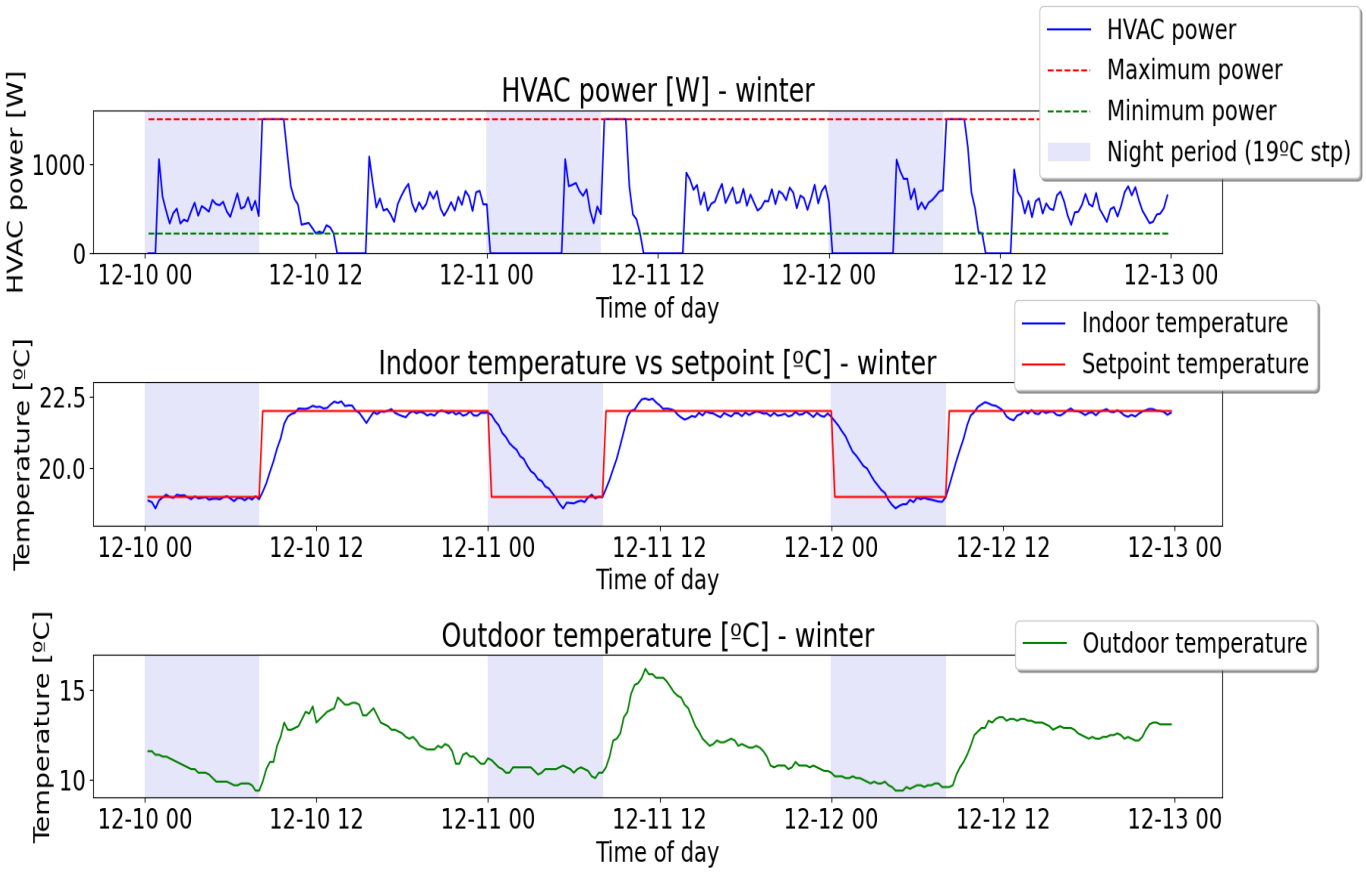

Figure 39: HVAC power consumption (residential base case, winter)

For the winter, we can see that the consumption pattern changes slightly in comparison to the summer. In the morning, when the temperature setpoint changes, the HVAC turns on at maximum power to increase the indoor temperature after the night. In a short period of time, the temperature arrives to the comfort one and the HVAC turns off. After some time, due to the low ambient temperatures, the indoor temperature starts decreasing and so the HVAC starts working again at medium powers until the night. During the night the HVAC is usually turned off in the beginning, but to avoid temperatures falling under 19ºC, it turns on after a while.

From the temperature evolution (middle of Figure 39) we can see a stable pattern where the indoor temperature follows the setpoint with little variation. The highest temperature difference is early in the morning when the HVAC starts working at maximum power to achieve the new setpoint of 22ºC and during the night just before the HVAC turns on to maintain the 19ºC.

The minimum, maximum and mean temperatures can be seen in the Table 20:

	Night 1	Day 1	Night 2	Day 2	Night 3	Day 3
Minimum Ti	18.60 ºC	19.18 ºCS	18.60 ºC	19.29 ºC	18.60 ºC	19.39 ºC
Maximum Ti	19.08 ºC	22.33 ºC	21.86 ºC	22.44 ºC	21.64 ºC	22.31 ºC
Average Ti	18.95 ºC	21.80 ºC	19.73 ºC	21.78 ºC	19.54 ºC	21.82 ºC

Table 20: Indoor temperature output (residential base case, winter)

The total consumption of the building, including the load and the HVAC can be seen in Figure 40 and is summarized in Table 21. In this case, the night and day consumptions are not so different. The peak consumption values are early in the morning where the HVAC is working at maximum power.

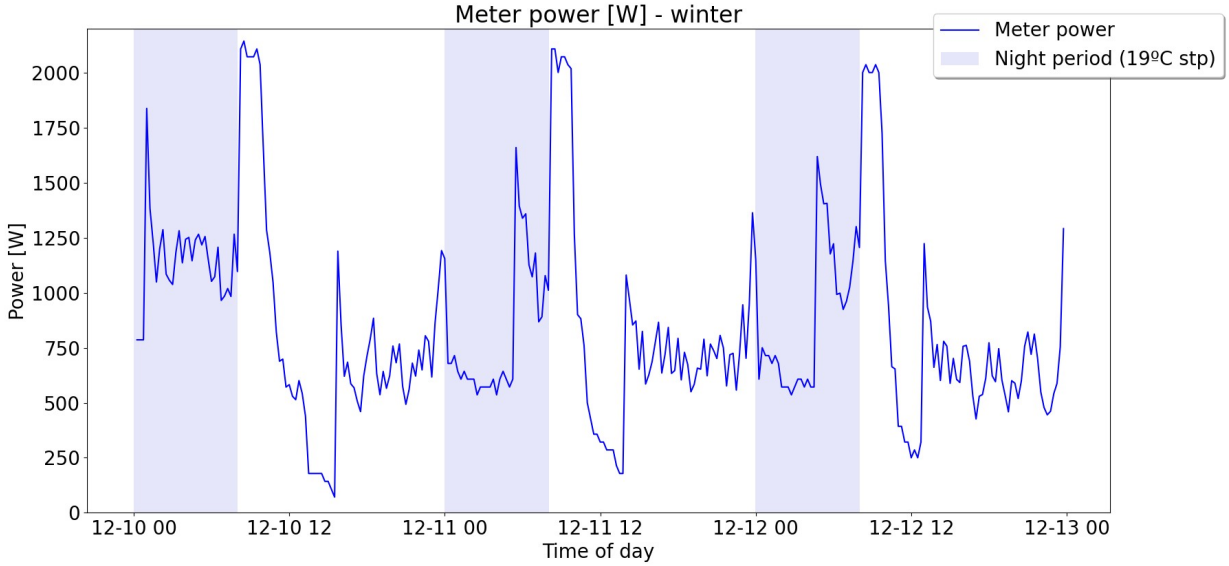

Figure 40: Meter power consumption (residential base case, winter)

	Day 1	Day 2	Day 3	Total
Load Mean Power	383.18 W	345.98 W	337.59 W	**355.65 W**
HVAC Mean Power	534.13 W	474.59 W	475.91 W	**494.94 W**
Meter Mean Power	917.31 W	820.58 W	813.51 W	**850.59 W**
Electricity consumption	22.02 kWh	19.69 kWh	19.32 kWh	**61.03 kWh**
Cost of electricity consumption	1.59 €	1.55 €	1.53 €	**4.66 €**

Table 21: Residential scenario results (base case, winter)

4.1.4 Case with aggregator - summer

In this case, the customer receives commands from the aggregator to modify the HVAC consumption. Instead of having two fixed temperature setpoints as in the previous case, for this scenario, the aggregator is the one defining these setpoints during flexibility activations. For the three day test, in Table 22, we can see the events that were created:

Event	Day	Start Time	Duration	Setpoint
1	1	08:00	30 minutes	23.5 ºC
2	1	18:30	30 minutes	23.5 ºC
3	2	12:00	1 hour	23.5 ºC
4	3	08:00	1 hour	23.5 ºC
5	3	18:00	1 hour	23.5 ºC

Table 22: Residential summer case events

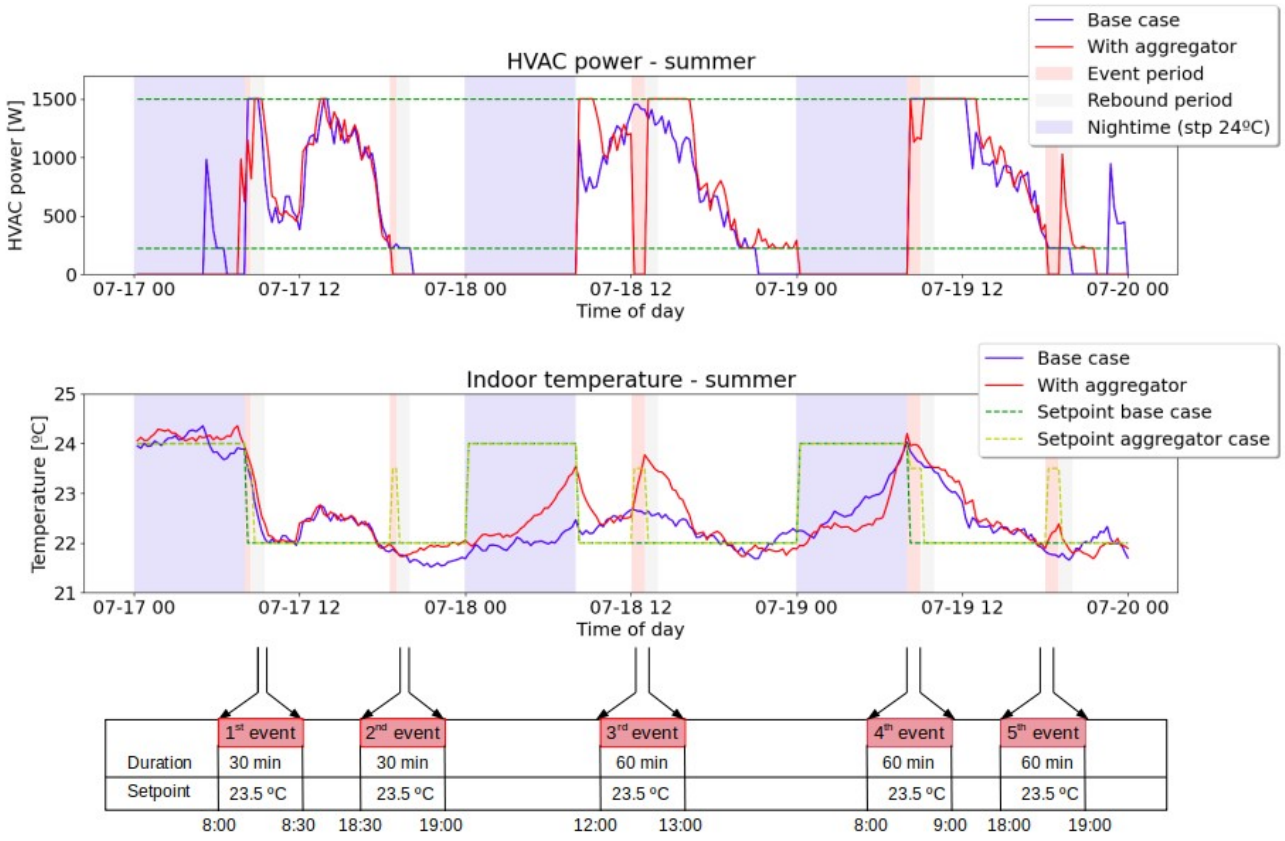

Figure 41: HVAC and indoor temperature evolution (residential summer, aggregator)

In the upper part of Figure 41 the difference in HVAC consumption compared to the base case can be seen. These variations are due to the following reasons:

- During the events defined by the aggregator, the temperature setpoint increase creates a reduction in the HVAC consumption. This decrease is less noticeable during the events created in the morning (events 1 and 4) than during the rest of the day (events 2,3 and 5). During the first hours of the morning, the indoor temperature is higher because the HVAC has followed a setpoint of 24ºC the previous hours. Therefore, even if the setpoint defined by the aggregator is 23.5ºC, the HVAC still has to turn on to achieve that temperature, but not at maximum power as the base case.
- After an event, there is a clear increase in the consumption, due to the rebound effect. When the event ends, the indoor temperature has increased and so, when the setpoint goes back to 22ºC, the HVAC has to increase its consumption.
- The rest of the differences are due to the factor of randomness included in the virtual HVAC indoor temperature calculation. In some cases, for example during the first night, there is a difference in the indoor temperature calculated by the model. Even if the difference is very low, it can be enough to change the consumption of the HVAC.

Regarding the indoor temperature, we see that, similarly to the base case, the indoor temperature follows the setpoint. However, because of the events defined by the aggregator, the indoor temperature increases during these periods. Once the event is over, the indoor temperature follows the base setpoint but it takes slightly longer to achieve it. This is specially noticeable during the third event, where the indoor temperature remains high for a while after the event. As mentioned previously, the rest of the differences are explained by the randomness factor of the model, which has created important differences, specially during the night of the second day.

Table 23 shows the indoor temperatures achieved each day. In parenthesis, the difference in degrees with respect to the base case is shown (green for colder temperatures, red for warmer and blue for equal). We can see how most times the temperatures for the aggregator scenario are higher. As discussed, a part of this is caused by the events and another by the stochasticity of the model. Even if the temperatures are higher, as will be seen in the KPIs in Section 4.1.6, this does not imply a loss of comfort for the user.

	Night 1	Day 1	Night 2	Day 2	Night 3	Day 3
Minimum Ti	23.96 ºC (+0.29 ºC)	21.73 ºC (+0.21 ºC)	21.93 ºC (+0.15 ºC)	21.74 ºC (+0.05 ºC)	21.95 ºC (-0.15 ºC)	21.68 ºC (+0.03 ºC)
Maximum Ti	24.36 ºC (=)	23.74 ºC (+0.23 ºC)	23.54 ºC (+1.08 ºC)	23.77 ºC (+1.10 ºC)	24.20 ºC (+0.17 ºC)	23.98 ºC (+0.14 ºC)
Average Ti	24.15 ºC (+0.14 ºC)	22.24 ºC (+0.13 ºC)	22.49 ºC (+0.47 ºC)	22.46 ºC (+0.21 ºC)	22.54 ºC (-0.25 ºC)	22.49 ºC (+0.09 ºC)

Table 23: Indoor temperature output (residential summer, aggregator)

The differences in the HVAC consumption, end up affecting the consumption read in the meter. Figure 42 shows the difference in meter power, calculated as the base consumption minus the aggregator case one. We can see in that the goal of the aggregator, which is to reduce the consumption during the time periods defined has been achieved. Depending on the event this reduction is different (see Figure 43). The rebound effect is also noticeable during the first hours after the event.

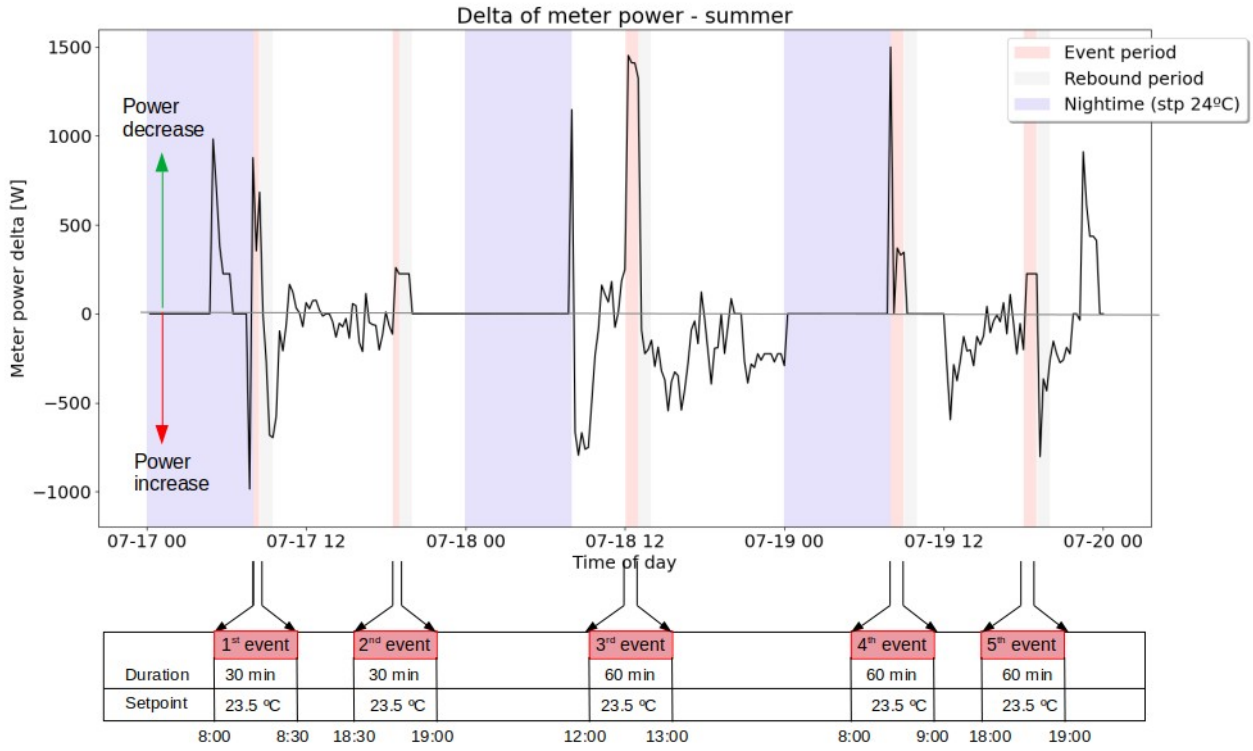

Figure 42: Difference in power consumption r/ base case (residential summer)

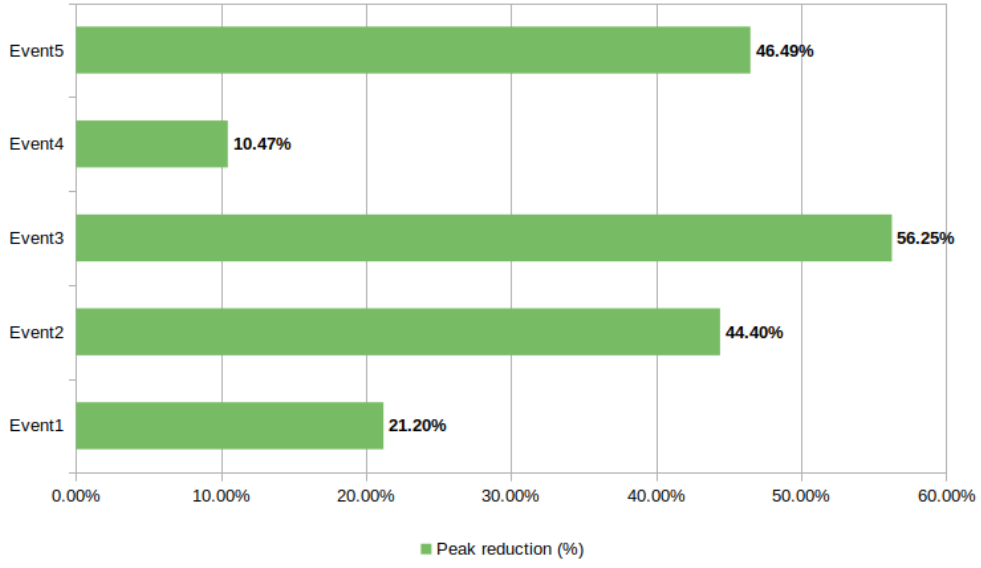

Figure 43: Reduction of power during events (residential, summer)

Table 24 shows the average power consumed by each load, the electricity consumed and its corresponding cost. In parenthesis, the difference with respect to the base case is shown (green for lower consumption or cost and red for higher ones).

	Day 1	Day 2	Day 3	Total
Load Mean Power	499.63 W	497.40 W	457.22 W	**484.75 W**
HVAC Mean Power	446.11 W	560.91 W	516.84 W	**507.95 W**
Meter Mean Power	945.73 W	1058.30 W	974.05 W	**992.70 W**
Electricity consumption	22.70 kWh (-0.36 kWh)	25.40 kWh (+1.59 kWh)	23.38 kWh (+0.17 kWh)	**71.47 kWh (+1.40 kWh)**
Cost of active electricity consumption	1.65 € (-0.01 €)	1.93 € (+0.22 €)	1.54 € (+0.05 €)	**5.12 € (+0.26 €)**

Table 24: Residential scenario results (summer, aggregator)

4.1.5 Case with aggregator - winter

Similarly to the summer case, when the communication with the aggregator is allowed, the commands received modify the HVAC consumption. For the three day test, Table 25 shows the events that were created:

Event	Day	Start Time	Duration	Setpoint
1	1	08:00	30 minutes	20.5 ºC
2	1	18:30	30 minutes	20.5 ºC
3	2	12:00	1 hour	20.5 ºC
4	3	08:00	1 hour	20 ºC
5	3	18:00	1 hour	20 ºC

Table 25: Residential winter case events

In this case, three of the events included a temperature setpoint of 20.5ºC (marked in orange in the following graphs) and two of 20ºC (marked in red). Figure 44 shows the effect of the aggregator in the HVAC output and indoor temperature, compared with the base case.

As mentioned in the previous scenario, these differences are due to the effect of the aggregator and the randomness factor of the thermal zone model. Regarding the events created by the aggregator, we see that during the events 1 and 3 the HVAC is not able to reduce the power consumption for different reasons:

- During the first event, even if the setpoint sent by the aggregator is lower than the one in the base case, the HVAC still needs to work at maximum power to achieve the set indoor temperature.
- During the third event, the HVAC was already turned off so the event sent by the aggregator did not have any effect.

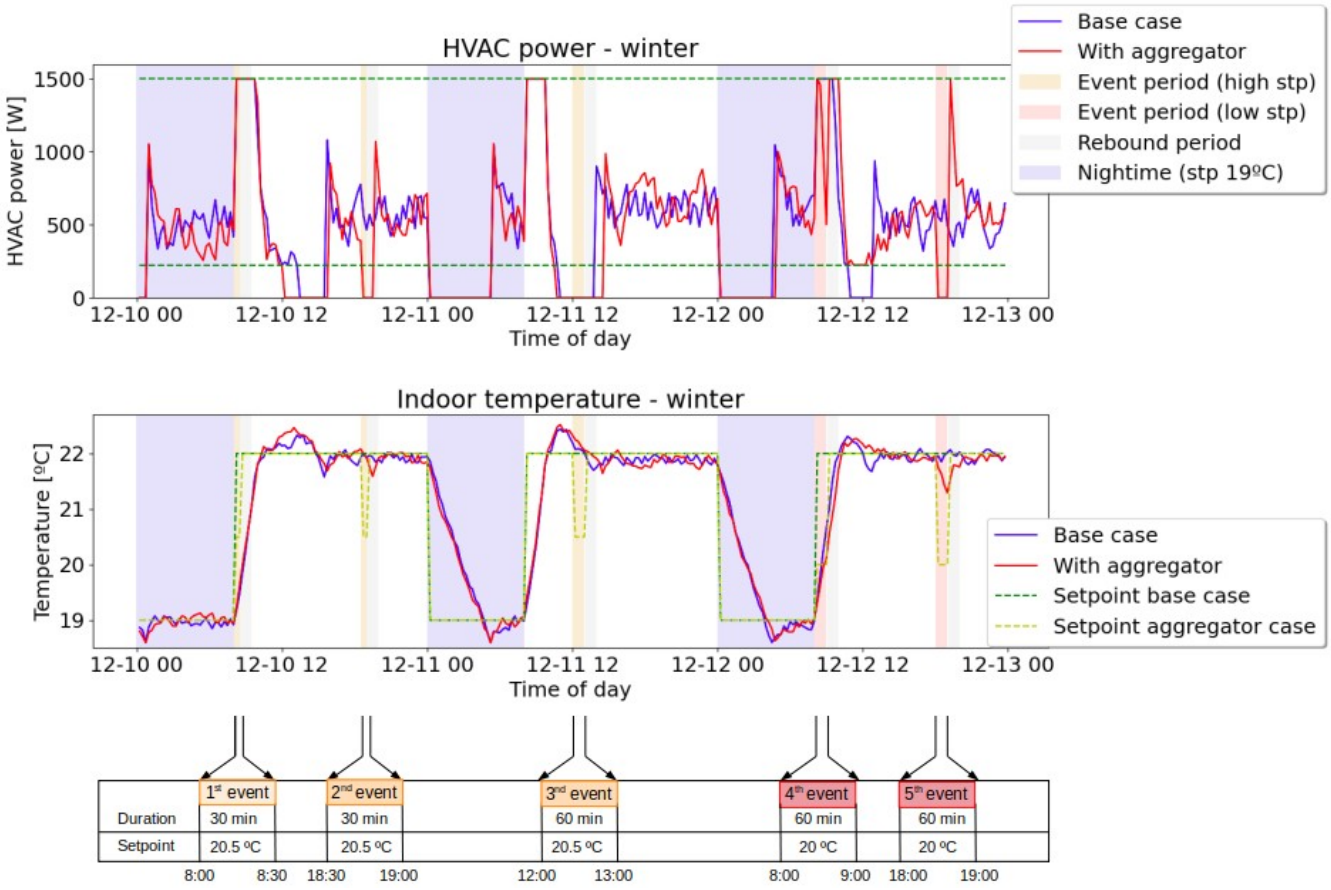

Figure 44: HVAC power consumption (residential winter, aggregator)

Regarding the indoor temperature, we see in the lower part of Figure 44, that it is considerably more stable than in the summer case. We only notice slight temperature drops during the events, when the HVAC reduced its consumption, specially during the last one. However this temperature difference is only of 0.5ºC approximately and it is quickly restored. During the second event, the lower temperature is achieved once the event has ended, since the starting temperature was higher than in the base case.

A summary of the indoor temperatures is shown in Table 26 (in green increases, red decreases and blue equal temperatures). In this case, we see that because of the randomness included, the temperatures are sometimes higher than the base case.

	Night 1	Day 1	Night 2	Day 2	Night 3	Day 3
Minimum Ti	18.60 ºC (=)	19.23 ºC (+0.05 ºC)	18.60 ºC (=)	19.43 ºC (+0.15 ºC)	18.63 ºC (+0.03 ºC)	19.33 ºC (-0.06 ºC)
Maximum Ti	19.13 ºC (+0.05 ºC)	22.46 ºC (+0.13 ºC)	21.74 ºC (-0.12 ºC)	22.52 ºC (+0.08 ºC)	21.60 ºC (-0.05 ºC)	22.27 ºC (-0.04 ºC)
Average Ti	18.97 ºC (+0.02 ºC)	21.83 ºC (+0.04 ºC)	19.67 ºC (-0.06 ºC)	21.80 ºC (+0.02 ºC)	19.54 ºC (+0.01 ºC)	21.75 ºC (-0.08 ºC)

Table 26: Indoor temperature output (residential winter, aggregator)

For the three events, that did affect the HVAC, we see a power reduction, as observed in the Figure 45. Following these events, we can also notice the rebound effect in the consumption.

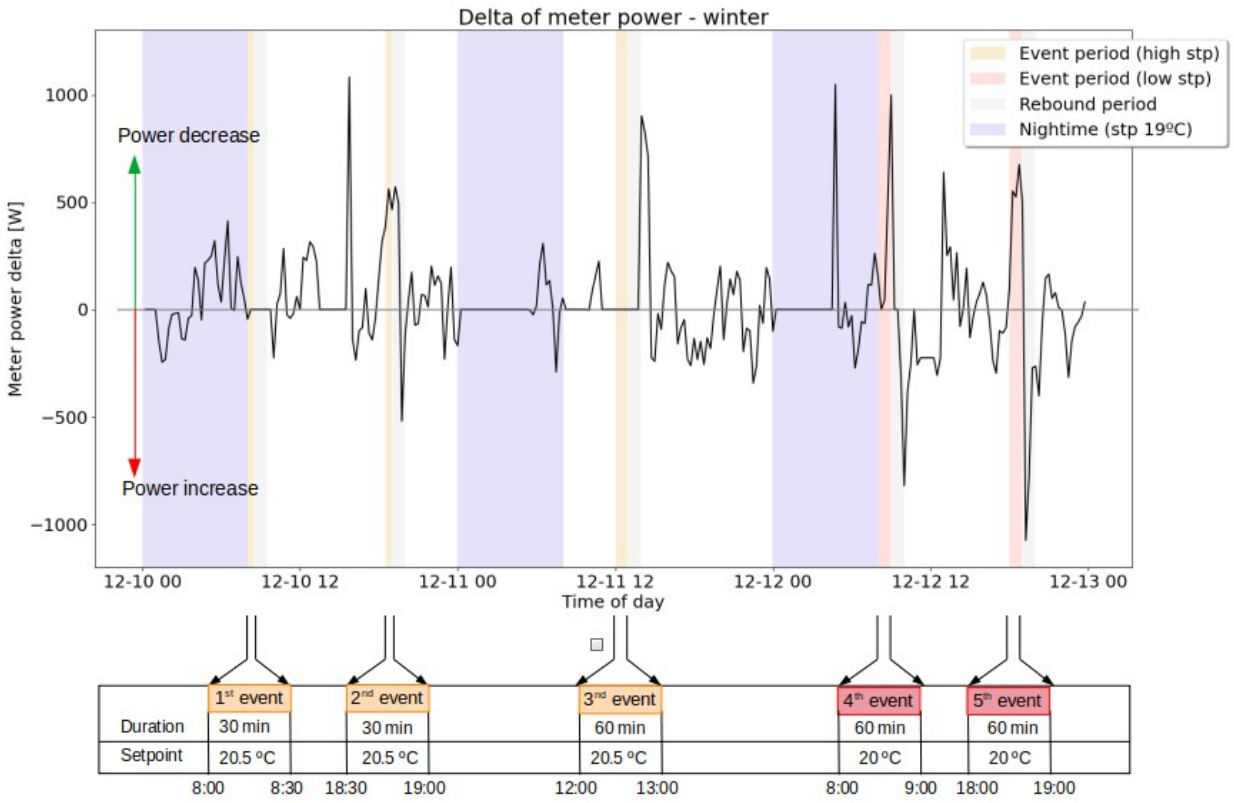

Figure 45: Difference in power consumption r/ base case (residential winter)

So, as a summary, we see in Figure 46 the power reduction of each event, highlighting events 1 and 3 where no reduction was achieved. Events 2 and 5 caused the greatest demand change.

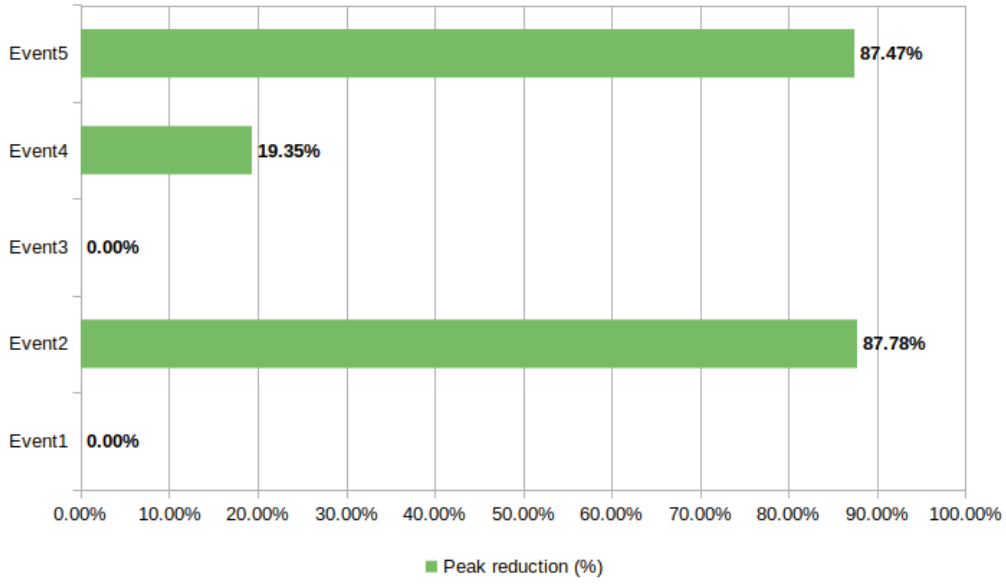

Figure 46: Reduction of power during events (residential winter)

Table 27 shows the average power consumed by each load, the electricity consumed and its corresponding cost. In parenthesis, the difference with respect to the base case is shown (green for lower consumption or cost and red for higher ones).

	Day 1	Day 2	Day 3	Total
Load Mean Power	383.18 W	345.98 W	345.98 W	**355.65 W**
HVAC Mean Power	472.58 W	458.89 W	488.26 W	**473.19 W**
Meter Mean Power	855.77 W	804.87 W	804.87 W	**828.84 W**
Electricity consumption	20.54 kWh (-1.48 kWh)	19.32 kWh (-0.38 kWh)	19.61 kWh (+0.29 kWh)	**59.47 kWh (-1.56 kWh)**
Cost of active electricity consumption	1.43 € (-0.16 €)	1.51 € (-0.04 €)	1.54 € (+0.02 €)	**4.48 € (-0.19 €)**

Table 27: Residential scenario results (winter, aggregator)

4.1.6 Flexibility potential of the residential consumer

After running the different scenarios, we can analyse the impact of the aggregator in the consumption and comfort of the user, both for summer and winter. For this purpose we analyse the KPIs defined in the beginning of this section, shown in Table 28.

Nº	KPI	Summer	Winter
1	Total energy consumption change	+2.0% (+1.40 kWh)	-2.56% (-1.56 kWh)
2	Total energy cost change	+5.30% (+0.26 €)	-4.02% (-0.19 €)
3	Peak load reduction	- 32.53 % (- 566.90 W)	-28.05% (-301.55 W)
4	Rebound effect	+12.92% (+201.06 W)	+15.78% (+157.39 W)
5	Reliability index	100%	100%
6	Thermal comfort level	0%	0%

Table 28: KPIs for the residential scenario

The KPIs of interest for the aggregator (3 and 4) have resulted in expected values. On one hand, the peak loads have been reduced around a 30% and the rebound effect has been calculated to be around a 15%. For the summer case, the peaks are reduced in a higher degree than in the winter scenario and the rebound is lower. However, these differences between seasons are relatively small. Regarding the reliability index (KPI 5), we see that the user did not loose the connection with the aggregation platform at any point and so was able to obtain the correct temperature setpoint at all times. However, as we saw during some of the winter events, the aggregator did not calculate the flexibility correctly and so, the events sent did not impact the consumption.

The rest of the KPIs measure the impact on the user. First, we see that the total energy consumption is not affected in a big degree. However, in the summer scenario we see an increase of 2% in the consumption whereas during the winter scenario the consumption drops

2.6%. These changes in consumption affect the cost of the electricity consumed. During the summer, even if the loads are moved to a different period, this is not reflected in a change in the cost. The reason behind this is that the periods to which the load is moved, still fall under the peak period for the tariff (as shown in Table 15). The aggregator tries to solve grid congestions that may appear at any time or contribute to grid balancing. For this reason, the electricity cost for the customer may even increase (summer scenario). Defining the adequate incentives is a key task of the aggregator to counteract this price increase.

Regarding the thermal comfort, it is noticeable that the number of intervals out of comfort, as measured by KPI6, is 0 for both winter and summer. We can conclude that the modification in the HVAC consumption introduced by the aggregator did not compromise the comfort of the user as the temperature was kept in the comfort range during events and rebound periods.

4.2 Scenario 2: Prosumer

To test how distributed energy resources with energy storage devices can provide flexibility, a prosumer has been defined. The term prosumer is used to describe a user that can both generate and consume electricity. This customer, as shown in Figure 47, has a PV panel to produce electricity, a battery to store it and an uncontrollable load. In this case, only a summer scenario has been carried out and the possible effects of having lower PV have been studied by including a day with very low PV generation. The PV, load and battery features are summarized in Table 29 and Table 30. A screenshot of the SCADA Demo is included in Annex G.

ELEMENT	MAXIMUM POWER	TYPE
PV	4 000 W	Emulated
Load	3 100 W	Emulated

Table 29: Prosumer elements (1)

ELEMENT	MAXIMUM POWER	CAPACITY	SoC RANGE	TYPE
Battery	10 000 W	5 000 Wh	20 – 80%	Real

Table 30: Prosumer elements (2)

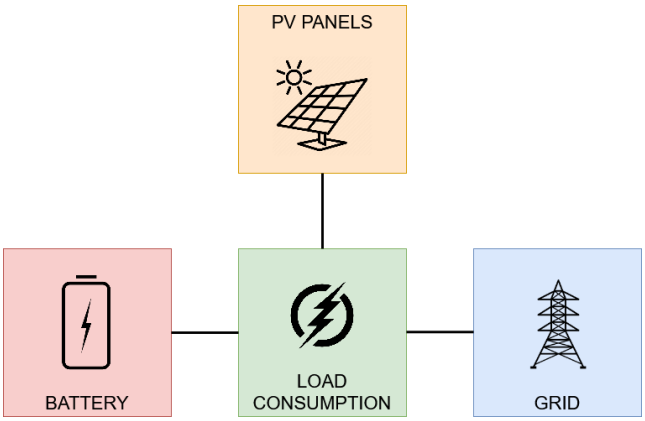

Figure 47: Prosumer diagram

The goals of this test are:

- Validate the battery control developed
- Verify the connection of the emulation cabinets and the battery with the SCADA
- Analyse the effect of the aggregator in the battery and determine the flexibility services that a prosumer can provide.

4.2.1 Power profiles

The prosumer test runs using the laboratory emulation cabinets for the PV generation and the load consumption and so, the frequency of the data has been reduced to 1s. The PV and load setpoint profiles, provided by IREC, are shown in the Figure 48.

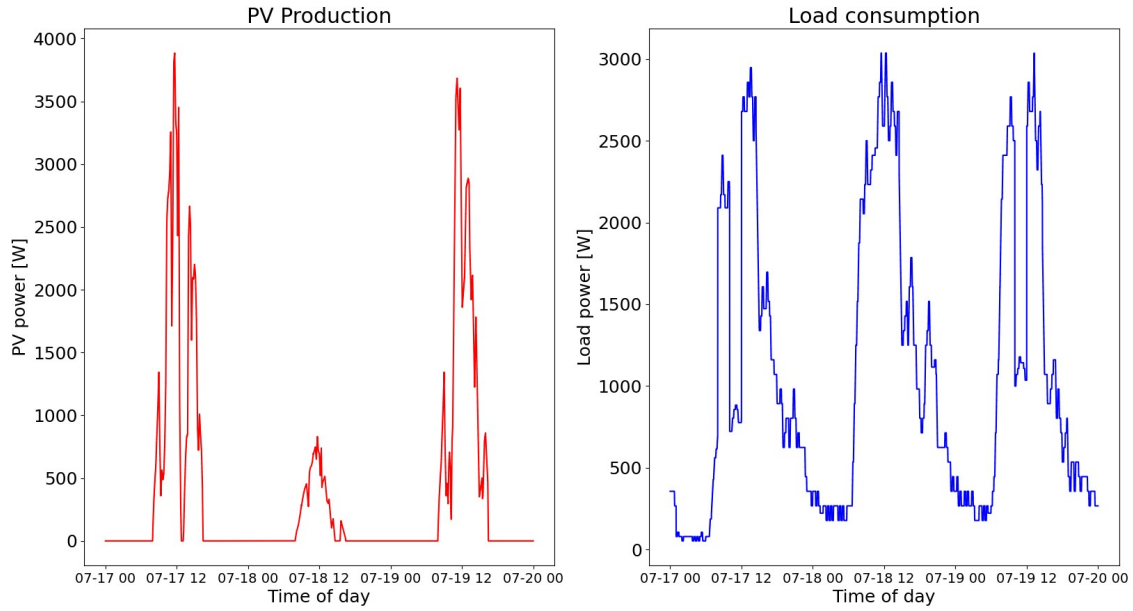

Figure 48: PV Production and load profile for the prosumer scenario

4.2.2 Base case

For the base case, the battery charge and discharge setpoints are established by the control explained in Chapter 3.4.1. Figure 49 shows the power balance between all the elements in the system and the SoC of the battery for the three days. In order to visualize properly the variables, the data has been grouped in intervals of 5 minutes. As a reminder, the sign criteria is as follow:

- PV generation: negative
- Load consumption: positive
- Battery: positive if charging, negative if discharging
- Meter: positive if exporting (selling), negative if importing (buying)

In the figure we can see what source is being used to provide the necessary power for the load at each moment (grid, PV or battery) by looking at the negative side of the graph. In addition we can also see when the battery is charging or discharging (in green) depending if it is on the positive or negative side of the graph, respectively. Finally, positive power of meter (in grey) indicate that there is power being injected to the grid. In the SoC figure, D refers to discharging, C to charging from the PV and G to charging from the grid.

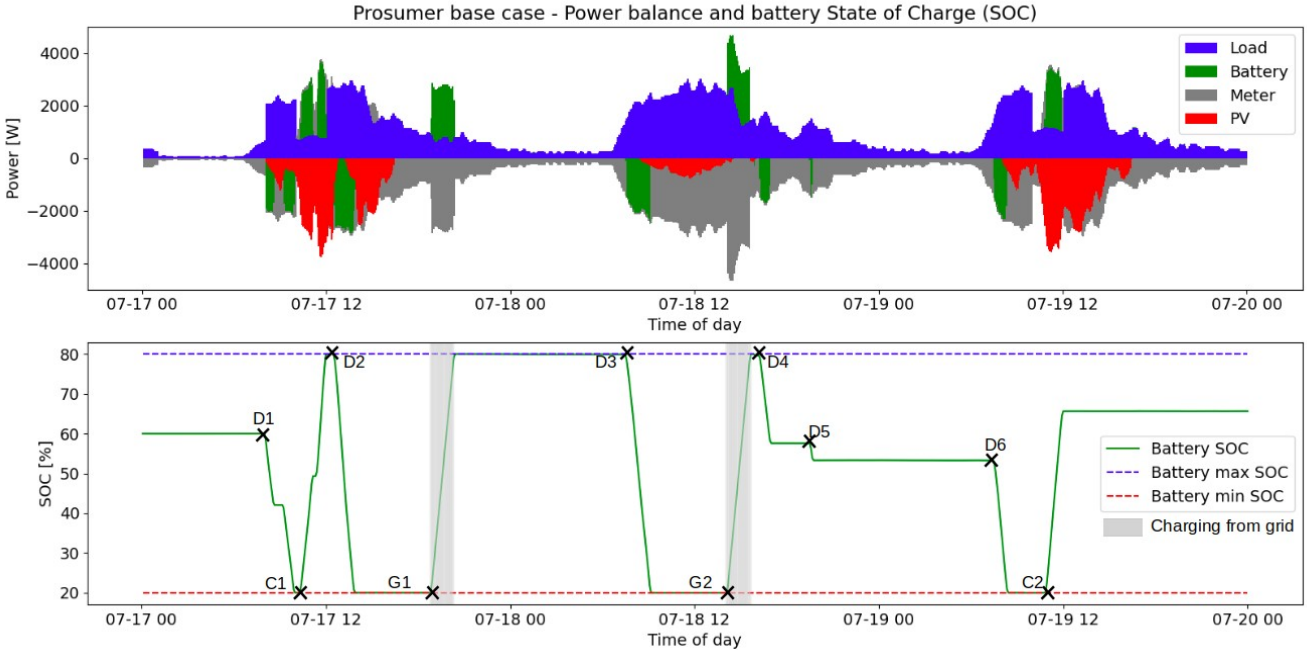

Figure 49: Prosumer base case results

The first thing to point out is that the battery did not follw any setpoints between -1500 and 1500 W and, therefore, only charged or discharged in specific moments of the test. This was caused by the characteristics of the real battery of the laboratory and can be seen in Figure 50. The dashed line represents the 1500 limit below which the battery did not follow any setpoints.

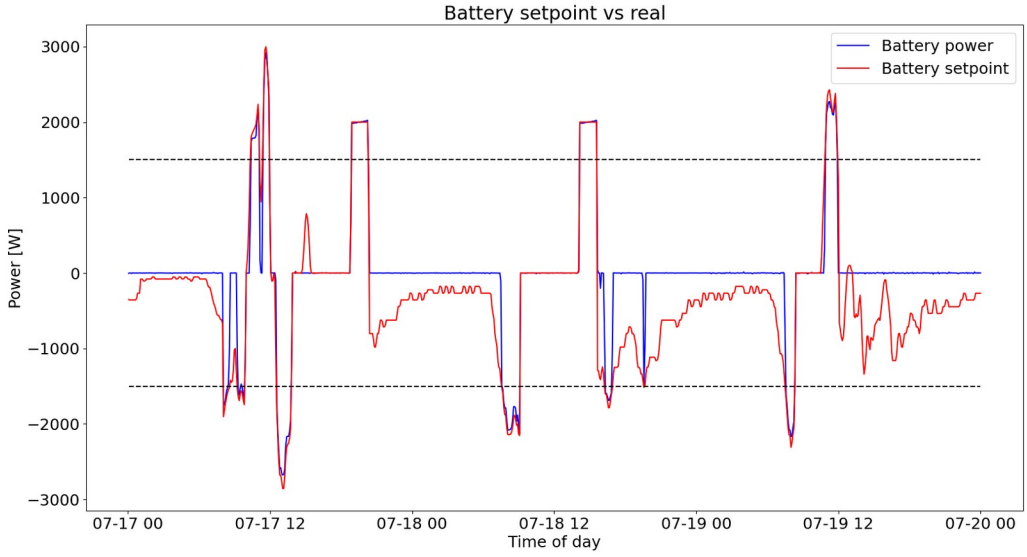

Figure 50: Battery setpoint vs real one

The main points we can highlight from the base case are:

- Charges from PV (marked as C in Figure 49): the battery only charged twice from the solar generation surplus. In both cases, this happened when the PV generation was high and the load decreased for a period of the morning (before midday of days one and three). The first day (during C1), we see a full battery charge (from minimum to maximum SoC) and a small time frame when the charging stopped because the setpoint dropped below 1500 W and the battery did not follow it. The third day (C2), the battery also started charging at minimum SoC but only arrived to 65% SoC because the load at this point increased back.
- Charges from grid (marked as G in Figure 49): the battery started to charge from the grid twice. The first time (G1) was during the first day, at 18:45, after spending 5 hours in minimum SoC state. Even if there was PV surplus at moments in this period, the setpoint sent to the battery was too low and the battery did not get charged. The second time that grid charging was necessary, was during the second day (G2). This day is the one with the lowest PV generation and, even during peak sun hours, there was not enough PV to charge the battery. So, the battery after 5 hours in minimum SoC (from 9h to 14h) started charging from the grid at around 14h.
- Discharging events (marked as D in Figure 49): if the battery was charged over 20% and the PV was low, the battery only discharged if the load was high enough to provide the setpoint required. The first discharge (D1) takes place early in the morning when there is not enough sun and the load increases. At the middle of the discharge we see that it stops for a brief period of time, caused by a slight increase in PV production. Similarly to D1 we see that during the morning of the second day (D3) and third day (D6) the battery gets discharged. The second discharge (D2) takes place during the first day after midday. There is a sudden drop of PV production, possibly because of a change in the weather, and the battery, that had been charged right before, gets discharged. The forth discharge (D4) happens right after the charge from grid G2. The discharging stops after some time due to a decrease in the load and returns only for a few minutes during D5. Discharges D2 and D3 correspond to full cycles (from maximum to minimum SoC), while the rest are only partial.

Table 31 show the electricity exported (sold) to and imported (consumed) from the grid. The net cost is calculated as the cost of consumption minus the revenue from exportations.

	Day 1	Day 2	Day 3	Total
Electricity exported	1.13 kWh	0.00 kWh	0.51 kWh	**1.64 kWh**
Cost of electricity exported	0.03 €	0 €	0.02 €	**0.05 €**
Electricity imported	12.93 kWh	25.60 kWh	12.76 kWh	**51.19 kWh**
Cost of electricity imported	1.22 €	2.03 €	0.91 €	**4.16 €**
Net cost of electricity (Imp - Exp)	1.19 €	2.03 €	0.89 €	**4.11€**

Table 31: Prosumer exported and imported power (base case)

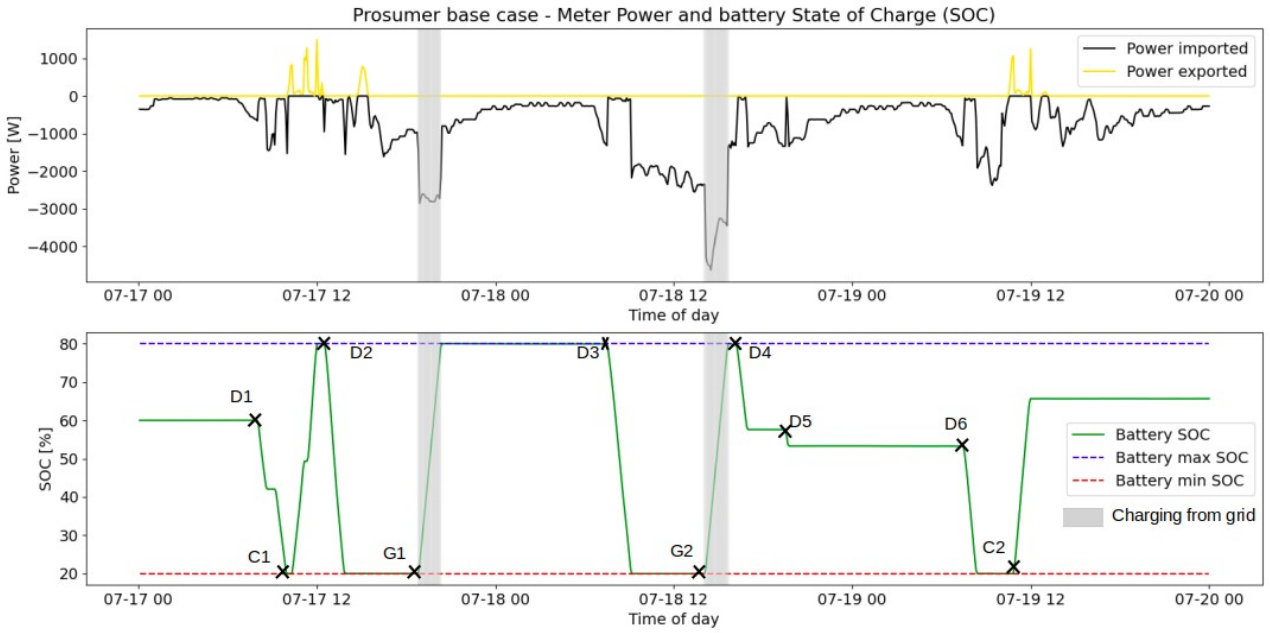

Figure 51: Exported and imported electricity (prosumer, base case)

From Figure 51 we can extract some behaviours. The peak consumptions take place during the charges from the grid. Overall, the consumption during the day is higher, as the load increases. In particular, the second day is the one with highest imports, caused by a lower PV generation.

Whenever the battery is able to discharge we see that, from the grid perspective, the consumption of the user reduces even if the real load is high. For example, during the discharges marked as D4 and D6, the grid consumption drops to zero. At these periods the load is high (see Figure 49) but it is being supplied by the battery discharge and not from the grid.

The moments when there is PV surplus and the battery is not able to charge, either because it is in maximum SoC or because the setpoint is lower than 1500 W, there is electricity exported to the grid. However, the electricity exported represents a small value compared to the imported one.

The self-consumption factor obtained for the base case is 33.14%, meaning that around one third of the load was supplied with energy generated in the PV panels directly or by discharging the battery.

Load consumption:	76.68 kWh
Electricity imported:	51.27 kWh
Self consumption factor:	**33.14 %**

4.2.3 Case with aggregator

The aggregator is able to overwrite the setpoints of the battery either to start charging, discharging or impose neither of them. Table 32 shows the events created for the three days of test:

Event	Day	Start Time	Duration	Power Setpoint
1	1	18:30	60 minutes	0
2	2	14:00	60 minutes	0
3	3	20:00	30 minutes	-2000 W

Table 32: Prosumer case events

Therefore, the aggregator established a setpoint of zero for two events, not allowing either charges or discharges of the battery during these periods when, during normal operation, the battery would charge. For the third event, the aggregator imposed a discharge of 30 minutes for the battery at 2000 W.

We can see how the activations received affected the battery setpoint and therefore the charge and discharge events. Figure 52 shows the battery power and SoC for this case and the base case.

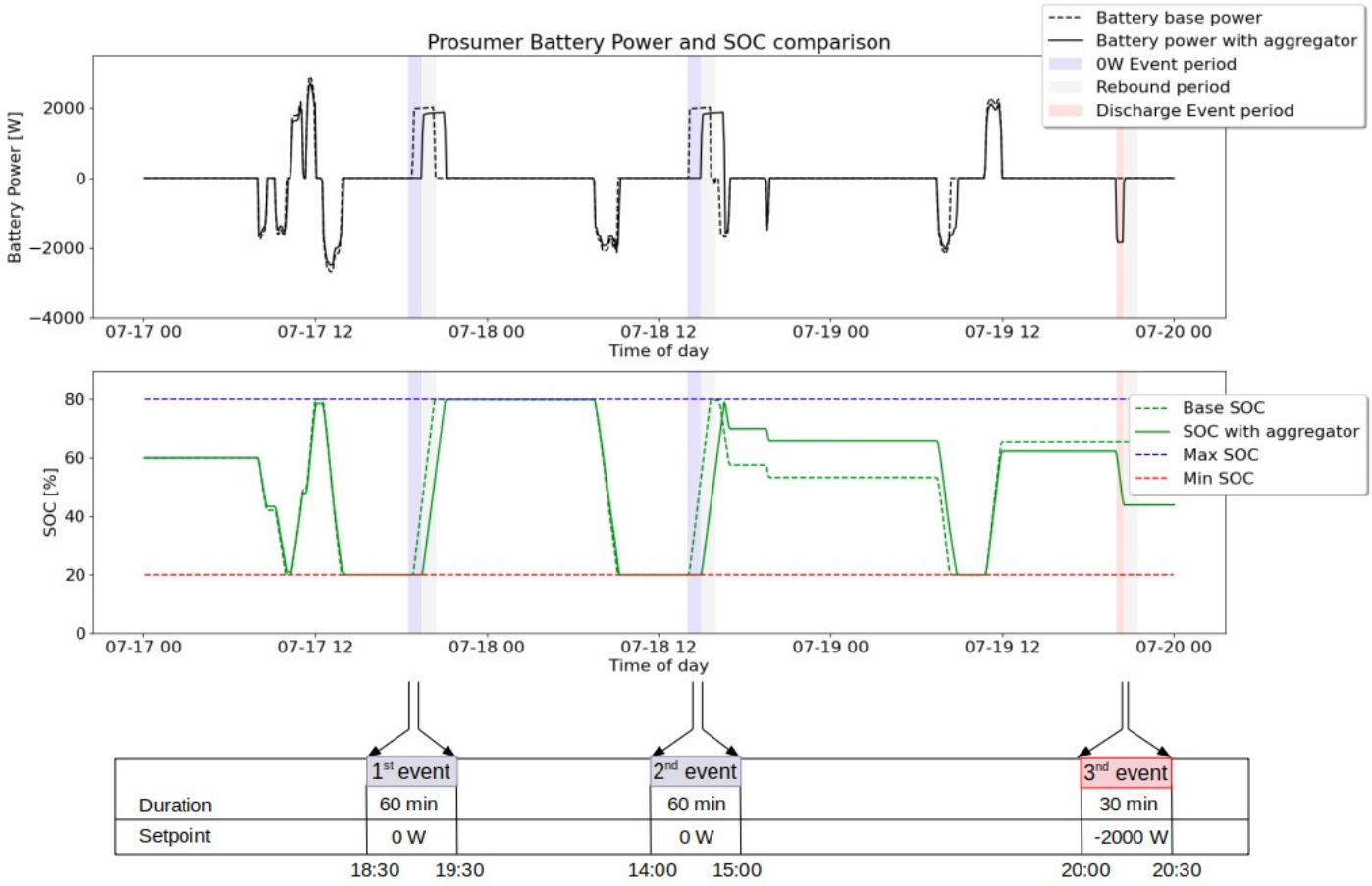

Figure 52: Prosumer battery power and SoC (aggregator case)

- Event 1: the first event was received when the battery was about to start charging from the grid. As we saw in the base case, during these moments the electricity imported from the grid arrives to peak values. The aggregator, by sending this event was able to postpone the grid charge to a latter moment.

- Event 2: this event, like to the first one, postponed the grid charge to another period.
- Event 3: the third event occurs during the last day of the test at 20:00. At this moment the battery is sightly over 60% SoC and is not able to charge or discharge.The aggregator sets a negative setpoint on the battery to force discharge. As will be seen in Figure 55, the load demand is lower than the power being discharged by the battery and therefore, the surplus electricity is sent to the grid. The battery ends with a lower SoC after the event, around a 44%.

We see an important difference in the behaviour of the battery in the morning of the third day. A zoom of this period can be seen in Figure 53 where the aggregator case is shown in red and the base case in blue. After the second event, the battery has a higher SoC (around 65%) than in the base case (around 50%). Then, in the morning of the third day, around 7:30, the battery starts discharging. In the base case, the battery arrives at minimum SoC (point 1) earlier than in the aggregator (point 2) and for this reason, the battery setpoint is set to zero in advance.

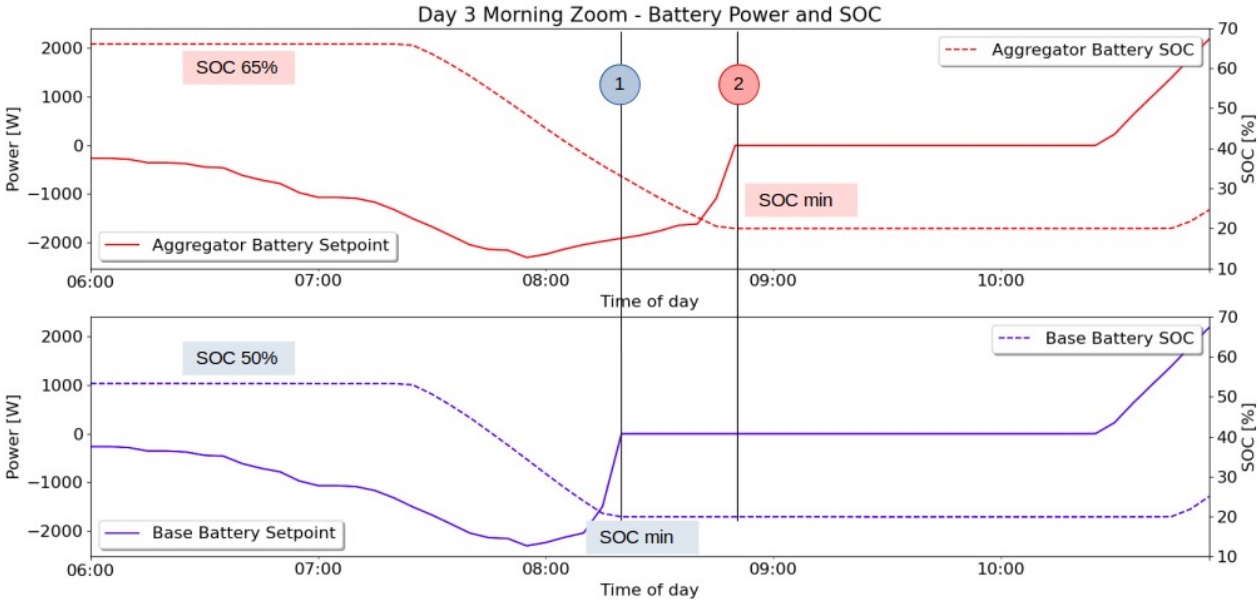

Figure 53: Prosumer Day 3 Zoom Morning

The remaining differences in battery power are due to using a real battery. Even if the setpoint sent is the same for base and aggregator case, the battery does not output the same exact values. For example, in the midday of the third day, even if the setpoint sent to the battery is the same, the final SoC of the base case is higher. Figure 54 shows how for equal setpoints, the final SoC is different for the aggregator and the base setpoint.

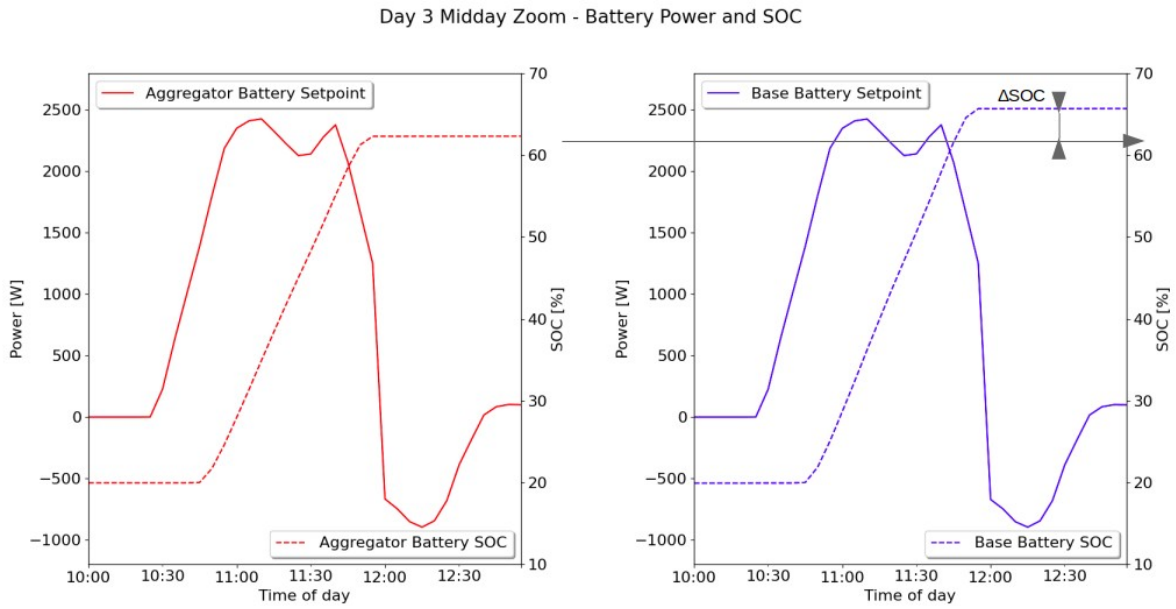

Figure 54: Prosumer Day 3 Zoom midday

Due to the commands received, the energy exported to and imported from the grid is different both in magnitude and in timing. Figure 55 shows the difference in power imported and exported between the case with aggregator and the base case. Positive values represent an increase of power for the aggregator case compared to the base case and negative values, the opposite.

There is a clear decrease in consumption during the first and second event, where the battery was not allowed to charge from the grid. Right after those events is when the charging starts. However because the battery was still charging in the base case, we do not see a consumption increase until the charge stops in the base scenario.

After the second event, we see an additional increase in the electricity imported. At that time, in the base case the battery gets discharged for a period until the setpoint is below 1500 W where it stops and the load is supplied by the grid. Because the battery grid charge has been postponed by the aggregator, in this case, the battery starts discharging latter on and provides a smaller amount of electricity until arriving to the 1500 W limit. However, this also means that the battery is left with a higher SoC and so, eventually it will be able to discharge more electricity as we saw on Figure 53. This higher discharge reduces the power imported from the grid during the morning of the third day.

The third event creates both a decrease in consumption and an increase in power exported to the grid. In the base case during this period, all the load was covered from the grid (there was no PV and the battery setpoint was below 1500 W, not being able to discharge). As the aggregator sets a discharge on the battery, some part of it goes to fully supply the load and the remaining gets injected to the grid.

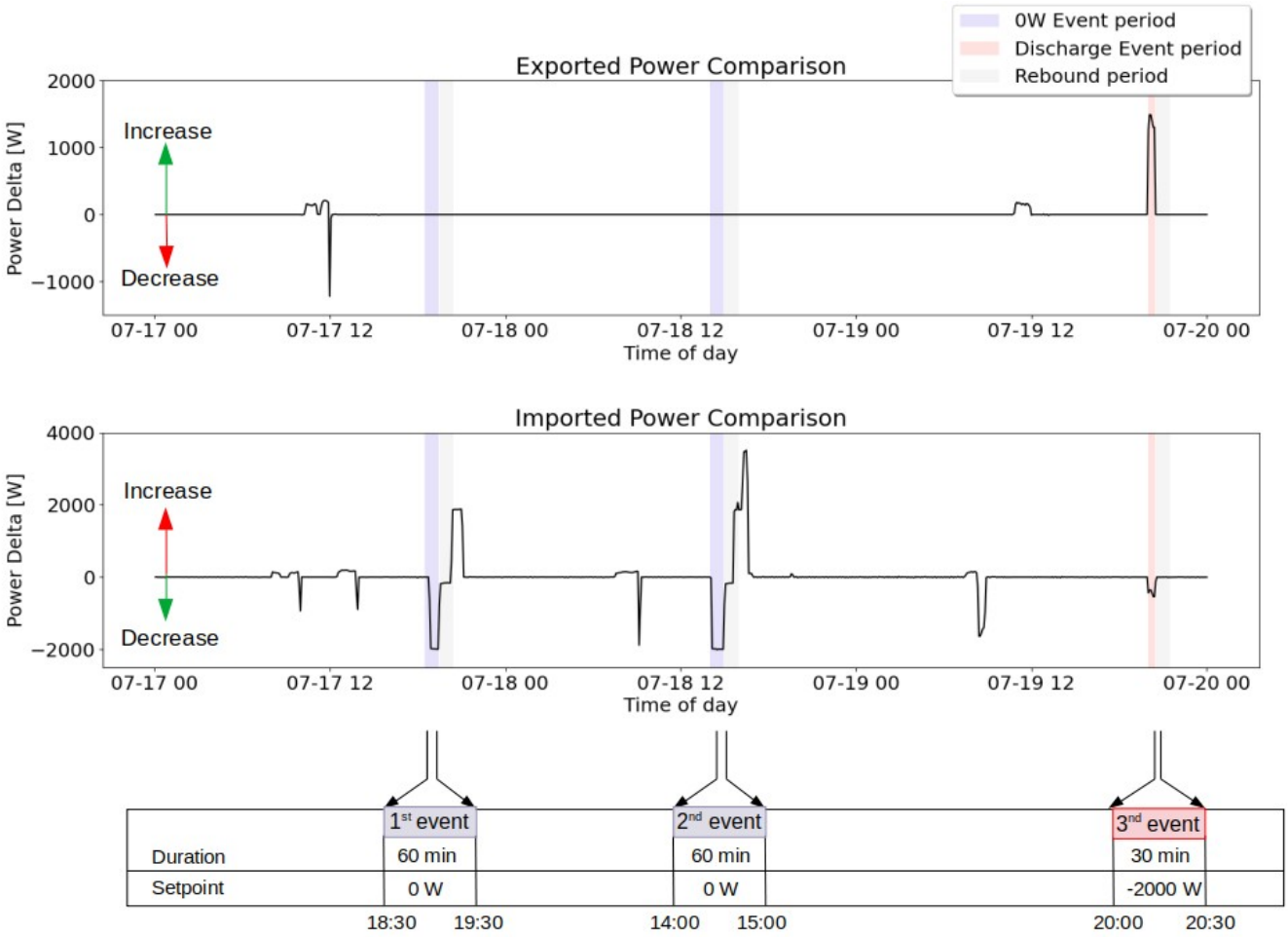

Figure 55: Prosumer meter power comparison

Figure 56 shows the reduction in peak power for each event.

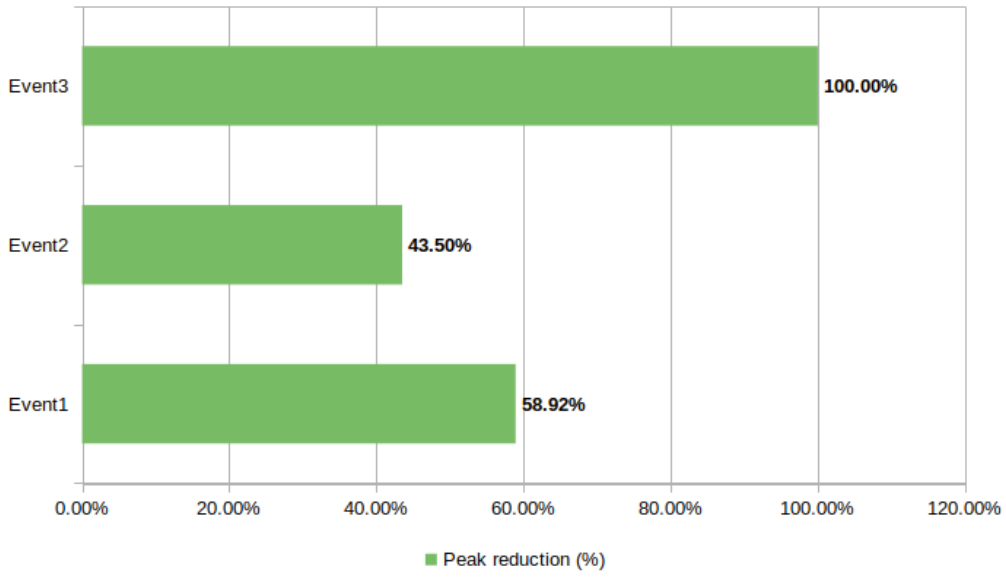

Figure 56: Reduction of power during events (prosumer)

Table 33 summarizes the electricity imported, exported and the cost. In parenthesis the comparison with the base case is shown (green for lower consumption or higher electricity sold, red for the opposite and blue for equal values). The net cost is calculated as the cost of consumption minus the revenue from exportations.

	Day 1	Day 2	Day 3	Total
Electricity exported	1.25 kWh (+0.12)	0.00 kWh (=)	1.36 kWh (+0.85)	**1.64 kWh (+0.97)**
Cost of electricity sold	0.04 € (+0.01)	0 € (=)	0.04 € (+0.02)	**0.08 € (+0.04 €)**
Electricity imported	13.08 kWh (+0.15)	26.22 kWh (+0.62)	11.88 kWh (-0.88)	**51.19 kWh (-0.1)**
Cost of electricity imported	1.23 € (+0.01)	2.10 € (+0.07)	0.91 € (-0.05)	**4.16 € (+0.03)**
Net cost of electricity	1.19 € (=)	2.10 € (+0.07 €)	0.82 € (-0.08 €)	**4.11 € (=)**

Table 33: Prosumer electricity cost and consumption (aggregator)

The self consumption factor obtained is 33.24%, almost equal to the base case one.

Load consumption:	76.68 kWh
Electricity imported:	51.19 kWh
Self consumption factor:	**33.24 %**

4.2.4 Flexibility potential of the prosumer

The KPIs defined in the beginning of the Chapter 4 are shown in Table 34.

	KPI	Summer
1	Total energy consumption change	-0.20% (-0.1 kWh)
2	Total energy cost change	0%
3	Peak load reduction	-50.73% (-1291.35 W)
4	Rebound effect	+17.82% (+364.94 W)
5	Reliability index	100 %
7	Self consumption factor	+0.39 % (+0.13%)

Table 34: KPIs for the prosumer scenario

The total energy consumption (KPI1) for the period is slightly lower, mainly due to the discharge imposed on the battery during the last event. The KPI2 shows that the aggregator did not affect the electricity cost of the customer. One one hand, there is an increase in the energy exports but this gets balanced by an increase in consumption cost. This increase is caused by the time shift of battery discharge as a consequence of the second event. The discharge instead of

taking place in off-peak periods (base case) occurs in peak periods, increasing the cost of electricity.

Where we do see a bigger impact of the aggregator is in the peak load reduction (KPI3) and rebound effect (KPI4). Peak powers are reduced during the three events sent, with an overall reduction of almost 51%. In contrast, the energy consumed the hour after the events is 18% higher. It is worth noticing that the KPI used to measure the rebound effect only considers the hour after the event. However, as explained in the previous section, the commands of the aggregator in the battery operation create longer term effects. For example, as we saw for the third event, the electricity that the battery is not allowed to discharge at some moment because of the commands from the aggregator, gets discharged later on. For this reason, the self consumption factor (KPI7) which takes the whole period for its calculation, does not experience big changes.

Finally, the test developed in this case received correctly all the setpoints from the aggregator and thus the reliability index (KPI5) is 100%. And, unlike in the residential scenario, all the activations received created a change in the consumption.

5 OpenADR Integration

In order to standardize the communication between the SCADA and the aggregation platform OpenADR (see Chapter 1.4.4) is an interesting protocol to be used in future projects. As a previous step, OpenADR tests have been carried out, using the openleadr module in Python. Openleadr is a tool specifically tailored to implement both client and servers (VENs and VTNs) in Python.

5.1 Communication between VTNs and VENs

In this first section, the interaction between the VEN and the VTN is explained. Before entering into detail, a list of the exchanged messages is presented in Table 35.

Message	Description	Sent by
oadrQueryRegistration	Used to get information from the VTN	VEN to VTN
oadrCreatedPartyRegistration	Used to send information or confirm registration	VTN to VEN
oadrCreatePartyRegistration	Used to request registration	VEN to VTN
oadrRegisterReport	Used to define the reporting capabilities	VEN to VTN
oadrRegisteredReport	Used to select the desired reports	VTN to VEN
oadrCreatedReport	Used to acknowledge the creation of reports	VEN to VTN
oadrUpdateReport	Used to send reports	VEN to VTN
oadrUpdatedReport	Used to acknowledge the reception of reports	VTN to VEN
oadrDistributeEvent	Used to send events	VTN to VEN
oadrCreatedEvent	Used to respond to the events	VEN to VTN
oadrPoll	Used to poll for new messages	VEN to VTN
oadrResponse	Used when no new messages must be sent or to acknowledge the creation of a report	VTN to VEN

Table 35: OpenADR message list

VEN registration:

To start the registration, optionally, the VEN can send an *oadrQueryRegistration* to get information from the VTN related to what protocols it supports and other basic information. The VTN sends the requested information through an *oadrCreatedPartyRegistration* payload.

Then, the VEN will request a registration through *oadrCreatePartyRegistration* which includes the VEN name and optionally the certificates for secure message transmission. If the VTN accepts this information, it will create a venID and a registrationID and send it to the VEN using the *oadrCreatedPartyRegistration* payload.

Report registration and sending:

Afterwards, the VEN report capabilities must be defined by the client through the *oadrRegisterReport* payload. In this message, the VEN defines what type of measurements it can send and at which sampling rate. The VTN will then choose the reports it wants and send the information through *oadrRegisteredReport*. The VEN will acknowledge this with *oadrCreatedReport*, to which the VTN acknowledges through *oadrResponse*. The VEN periodically will send the required reports through the *oadrUpdateReport* payload to which the VTN responds with *oadrUpdatedReport*.

Event sending:

After the VEN has been created, it will start polling for new messages from the VTN through the oadrPoll (when in PULL mode) which only includes the VEN ID. The VTN will reply with an empty message if there is no new information to be transmitted (*oadrReponse*). In case of an event it will transmit the *oadrDistributeEvent* with the event information. The VEN will then decide if they want to participate in the event or not by sending an optIn or *optOut* message in *oadrCreatedEvent*.

Figure 57 shows the interaction explained in this section.

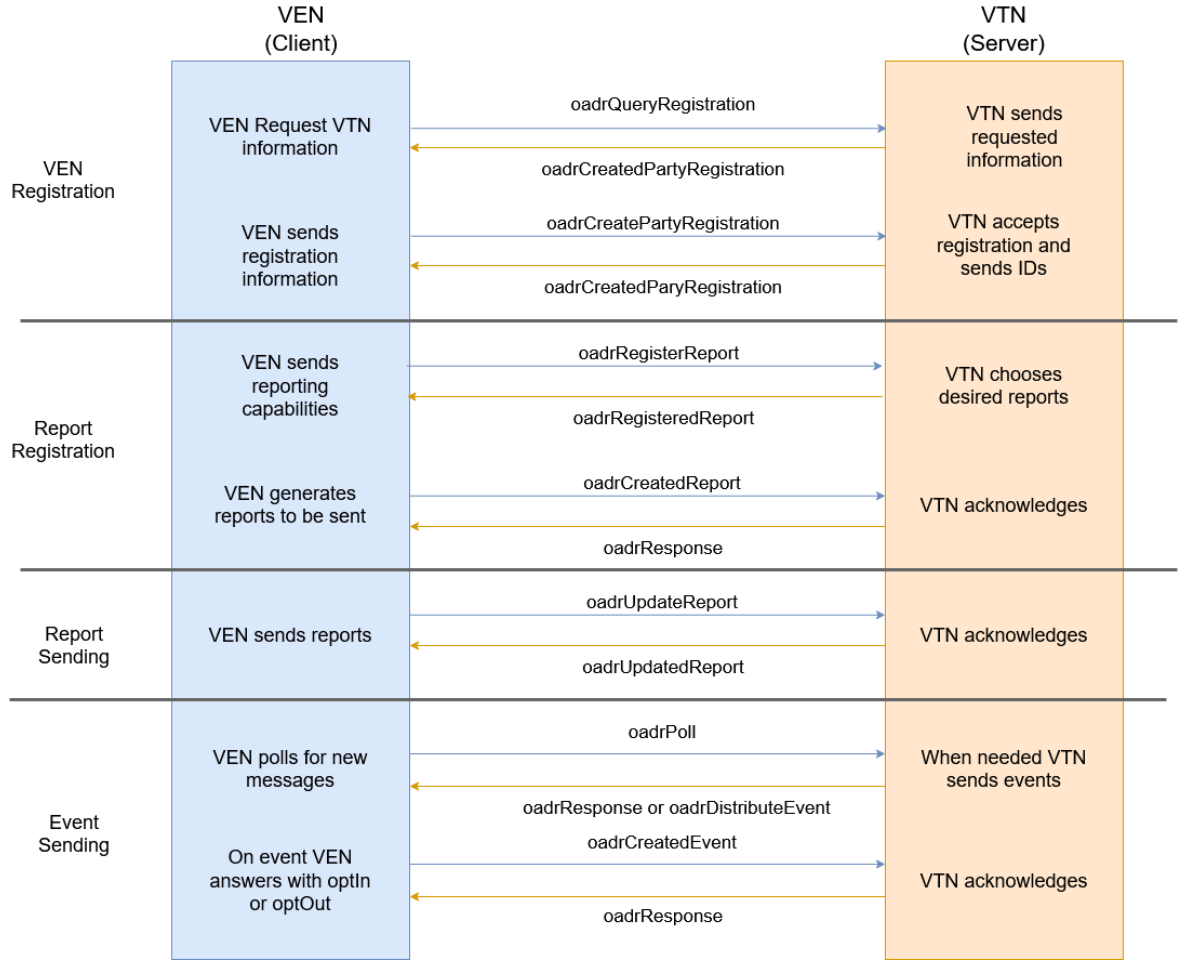

Figure 57. OpenADR VTN-VEN interaction

5.2 Events in OpenADR

One of the key features of the OpenADR protocol is that it allows to send different types of events and therefore implement different types of DR Programs. The most important components of the events are the following:

- Event descriptor: contains basic information about the event. The elements that can be defined inside the event descriptor are:
    - Event ID: unique identification of the event
    - Modification number: used when the same event has a modification
    - Modification reason: reason for modification of the event
    - Market context: identifies a DR program to which the event belongs
    - Event Status: defines how distant an event is. It can hold the values “far”, “near”, “active” or “cancelled” (openleadr also include “completed” and “none”)
    - Created date time: date of creation of the event
    - Modification date time: date of modification of the event
    - Priority: allows to define the priority of an event through an integer
    - Test event: a Boolean that defines whether the event is a test or not
    - VTN comment: the VTN can send a comment to the VEN through this element.
- Event signals: the event signal contains the most important information that defines an event. In one event one or more signals can be sent: a SIMPLE signal is mandatory and additional ones are optional. The SIMPLE signal is the most basic one that can be sent. It represents a level (ranging from 0 to 3) that can be related to different variables. For example, it can represent a level of load shed by the client, being 0 no change at all and 3 the highest possible change.
    - Signal name: the OpenADR implementation guide defines standard signal names for interoperability purposes (some are presented in Table 36).
    - Signal type: for each signal name, standard signal types are defined (some are presented in Table 36).
    - Item-base: unit of the signal (in openleadr this is defined through the measurement object)
    - Intervals: an event can contain one or more intervals. Each interval is defined by a start time, a duration and a value of the signal.
- Targets: this element defines the target to which the event is directed. Targets can be a VEN or a group of elements like different resource IDs that belong to a VEN.
- Active period: defines the active time of an event. An event can be defined by a notification time, a ramp up period, a randomization to the start of the event, an event duration and a recovery period. Figure 58 shows the different timings of an event.

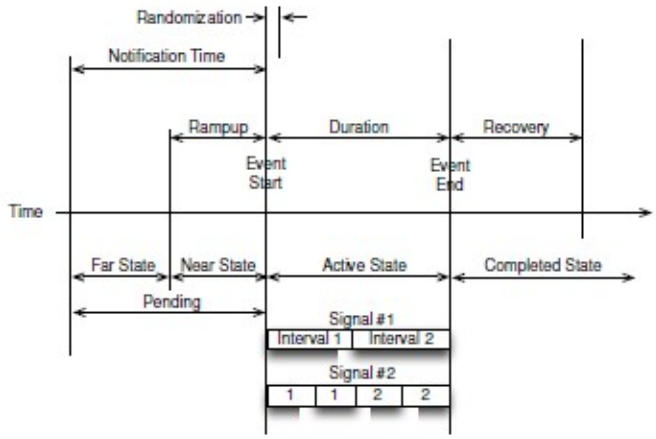

Figure 58: Times of an event in OpenADR [36]

- Response required: an element that defines whether the VEN should respond or not to an event. It must be either “always” or “never”.

Out of all the possible signals defined in the OpenADR guide, different types have been chosen to explore both explicit and implicit programs, which are shown in Table 36.

SIGNAL CATEGORY	SIGNAL NAME	SIGNAL TYPE	SIGNAL UNIT	ALLOWED VALUES	DESCRIPTION
SIMPLE LEVELS	SIMPLE	level	-	0,1,2,3	Simple levels
PRICE OF ELECTRICITY	ELECTRICITY_ PRICE	price	Currency /kWh	any	Price of electricity
		priceRelative	Currency /kWh	any	Delta of price over regular tariff
		priceMultiplier	-	any	Multiplier to the current cost of electricity
DIRECT LOAD CONTROL	LOAD_CONTROL	x-loadControlLevel-Offset	-	Integer value, Positive or negative	Discrete integer levels that are relative to normal operations, where 0 is normal operations.
		X-loadControl Percent-Offset	None	Percentage (+ or -)	Percentage change from normal operations

Table 36. OpenADR standard event signals

5.3 Implementation using openleadr

Openleadr is a compliant Python implementation of OpenADR. It allows to define VTNs and VENs in a more friendly way by offering the following services:

- Automatic VEN registration: just by providing the necessary information.
- Automatic reporting: allows to easily define which reports are going to be send and once this is defined, openleadr periodically sends the required reports.
- Event-driven: by running clients and servers in a loop all the routine interactions happen automatically (like polling for new events or sending reports).
- Dictionary style messages: openleadr translates the harder to read XML messages into Python dictionaries.
- Security features: when providing VEN certificates and VTN fingerprints, openleadr automatically sings all messages sent by the VEN and checks if the received messages come from the known VTN.

For more information on the openleadr module, refer to the official website [51].

5.3.1 OpenADR Client

In order to create a VEN or client using openleadr, the following main steps are necessary:

1 Create an OpenADRClient with basic information about the VEN and the VTN to be connected to. This must include the VEN name and the VTN URL. Optionally, certificates from the VEN can be provided to sign each XML sent to the VTN for security purposes. Also, the VTN fingerprint can be defined in order to check that the incoming messages are coming from the defined VTN.

2 Specify which reports the VEN can generate and create the callback functions that can generate the report values. In the report information, the resource ID, measurement type and sampling rates must be provided.

3 Give the OpenADRClient a callback function that will be run when an event is received to decide if the client opts in or out of the event.

After defining the previous requirements, the client will run on a loop and will be polling for new messages to the VTN periodically.

5.3.2 OpenADR Server

Similarly, to implement a server or VTN using openleadr, there are a few steps that must be completed:

1 Create an OpenADRServer with basic information about the VTN, like the VTN name.

2 Add a handler for the party registration. This way, every time a VEN requests a registration, the VTN will decide if it is accepted or not. If yes, it will create a venID and a registrationID.

3 Add a handler to deal with the reporting capabilities of the client. After receiving the possible reports, the server must choose which one it wants and send them to the VEN. Once the VEN starts sending reports, a handler will be used to store the datapoints into a database.

4 Define and send the events. After analysing the datapoints received from the VEN, the VTN might want to send an event. The event sent must contain all the necessary information like signal type, signal name, event interval and targets. A callback function is also defined to be notified if the VEN has accepted to participate in the event or not.

5.4 OpenADR communication tests

As a previous step to a the implementation of OpenADR, some communication tests have been carried out in order to test the protocol. A client (“test_VEN”) with two assets, a HVAC and a controllable load, and a server (“test_VTN”), running at localhost, have been created with the goal of mimicking a real-life scenario. Figure 59 is a representation of the test defined.

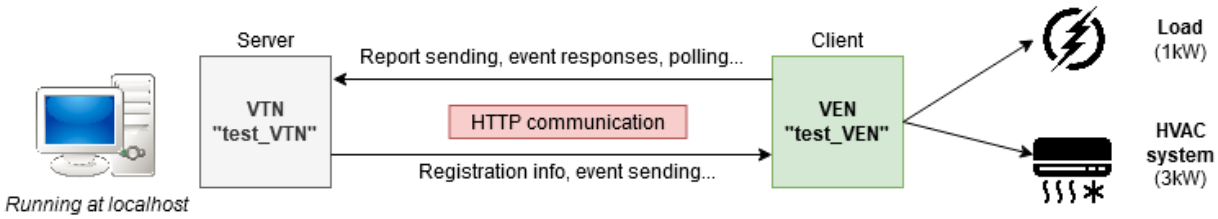

Figure 59. OpenADR communication test

The client and server will exchange DR communication messages according to the OpenADR specification. The main goals of the test implementation are the following:

1 Define an OpenADR compliant client and server

2 Understand the VEN registration process

3 Understand the sending of reporting capabilities by the VEN and the selection of desired reports by VTN

4 Understand report sending by the VEN

5 Define and explore different event types that the VTN can send to the VEN

6 Define the response of the VEN when events are received.

In order to fulfil the goal number 5, different event types have been defined, each explained in one section of the current chapter. The event types are the following:

- Simple events: the implementation of this test was the first to be carried out due to its

simplicity. The main goals explained above were achieved through this first implementation. Once this was defined, more complex events were tested (CPP and DLC).

- Critical Peak Pricing (CPP) event: this test was developed to explore events that contain pricing information and as an example of an implicit DR program
- Direct Load Control (DLC) event: this test was developed to explore events that contain load control information and as an example of an explicit DR program.

The test implementation has the characteristics shown in Table 37:

Client VEN	Name: "test_VEN" Assets: a HVAC system and a controllable load with a maximum power of 3kW and 1kW respectively
Server VTN	Name: "test_VTN" Running at localhost
OpenADR implementation information	OpenADR 2.0 B profile VTN and VEN VTN and VEN communicate in PULL mode The VEN polls for new messages from the VTN every 10 seconds Transport mechanism: simple HTTP Certificates: not used
Registration process	The VTN only allows "test_VEN" to register
Reporting	The VEN is able to send the power data of each asset every 30 seconds The power point of each asset is calculated as a random value between 0 and the maximum power of each asset The VTN accepts all possible reports coming from the VEN and stores the power points in a .txt file
Events	The VTN when receiving a report from the VEN randomly decides to create an event or not. The created event will be sent when the VEN polls for new messages The characteristics of the events depend on the type (simple, CPP or DLC) and are explained in each of the following chapters
Event targets	Simple events and the DLC events: targets are defined randomly from the list of assets of the VEN (either the HVAC system or the load) CPP: as suggested by the OpenADR implementation guide, only the VEN ID is specified as the target
Event duration	The event start time is randomly defined with a delay of 30 to 90 seconds from the event creation time. The event duration is also a random integer between 30 to 90 seconds. These timings were chosen to develop test in a fast way

Table 37. OpenADR test characteristics

The definition of each test is structured in the following way:

1 The real-life interest of the test is presented (potential VENs and VTNs)
2 Specific features of the test are presented.
3 Main characteristics of the communication test are shown.
4 Expected results and obtained results are presented and compared.
5 Screenshots of the tests are shown.

For further detail, the full Python logging of each test can be found in Annex H along with an example of each XML message sent between VTN and VEN. The full Python scripts for the test implementation can also be seen in Annex H.

5.4.1 Simple events

The simple signal is the most basic implementation of OpenADR. It allows to send events where values from 1 to 3 represent the importance of an event. It is usually sent along with other signal types except for OpenADR 2.0A implementations, where the simple signal is the only one supported. There is a wide variety of potential DR Programs that could use these events. However, VTNs and VENs must predefine what the simple levels represent.

Besides the characteristics shown in Table 37, this particular implementation has the features shown in Table 38:

Event Signal	Signal type: level Signal name: simple Payload: random value between 1 and 3 for each event created
Active Period	Only defined by start time and duration of the event (as per Table 37)
VEN opt response	The VEN can either optIn or optOut of events (it will optOut to the second event received)

Table 38. Simple event features

Table 39 shows the time of start and end of the test, along with the number of events created and opt responses of the VEN.

Start of the test	17:05:20
End of the test	17:07:50
Events created	2
Opt responses of VEN	“OptIn” to the first, “OptOut” to the second
Completed events at end of test	1
Active events at end of test	0
Pending events at end of test	0

Table 39. Simple event test results

Figure 60 shows the timeline of the test developed and the VTN and VEN screenshots (Figure 61 and Figure 62 respectively).

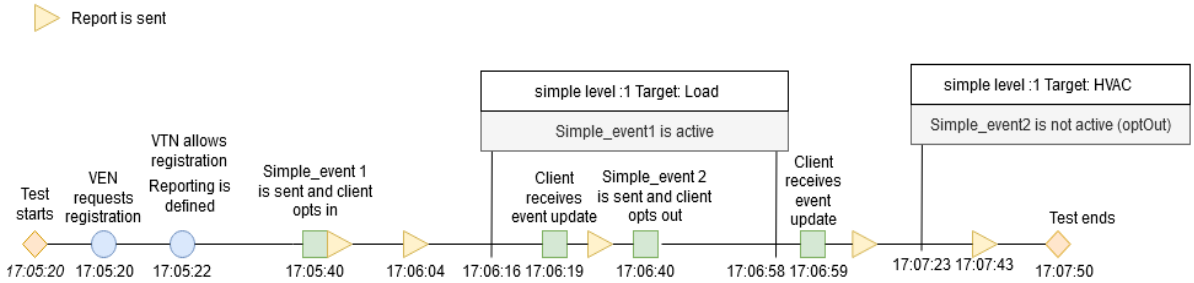

Figure 60: Simple event timeline

```
**********************************************************************************
                    Your VTN Server is now running at
                 http://127.0.0.1:8080/OpenADR2/Simple/2.0b
**********************************************************************************

[17:05:22]Client test_VEN has registered succesfully as ven1
[17:05:22]Report for HVAC containing power in W will be received every 0:00:30
[17:05:22]Report for Load containing power in W will be received every 0:00:30
[17:05:40]Adding Simple event for ven1
---------------------------------------------------------------------------
Event ID: Simple_event1
Event target is [Target('resource_id=Load')]
Event only contains a SIMPLE Signal with the following information.
The event has 1 interval(s)
 -------- Interval number 1 ---------
Event starts at: 2021-01-30 17:06:16.764849+00:00
Event duration is: 0:00:42
Event level is: 1
---------------------------------------------------------------------------
[17:05:40]Client responded optIn to event Simple_event1
[17:06:33]Adding Simple event for ven1
---------------------------------------------------------------------------
Event ID: Simple_event2
Event target is [Target('resource_id=HVAC')]
Event only contains a SIMPLE Signal with the following information.
The event has 1 interval(s)
 -------- Interval number 1 ---------
Event starts at: 2021-01-30 17:07:23.568423+00:00
Event duration is: 0:00:37
Event level is: 1
---------------------------------------------------------------------------
[17:06:40]Client responded optOut to event Simple_event2
```

Figure 61. VTN Simple event output

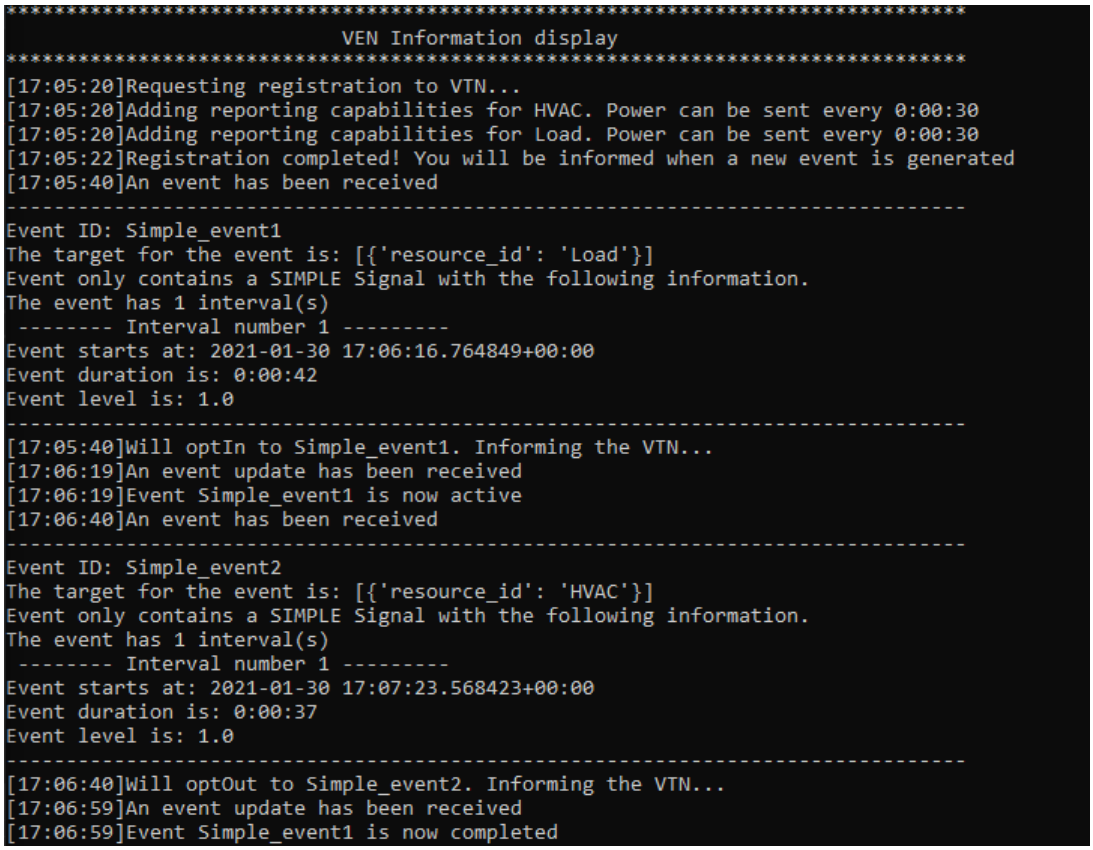

Figure 62. VEN Simple event output

The main goals of this test were fulfilled. Table 40 shows the expected and obtained results:

Expected results	***Obtained results***
VEN requests registration and is allowed by VTN	✓ VEN sends oadrCreatePartyRegistration with registration info and VTN allows it when sending oadrCreatedPartyRegistration
VTN receives reporting capabilities from VEN and chooses desired reports	✓ VEN sends capabilities in oadrRegisterReport and VTN chooses reports through oadrRegisteredReport
VEN periodically sends reports as requested by VTN	✓ This is done through oadrUpdateReport
VTN receives these reports	✓ This is done through oadrUpdatedReport
VEN polls for new messages every 10 seconds	✓ This is done through oadrPoll
VTN creates events and VEN collects them when polling for messages	✓ VEN collects the events created by the VTN through oadrDistributeEvent
The VEN is able to interpret the	✓ The VEN responds to events with oadrCreatedEvent

events and respond to them	
VEN gets notified when the event starts and ends	✓ The VEN receives event updates through oadrDistributeEvent. When event starts the event_status goes from “far” to “active”. When the event ends it goes from “active” to “completed”
VEN can opt out to an event	✓ The VEN opted out to the second event
VTN does not update an event to Active when the VEN has opted out	✓ The VEN did not receive event updates related to the second event, to which it opted out

Table 40. Expected and obtained results of the simple test

5.4.2 CPP events

A CPP program is a type of implicit DR that allows to define different pricing when it is anticipated that high wholesale market prices or power system emergency conditions will arise, The CPP tariff is set on top of the existing one, which can be flat or TOU based. Potential VTNs for this program are energy utilities and VENs are residential, commercial and industrial consumers.

Besides the characteristics shown in Table 37, this particular implementation has the following features:

Event Signals	The CPP events defined as part of this test contain two signals: • An ELECTRICITY_PRICE signal of type priceMultiplier (see Table 36) with a payload of 6 or 10. This means that when an event applies, the customers usual electricity tariff is multiplied by that amount for the event period. • A Simple signal containing the importance of the event. In this case, for the priceMultiplier 6, the simple signal payload has a value of 1 and for the priceMultiplier 10, a value of 2.
Active Period	The event contains two intervals, with the same duration each and no time interval between them
VEN opt response	VEN always opts in to events

Table 41. CPP event features

The following table shows the time of start and end of the test, along with the number of events created and opt responses of the VEN.

Start of the test	17:13:26
End of the test	17:16:00
Events created	1
Opt responses of VEN	“OptIn”
Completed events at end of test	1
Active events at end of test	0
Pending events at end of test	0

Table 42. CPP test results

Figure 63 shows the timeline of the test developed.

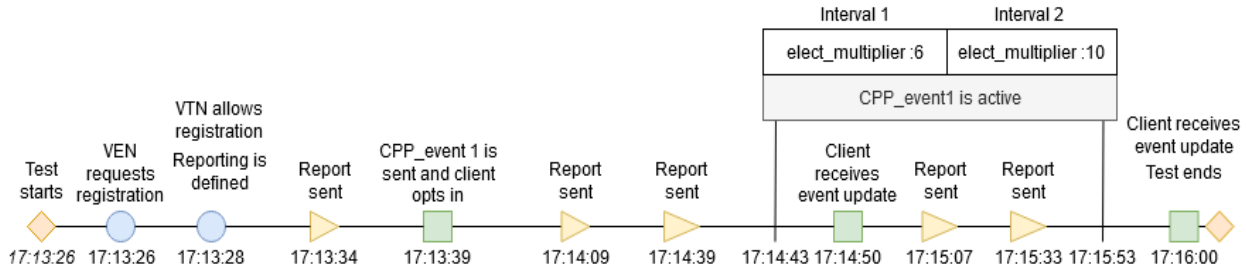

Figure 63. CPP event timeline

Figure 64 and Figure 65 show the output obtained in the VTN and VEN screens, respectively.

```
**********************************************************************************
                    Your VTN Server is now running at
                http://127.0.0.1:8080/OpenADR2/Simple/2.0b
**********************************************************************************

[17:13:28]Client test_VEN has registered succesfully as ven1
[17:13:28]Report for HVAC containing power in W will be received every 0:00:30
[17:13:28]Report for Load containing power in W will be received every 0:00:30
[17:13:34]Adding CPP event for ven1
----------------------------------------------------------------------------------
Event ID: CPP_event1
Event target is [Target('ven_id=ven1')]
Event received contains two signals.
The event has 2 interval(s)
 -------- Interval number 1 ---------
Interval starts at: 2021-01-06 17:14:43.349088+00:00
Interval duration is: 0:00:35.500000
The electricity price for this interval is:
6 times the usual tariff
Event priority is: 1
 -------- Interval number 2 ---------
Interval starts at: 2021-01-06 17:15:18.849088+00:00
Interval duration is: 0:00:35.500000
The electricity price for this interval is:
10 times the usual tariff
Event priority is: 2
----------------------------------------------------------------------------------
[17:13:39]Client responded optIn to event CPP_event1
```

Figure 64: VTN CPP test output

```
**********************************************************************************
                          VEN Information display
**********************************************************************************
[17:13:26]Requesting registration to VTN...
[17:13:26]Adding reporting capabilities for HVAC. Power can be sent every 0:00:30
[17:13:26]Adding reporting capabilities for Load. Power can be sent every 0:00:30
[17:13:28]Registration completed! You will be informed when a new event is generated
[17:13:39]An event has been received
----------------------------------------------------------------------------------
Event ID: CPP_event1
The target for the event is: [{'ven_id': 'ven1'}]
Event received contains two signals.
The event has 2 interval(s)
 -------- Interval number 1 ---------
Interval starts at: 2021-01-06 17:14:43.349088+00:00
Interval duration is: 0:00:35
The electricity price for this interval is:
6.0 times the usual tariff
Interval priority is: 1.0
 -------- Interval number 2 ---------
Interval starts at: 2021-01-06 17:15:18.849088+00:00
Interval duration is: 0:00:35
The electricity price for this interval is:
10.0 times the usual tariff
Interval priority is: 2.0
----------------------------------------------------------------------------------
[17:13:39]Will optIn to CPP_event1. Informing the VTN...
[17:14:50]An event update has been received
[17:14:50]Event CPP_event1 is now active
[17:16:00]An event update has been received
[17:16:00]Event CPP_event1 is now completed
```

Figure 65: VEN CPP test outputs

The main goals of this test were fulfilled. Besides the points already checked through the simple test, the remaining expected results and the obtained ones for this test are shown in Table 43:

Expected results	***Obtained results***
The VTN can send pricing information	✓ Through the ELECTRICITY_PRICE signals sent
Events defined contain more than one signal	✓ Simple and ELECTRICITY_PRICE signals are sent
Simple signal reflects pricing impact	✓ Simple signal has a higher value when the price multiplier is higher
Events contain more than one interval	✓ Two intervals are defined for each event
VEN gets notified when one interval finishes	✘ The VEN only gets notified when the event starts and ends but not when the second interval starts (compliant with OpenADR)
The event does not specify the assets	✓ The event target only includes VEN ID

Table 43. Expected and obtained results of the CPP test

5.4.3 DLC events

DLC Programs are a type of explicit DR that allow the VTNs to directly control the VENs load, for example a HVAC system. Potential VENs for this type of programs are residential or small commercial ones that have HVAC systems and are willing to get an incentive for participating in DR Programs. DLC is also sometimes used for industrial customers. Potential VTNs are energy utilities or demand aggregators.

Besides the characteristics shown in Table 37, this particular implementation has the features shown in Table 44:

Event Signals	A DLC event contain two signals • A LOAD_CONTROL signal of type x-loadControlLevelOffset (see Table 36) with a payload between -10 and 10. This value represents the level offset from the normal operation. Therefore, negative values indicate that the load will be working at a lower setpoint that usual. The higher the offset the higher the change is from normal operation. For the test implementation, this value is randomly defined. • A Simple signal containing the importance of the event. In this case, it is calculated depending on the LOAD_CONTROL signal. Higher offsets indicate higher importance of the event.
Active Period	In addition to the start time and duration, the event active period is also defined by a ramp-up period of 15 seconds for the HVAC system. This way, the VEN receives an event update when the event is “near” (ramp up period starts).
VEN opt response	VEN always opts in to events, because of the type of DR program

Table 44. DLC event features

Table 45 shows the time of start and end of the test, along with the number of events created and opt responses of the VEN.

Start of the test	20:10:18
End of the test	20:11:59
Events created	2
Opt responses of VEN	“OptIn” to both
Completed events at end of test	1
Active events at end of test	1
Pending events at end of test	0

Table 45. DLC event test results

Figure 66 shows the timeline of the test developed and Figure 67 and Figure 68 of the VEN and VTN screens respectively.

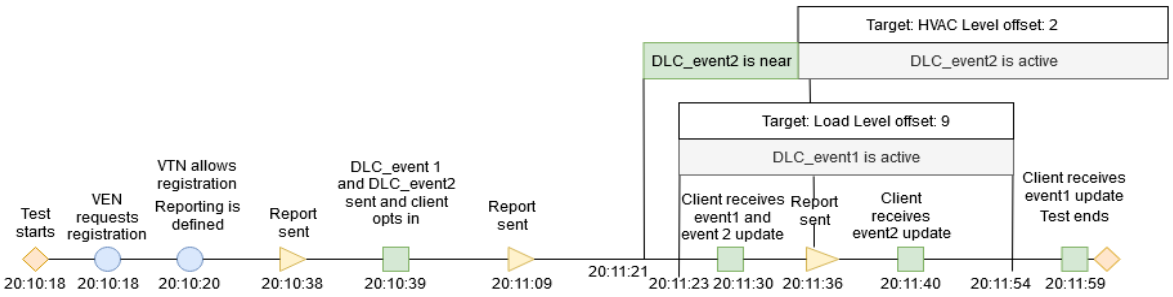

Figure 66: DLC event timeline

```
*****************************************************************************************
                            VEN Information display
*****************************************************************************************
[20:10:18]Requesting registration to VTN...
[20:10:18]Adding reporting capabilities for HVAC. Power can be sent every 0:00:30
[20:10:18]Adding reporting capabilities for Load. Power can be sent every 0:00:30
[20:10:20]Registration completed! You will be informed when a new event is generated
[20:10:39]An event has been received
-----------------------------------------------------------------------------------------
Event ID: DLC_event1
The target for the event is: [{'resource_id': 'Load'}]
Event received contains 2 signals.
The event has 1 interval(s)
 -------- Interval number 1 ---------
Interval starts at: 2021-01-06 20:11:23.414215+00:00
Interval duration is: 0:00:31
The VTN will change the setpoint of the load...
The load will work over normal operation. Load control level: 9.0
Interval priority is: 3.0
-----------------------------------------------------------------------------------------
[20:10:39]Will optIn to DLC_event1. Informing the VTN...
[20:10:39]An event has been received
-----------------------------------------------------------------------------------------
Event ID: DLC_event2
The target for the event is: [{'resource_id': 'HVAC'}]
Event received contains 2 signals.
The event has 1 interval(s)
 -------- Interval number 1 ---------
Interval starts at: 2021-01-06 20:11:36.416238+00:00
Interval duration is: 0:00:43
The VTN will change the setpoint of the load...
The load will work over normal operation. Load control level: 2.0
Interval priority is: 1.0
-----------------------------------------------------------------------------------------
[20:10:39]Will optIn to DLC_event2. Informing the VTN...
[20:11:30]An event update has been received
[20:11:30]Event DLC_event2 is now near
[20:11:30]An event update has been received
[20:11:30]Event DLC_event1 is now active
[20:11:40]An event update has been received
[20:11:40]Event DLC_event2 is now active
[20:11:59]An event update has been received
[20:11:59]Event DLC_event1 is now completed
```

Figure 67: VEN DLC event output

```
********************************************************************************
                    Your VTN Server is now running at
                 http://127.0.0.1:8080/OpenADR2/Simple/2.0b
********************************************************************************

[20:10:20]Client test_VEN has registered succesfully as ven1
[20:10:20]Report for HVAC containing power in W will be received every 0:00:30
[20:10:20]Report for Load containing power in W will be received every 0:00:30
[20:10:38]Adding DLC event for ven1
--------------------------------------------------------------------------------
Event ID: DLC_event1
Event target is [Target('resource_id=Load')]
Event received contains 2 signals.
The event has 1 interval(s)
 -------- Interval number 1 ---------
Interval starts at: 2021-01-06 20:11:23.414215+00:00
Interval duration is: 0:00:31
The VTN will change the setpoint of the load...
The load will work over normal operation. Load control level: 9
Event priority is: 3
--------------------------------------------------------------------------------
[20:10:38]Adding DLC event for ven1
--------------------------------------------------------------------------------
Event ID: DLC_event2
Event target is [Target('resource_id=HVAC')]
Event received contains 2 signals.
The event has 1 interval(s)
 -------- Interval number 1 ---------
Interval starts at: 2021-01-06 20:11:36.416238+00:00
Interval duration is: 0:00:43
The VTN will change the setpoint of the load...
The load will work over normal operation. Load control level: 2
Event priority is: 1
--------------------------------------------------------------------------------
[20:10:39]Client responded optIn to event DLC_event1
[20:10:39]Client responded optIn to event DLC_event2
```

Figure 68: VTN DLC event output

The main goals of this test were fulfilled. Besides the points already checked through the simple test, the remaining expected results and the obtained ones for this test are shown in Table 46:

Expected results	***Obtained results***
The VTN can send load control signals to VEN assets	✓ Through the LOAD_CONTROL signals sent
Events defined contain more than one signal	✓ Simple and LOAD_CONTROL signals are sent
Simple signal reflects event impact	✓ Simple signal has a higher value when the control level offset is higher
Both demand increase and demand decrease can be sent	✓ Negative values of the load control signal are demand reduction and higher values are demand increases
The VEN can define a ramp-up period for their assets	✓ A ramp-up period has been defined for the HVAC system

The VEN gets notified when ramp-up period starts and ends	✓ When ramp-up period starts the VEN receives an update status through oadrDistributeEvent, where event_status goes from “far” to “near”. When ramp-up period ends oadrDistributeEvent is received, where event_status goes from “near” to “active”

Table 46. Expected and obtained results of the DLC test

5.5 OpenADR implementation in the laboratory

In order to analyse the protocol in real conditions, an OpenADR VEN has been defined in the laboratory SCADA. As a first approach, one of the laboratory emulation cabinets has been selected (LC4) and used to test the OpenADR communication.

As Figure 69 shows, the laboratory SCADA acts as a VEN communicating with the aggregation platform’s VTN. The information received from the VTN can then be translated into commands for the laboratory elements.

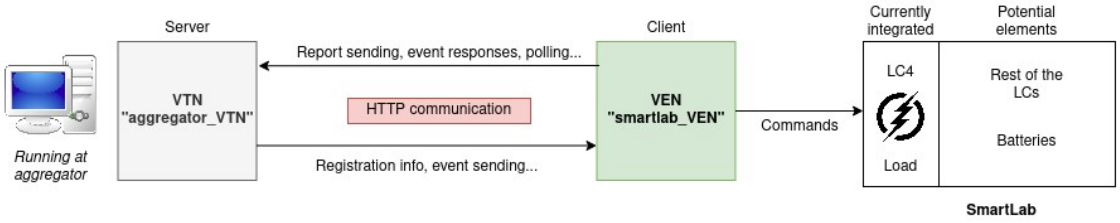

Figure 69: Laboratory OpenADR implementation

5.5.1 SmartLab VEN definition

Similarly to the test implementation in the previous section, the laboratory VEN has been defined using the openleadr module in Python (the script is included in Annex I). The characteristics of the VEN in this case are shown in Table 47.

Client VEN	Name: “smartlab_VEN” Assets: LC4 (RB12) that acts as a controllable load that can curtail its consumption when defined by the aggregator
Server VTN	Name: “aggregator_VTN” Running at the aggregator platform
OpenADR implementation information	OpenADR 2.0 B profile VEN VEN communicates in PULL mode The VEN polls for new messages from the VTN every 10 seconds Transport mechanism: simple HTTP Certificates: not used
Reporting	The VEN is able to send the power data of the asset every minute. The current load power value is read from a recipe (LOAD.DAT)
Events	When the VEN receives a new event or an event update it stores the relevant information in a recipe (ACTIVATIONS_OPENADR.DAT) This recipe is then read from the SCADA to define the LC setpoint during events that are active Only events of one interval and x-loadControlPercentOffset have been considered for this implementation. Therefore, the signal payload received represents the percentage over the current setpoint at which the load must work.

Table 47: OpenADR laboratory characteristics

5.5.2 OpenADR test in the laboratory

To test the communication using OpenADR, the scenario shown in Table 48 has been defined. A screenshot of the SCADA Demo can be found in Annex G.

Test start	18:00
Test duration	2 hours
Emulation cabinet used	LC 4 - RB12
Emulation type	20 - Load

Table 48: OpenADR lab test characteristics

For this scenario, a two hour profile was chosen for the load and one event was defined. During 30 minutes, the load is demanded to work at 50% of its current setpoint. The event details are shown in Table 49:

Event start	18:30
Event duration	30 minutes
Event notification	15 minutes before the event
Event signal number	2

Event signal names	SIMPLE and LOAD_CONTROL
Event signal types	simple and x-loadControlPercentOffset
Event signal payloads	1 for simple signal and 0.5 for x- loadControlPercentOffset

Table 49: OpenADR lab test event description

In order to understand the advantages of OpenADR, the same scenario will be executed both using OpenADR and using the current communication with the aggregator which currently is through InfluxDB database. For the Aggregator API, the same event setpoint is written into the database under the field Activations. The queries for new commands are done every minute instead of 15 minutes, to have the same frequency of query as the OpenADR communication.

After running the scenario, we can see how the power consumed by the load was affected by the events received from the aggregator. In Figure 70 we can see three values: the base consumption of the load (in red in both graphs), the consumption when communicating with the aggregator using OpenADR (top, in green) and the consumption when communicating with the aggregator using their API (bottom, in blue).

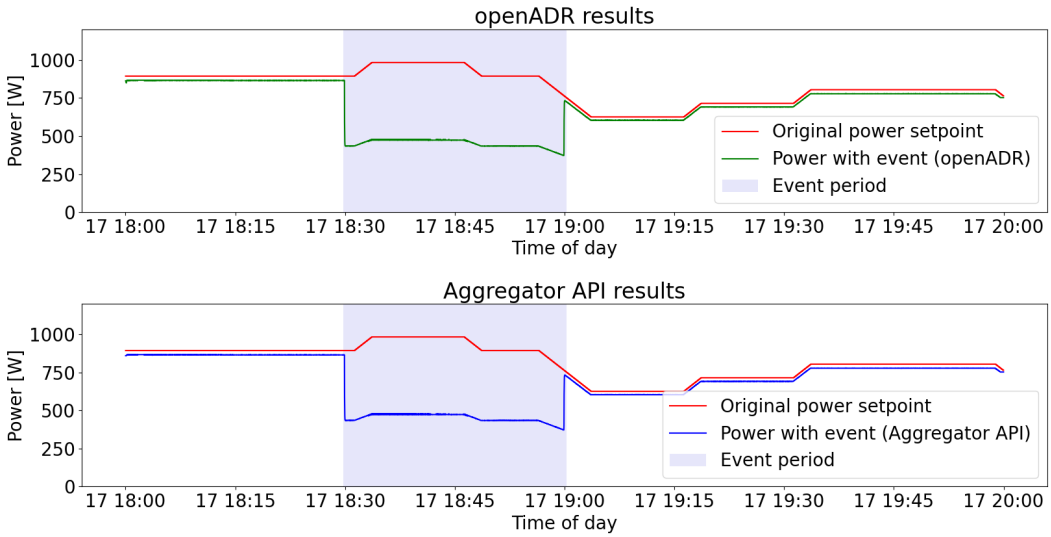

Figure 70: OpenADR Lab test results

We can see that the event was received in both cases successfully. Indeed, the results obtained for the load power are the same, as the curves on the previous figure follow the same behaviour. The main logging results on the OpenADR VEN side are shown below. The VEN polls for new messages every 10 seconds and sends reports every minute. As an example one of the report messages has been included:

17:58:19 : VEN is sending **oadrCreatePartyRegistration** message
17:58:19 : VEN is now registered with ID smartlab
17:58:19 : The polling frequency is 0:00:10
17:58:19: VEN is sending **oadrRegisterReport** message
17:58:19: VEN is sending **oadrCreatedReport** message

Registration

18:00:00: Sending **oadrPoll** message 18:00:00: A new report for Load will be sent to VTN. Current power: 435.0 W 18:00:00: VEN is sending **oadrUpdateReport** message	Report sending
18:15:20: Sending **oadrPoll** message 18:15:20: The VEN received an event 18:15:20: Sending **oadrCreatedEvent** message (will optIn to event)	Event notification received
18:30:00: Sending **oadrPoll** message 18:30:00: An event update has been received. Event1 is now active 18:30:00: Responding to event with **oadrCreatedEvent**	Event is active
18:30:00: Sending **oadrPoll** message 18:30:00: An event update has been received. Event1 is now completed 18:30:00: Responding to event with **oadrCreatedEvent**	Event is completed

The logger from the SCADA Demo screen is shown in Figure 71.

Figure 71: Event logger for the OpenADR lab test

We can see how the event was received by the VEN 15 minutes before the start time. During this time the event is in the state “far”. For the duration of the event this state is set to “active” and once it ends to “completed”.

5.5.3 Aggregator API vs OpenADR comparison

As we have seen, both the aggregator API and OpenADR can be used to communicate events. The main advantage of the current API vs using OpenADR is the simplicity. A higher number of clients can be reached because they do not need to be OpenADR compliant VEN clients.

However, for the cases where the clients are VENs, using OpenADR provides several advantages over the current API. Mainly, the fact that OpenADR is a standard solution that can be used to communicate with a variety of assets.

- The possibility of sending events before they start being active. This way, the customer can know in advance how the loads are going to be modified.
- The client also knows in advance when the event is going to end, unlike in the current communication with the aggregator.
- Even if it was not used for the lab test, with OpenADR the option of adding a ramp-up period is also an additional feature.

- Using the opt responses, the customers can accept or deny the events received and inform the server.
- Events can contain detailed and complex information, like several signals or time intervals with different payloads.
- The report selection happens in the beginning of the communication, where the client sends the possible report and the server decides which ones will be sent periodically. This step was done manually before starting any test with the aggregator.

Even if some of these features could be implemented using the aggregator API, the advantage of OpenADR is that they are already included in the protocol. To include the features, the aggregator would have to define them and provide a detailed explanation to the clients. Using OpenADR, starting the communication with a new client would be fast and easy. However, if the aggregator would like to implement OpenADR certain steps like the microgrid definition would still need to be sent as a json before the communication can happen.

6 Time planning

The development of the thesis took place between October 2020 and June 2021, in a part time schedule (around 5 hours/day). In order to meet the objectives of the thesis the project was divided into different tasks. A short explanation of each one is presented in the Table 50, along with its duration in days. The detailed time planning of the project and each subtask is presented as a Gantt Diagram in Annex J.

Title	Description	Days
1 Literature review and other learning	Literature includes concepts of Demand Response, flexibility and OpenADR. Other learning includes Python programming, ITME and SCADA functioning and InfluxDB API understanding.	58
2 Laboratory preparation work	Including scenario and profile definition, InfluxDB scripts, HVAC definition, battery control definition and SCADA adaptation and Demo screen creation	198
3 OpenADR Integration	Including openleadr understanding and VTN and VEN script creation	47
4 Experimental Work	Residential, prosumer and OpenADR scenario development	35
5 Memory	Preparation of the memory and annexes	161
	Total Project	252

Table 50: Task description and duration

This thesis is the final degree project of the Double Masters Degree in Energy Engineering and Industrial Engineering, corresponding to 42 ECTS credits. Considering that each credit corresponds to 25 hours of work, we can see in Table 51 that the dedication time was fulfilled.

ECTS credits in hours	42 * 25 = **1050 hours**
Project Dedication	252 days * 5 hours/day = **1260 hours**

Table 51: Project dedication according to ECTS credits

7 Environmental and Economic Analysis

This section describes the environmental and economic impact of the execution of the project. The elements that exist exclusively as a consequence of this thesis have been considered.

7.1 Environmental Analysis

In terms of the laboratory, the electricity consumed during the tests is included (Table 52). Since the laboratory works in a closed system, consumption from the public grid is only needed to balance the losses, which are assumed to be 500W per emulation cabinet. The battery however is not connected in this closed system, so all the energy used for charging comes from the grid.

Scenario	Hours	Cabinet losses	Battery charge	Total energy
Prosumer base	72 h	2 x 500 W	12.14 kWh	84.14 kWh
Prosumer aggregator	72 h	2 x 500 W	11.80 kWh	83.80 kWh
OpenADR	4 h	1 x 500 W	0	2 kWh
			Total	**169.94 kWh**

Table 52: Real grid consumption for the laboratory scenarios

In addition, because the scenarios run at night, there was an increase in the cooling of the installations during this time. The extra cooling is assumed to be provided by two splits of 1 kW each during 3 hours every night for the month of May, when the scenarios took place.

Besides considering the increase of electricity consumption in the laboratory, to evaluate the emissions of the project, the use of the personal computer, monitor, lighting and the SCADA computer is added. An average use of 100 W for the computers, 30 W for the monitor and 10 W for the lightbulbs was assumed. Also, it was considered that the SCADA computer was only used half of the time and the lighting a 30% due to the presence of natural light.

A factor of 0.119 kg of CO2 equivalent produced per kWh consumed from the Spanish grid (October 2020 to May 2021) has been used, extracted from Red Electrica Española [52].

Concept	Quantity	Av. Power	Electricity	Emissions
Personal Computer use	1260 h	100 W	126 kWh	15.0 kg CO_2 eq
Monitor use	1260 h	30 W	37.8 kWh	4.5 kg CO_2 eq
Lighting	378 h	10 W	3.78 kWh	0.45 kg CO_2 eq
SCADA Computer use	630 h	100 W	63 kWh	7.5 kg CO_2 eq
Laboratory additional cooling	93 h	2 000 W	186 kWh	22.13 kg CO_2 eq
Scenario electricity consumption	-	-	168.94 kWh	20.10 kg CO_2 eq
			Total emissions	**50.18 kg CO_2 eq**

Table 53: Total CO2 equivalent emissions of the project

The final calcuated emissions of the project sum up to **50.18 kg CO_2 eq** as per Table 53.

7.2 Economic Analysis

The electricity used in the laboratory carries a cost associated to it. To translate the electricity consumption to economic cost, a 3.0A tariff (3 periods, >15 kW) has been considered for the laboratory (Table 54) Then, with this tariff and knowing when each scenario took place, the associated cost has been estimated. For the authors electricity use, a one-period tariff has been considered (0.1395 €/kWh). These prices are reference values from an spanish distributor [53].

	Low	Medium	High
Period	00:00-08:00h	08:00 - 11:00 15:00 - 00:00	11:00 - 15:00h
Price tariff 1 (summer)	0.0701 €/kWh	0.0916 €/kWh	0.1053 €/kWh

Table 54: Electricity tariff considered for the project

Although the thesis was not done under a paid internship, for estimating the authors labour cost, a unitary cost of 8€/h for the duration of the project has been considered.

It is reminded that only costs that exist as a consequence of this thesis have been considered. Therefore, other fixed costs like the cost of the laboratory facilities, the labour of the rest of the collaborators of the project or the Internet connection have not been included.

All the elements create a total project budget of **12 279.29 €** (tax included) as per Table 55. Without considering the labor cost, the actual economic impact of the project would be **82.49 €** (tax included).

Concept	Quantity	Unitary cost	Cost
Labour Cost	1260 h	8 €/h	10 080 €
Laboratory consumption	355.94 kWh	see tariff 1	26.00 €
Other consumptions	230.58 kWh	0.1395 €/kWh	32.17 €
Office material	-	-	10 €
		Total cost before taxes	**10148.17 €**
		21 % IVA	2131.12 €
		Total cost after taxes	**12279.29 €**

Table 55: Total cost of execution of the project

8 Conclusions

The work developed in this thesis has contributed to increase the knowledge on demand-side flexibility services, by integrating the laboratory with an aggregation platform. The main results of the work developed are:

- Implementation of a new tool in IREC's SmartLab to allow the communication with the aggregator.
- Creation of new functionalities on the SCADA system to allow the configuration of scenarios and the inclusion of different elements
- OpenADR implementation and testing in the laboratory to validate its usability.

Specifically, the laboratory SCADA has been modified to allow the communication with the aggregator using their API. Several elements of the laboratory have been selected in order to test the communication with the aggregator in different scenarios. The battery on the laboratory has been used, for which a simple control strategy has been defined. In addition, two of the laboratory emulation cabinets have been connected with the SCADA and used as a load and a PV system. Finally, a virtual HVAC has been defined for one thermal zone. With all these elements two test types have been carried out:

1. A residential customer containing a virtual HVAC and a virtual uncontrollable load. The base case for summer and winter periods has been developed. Then, by allowing the communication with the aggregator, the HVAC consumption has been modified depending on the events received. The aggregator was able to control the temperature setpoint of the HVAC depending on the grid necessities. The tests developed showed that, when the aggregator calculated the assets' flexibility correctly, the user was able to modify their consumption and shift the demand to other periods, right after the events. In the winter case, we could see that the aggregator did not calculate the flexibility correctly and therefore some of the events received did not have an impact on consumption. From the users point of view, the thermal comfort was not compromised in any of the tests as the indoor temperature was maintained in an adequate range.

2. A prosumer that had a PV system, an uncontrollable load and a battery. In this case the two emulation cabinets and the laboratory second-life EV battery were used. After running the base case, the aggregator scenario was developed where the battery setpoint could be modified. We saw that the battery shifted the charging from the grid to a latter period during the first two events and in the third one, a battery discharge was forced. As a consequence of the aggregator, the grid consumption was moved to other periods and in the last event, the electricity surplus from the battery discharge was injected to the grid.

Since the goal of the aggregator is not to decrease consumption but to solve grid constraints or to provide balancing services, for both scenarios we could see that the cost of electricity and the grid consumption were either slightly lower, equal or in some cases even higher. This highlights the importance of defining proper incentives in order to attract customers into flexibility trading.

Finally, a relevant protocol in Demand Response programs, OpenADR was implemented and tested in the laboratory. The testing phase allowed to understand the possibilities that the protocol provides in terms of signal types, number of intervals, reporting, responses to events etc. To implement the protocol in the laboratory a VEN was created for the SCADA. A simple scenario was defined using one of the emulation cabinets. The same scenario was developed using the aggregator API, in order to compare both. Although both methods allowed sending events, OpenADR brings several advantages. Mainly, it provides a standardized method to communicate and allows sending more detailed information, even before the events start.

Lessons learnt

From the development of the project, several obstacles were found, from which we can extract some lessons learnt:

- Using open-source projects that are still in development, like the openleadr module, poses some challenges. The documentation is limited, there are bugs that still need fixing and there are not many discussion forums to solve doubts.
- For the controls developed (HVAC and battery), adding a hysteresis to them is necessary to avoid sudden and constant setpoint jumps.
- Unexpected computer updates, can stop one of the running scenarios.
- The second-life battery was very sensitive to changes in temperature. During a failed test, the battery was not correctly cooled and due to the temperature increase the battery stopped responding and went to fail mode. For this reason, additional cooling was programmed for the periods when scenarios took place.
- Even if the setpoint sent to the battery was zero, the fact that it was still connected created small power absorption from the laboratory microgrid. For this reason, the battery charged over time. Before running one of the scenarios, due to this fact, the battery was overcharged and doing the connection with it failed. It was necessary to discharge the battery before using it.
- The battery did not follow setpoints below 1500 W, which brought to light the difference between ideal simulations and real scenarios.
- When programming the tasks for the scenario development it was key to make sure that each one run in the correct order and only after the previous had finished. For this, defining them as synchronous tasks was necessary.
- Doing scenarios in laboratory environments is a very sensitive process. Since there are other people working in the laboratory, it is important to notify everybody to avoid possible errors (cabinet disconnection, pressing something in the SCADA…), specially for tests that take several days.

Future lines of improvement

Future work is expected to be developed in the laboratory. Some potential lines of work extracted from this project are the following:

- Increase the complexity of the HVAC control and the thermal model developed. The current virtual HVAC included several simplifications that could be improved in order to have a more accurate model. For example, a more realistic model would consider the interaction with adjacent thermal zones.
- Include pricing information for the battery control. The control defined for the battery constitutes a simple approach that does not consider the price of electricity. By introducing this variable, more price-aware strategies can be defined. For example, during off-peak periods when the price of electricity is cheap, the control would not allow discharges of the battery and would prioritize grid consumption at these moments. Later, during peak periods of higher price, the loads could be supplied from the battery.

- Regarding OpenADR, there are several features that can be implemented to increase the complexity of the events that can be received in the laboratory. Further work may include the ability to receive and translate into setpoints different types of events. This includes events with several intervals, that target different assets or that include other signal types. In addition, secure communication can be included by adding client side certificates. Finally, currently the client participates in all events received. Making use of the optIn/optOut feature of OpenADR, the client response can be defined.

- In terms of laboratory testing, further work includes connecting to more cabinets and emulating other types of controllable assets or analysing different types of events that the aggregator may send. With this purpose, further Demos should be created in the laboratory SCADA to define new scenarios. In addition, another functionality to implement would be to automatically calculate the KPIs of each scenario in the SCADA and show them on the Demo screen.
- Another possible line of work includes using the Energy Management System (EMS) of the laboratory to receive the aggregator commands instead of directly communicating them to the devices. This option would be interesting to analyse how the EMS may react to the events received by the aggregator.

- Finally, an extended economic analysis for the developed scenarios is possible in order to determine the appropiate incentives that the aggregator would have to define in order to guarantee economic benefit to their customers.

Acknowledgement

In the first place I would like to thank Alba and Mattia for the constant help and support throughout the entire project. I would also like to thank Cristina for giving me the opportunity to join IREC to develop the thesis. Also, to the whole Energy Systems Analytics group in IREC and to Bamboo Energy for providing the resources and help needed to carry out the project.

I would also like to thank those who have been part of my routine these years of master studies. My flatmates and university friends, who have made me keep going and put a smile on my face every day.

Finally, to my family for the support and love they have given me during all my university years. And specially, to my parents for giving me the opportunity to come to study in Barcelona and always feeling close to me even in the distance.

9 Bibliography

[1] IPCC, 'Summary for policy makers', 2018, doi: 10.1016/j.oneear.2019.10.025.

[2] G. Notton *et al.*, 'Intermittent and stochastic character of renewable energy sources: Consequences, cost of intermittence and benefit of forecasting', *Renew. Sustain. Energy Rev.*, vol. 87, no. August 2017, pp. 96–105, 2018, doi: 10.1016/j.rser.2018.02.007.

[3] IRENA, 'Flexibility in 21st Century Power Systems', 2014, doi: 10.2172/1130630.

[4] IRENA, *System Operation: Innovation Landscape Briefs*. 2020.

[5] H. Chandler, *A Guide to the Balancing Challenge*. 2011.

[6] IEA ETSAP and IRENA, 'Renewable Energy Integration in Power Grids. Technology Brief', no. April, pp. 1–36, 2015.

[7] B. Li, J. Shen, X. Wang, and C. Jiang, 'From controllable loads to generalized demand-side resources: A review on developments of demand-side resources', *Renew. Sustain. Energy Rev.*, vol. 53, pp. 936–944, 2016, doi: 10.1016/j.rser.2015.09.064.

[8] M. Radenković, Z. Bogdanović, M. Despotović-Zrakić, A. Labus, and S. Lazarević, 'Assessing consumer readiness for participation in IoT-based demand response business models', *Technol. Forecast. Soc. Change*, vol. 150, no. November 2019, p. 119715, 2020, doi: 10.1016/j.techfore.2019.119715.

[9] E. Klaasen and M. van der Laan, 'USEF White paper: Energy and Flexibility Services for Citizens Energy Communities', p. 19, 2019, [Online]. Available: https://www.nweurope.eu/media/6768/usef-white-paper-energy-and-flexibility-services-for-citizens-energy-communities-final-cm.pdf.

[10] U.S Department of Energy, 'How microgrids work'. https://www.energy.gov/articles/how-microgrids-work.

[11] S. Parhizi, H. Lotfi, A. Khodaei, and S. Bahramirad, 'State of the art in research on microgrids: A review', *IEEE Access*, vol. 3, pp. 890–925, 2015, doi: 10.1109/ACCESS.2015.2443119.

[12] O. Abrishambaf, M. A. F. Ghazvini, L. Gomes, P. Faria, Z. Vale, and J. M. Corchado, 'Application of a Home Energy Management System for Incentive-Based Demand Response Program Implementation', *Proc. - Int. Work. Database Expert Syst. Appl. DEXA*, pp. 153–157, 2017, doi: 10.1109/DEXA.2016.043.

[13] B. Pásztor, 'Design and implementation of a supervisory control and data acquisition system (SCADA) for a microgrid', no. June, 2018.

[14] IEA, 'Demand Response'. https://www.iea.org/reports/demand-response.

[15] C. W. Gellings, *Smart Grids: Enabling Energy Efficiency and Demand Response*. The Fairmont Press, 2009.

[16] SEDC, 'Explicit and Implicit Demand-Side Flexibility', *Position Pap.*, no. September, 2016, [Online]. Available: http://www.smartenergydemand.eu/wp-content/uploads/2016/09/SEDC-Position-paper-Explicit-and-Implicit-DR-September-2016.pdf.

[17] N. G. Paterakis, O. Erdinç, and J. P. S. Catalão, 'An overview of Demand Response: Key-elements and international experience', *Renew. Sustain. Energy Rev.*, vol. 69, no. July 2016, pp. 871–891, 2017, doi: 10.1016/j.rser.2016.11.167.

[18] G. De Zotti, 'Leveraging Consumers ' Flexibility for the Provision of Ancillary Services', 2019.

[19] P. Bertoldi, P. Zancanella, and B. Boza-Kiss, *Demand Response status in EU Member States*. 2016.

[20] L. Glover, M. Villa, L. Murley, J. Coelho, R. Adey-Johnson, and A. Pinto-Bello, 'Eu Market Monitor for Demand Side Flexibility', 2021, [Online]. Available: www.delta-ee.com.

[21] BOE 13/2011, 'BOE-A-2019-18423', *Boletín Oficial del Estado*. pp. 61561–61567, 2019.

[22] M. Barbero, C. Corchero, and L. Igualada, 'Estat de l'art d'altres tecnologies de generació i emmagatzematge i proveïdors de qualitat de subministrament per a l'edificació'. REFER (Reducció Energètica i Flexibilitat en Edificis en Rehabilitació), 2019.

[23] S. Mohagheghi and N. Raji, 'Managing industrial energy intelligently: Demand response scheme', *IEEE Ind. Appl. Mag.*, vol. 20, no. 2, pp. 53–62, 2014, doi: 10.1109/MIAS.2013.2288387.

[24] Eurostat, 'Energy consumption and use by households'. https://ec.europa.eu/eurostat/web/products-eurostat-news/-/DDN-20190620-1.

[25] M. Ali, A. Safdarian, and M. Lehtonen, 'Demand response potential of residential HVAC loads considering users preferences', *IEEE PES Innov. Smart Grid Technol. Conf. Eur.*, vol. 2015-Janua, no. January, pp. 1–6, 2015, doi: 10.1109/ISGTEurope.2014.7028883.

[26] S. C. Sugarman, *HVAC Fundamentals, Third Edition.* 2020.

[27] L. C. Casals, B. Amante García, and C. Canal, 'Second life batteries lifespan: Rest of useful life and environmental analysis', *J. Environ. Manage.*, vol. 232, no. November 2018, pp. 354–363, 2019, doi: 10.1016/j.jenvman.2018.11.046.

[28] USABC, 'Electric Vehicle Battery Test Procedures'. .

[29] L. Canals Casals, M. Barbero, and C. Corchero, 'Reused second life batteries for aggregated demand response services', *J. Clean. Prod.*, vol. 212, pp. 99–108, 2019, doi: 10.1016/j.jclepro.2018.12.005.

[30] International Renewable Energy Agency (IRENA), 'Aggregators', *Irena*, 2019, [Online]. Available: https://www.irena.org/-/media/Files/IRENA/Agency/Publication/2019/Feb/IRENA_Innovation_Aggregators_2019.PDF?la=en&hash=EB86C1C86A7649B25050F57799F2C0F609894A01.

[31] X. Lu, K. Li, H. Xu, F. Wang, Z. Zhou, and Y. Zhang, 'Fundamentals and business model for resource aggregator of demand response in electricity markets', *Energy*, vol. 204, no. May, p. 117885, 2020, doi: 10.1016/j.energy.2020.117885.

[32] L. Gkatzikis, I. Koutsopoulos, and T. Salonidis, 'The role of aggregators in smart grid demand response markets', *IEEE J. Sel. Areas Commun.*, vol. 31, no. 7, pp. 1247–1257, 2013, doi: 10.1109/JSAC.2013.130708.

[33] N. Mahmoudi, E. Heydarian-Forushani, M. Shafie-khah, T. K. Saha, M. E. H. Golshan, and P. Siano, 'A bottom-up approach for demand response aggregators' participation in electricity markets', *Electr. Power Syst. Res.*, vol. 143, pp. 121–129, 2017, doi: 10.1016/j.epsr.2016.08.038.

[34] P. Olivella-Rosell *et al.*, 'Local flexibility market design for aggregators providing multiple flexibility services at distribution network level', *Energies*, vol. 11, no. 4, pp. 1–19, 2018, doi: 10.3390/en11040822.

[35] 'openADR momentum'. https://www.openadr.org/index.php?option=com_content&view=article&id=191:openadr-momentum-2020&catid=21:press-releases&Itemid=121.

[36] 'OpenADR Alliance website'. https://www.openadr.org/.

[37] T. Péan and J. Salom, 'Laboratory facilities used to test energy flexibility in buildings', 2017, [Online]. Available: http://www.annex67.org/media/1373/lab-description-report-first-edition.pdf.

[38] A. Elias-alcega, M. Roman-barri, A. Ruiz-alvarez, I. Cairo-molins, and A. Sumper, 'Implementation of a Microgrid in Barcelona', *21 st Int. Conf. Electr. Distrib. Frankfurt*, no. 0342, pp. 6–9, 2011.

[39] A. Ruiz-Álvarez, A. Colet-Subirachs, F. Álvarez-Cuevas Figuerola, O. Gomis-Bellmunt, and A. Sudrià-Andreu, 'Operation of a utility connected microgrid using an IEC 61850-based multi-level management system', *IEEE Trans. Smart Grid*, vol. 3, no. 2, pp. 858–865, 2012, doi: 10.1109/TSG.2012.2187222.

[40] A. Ruiz-Álvarez *et al.*, 'Design, management and comissioning of a utility connected microgrid based on IEC 61850', *IEEE PES Innov. Smart Grid Technol. Conf. Eur. ISGT Eur.*, pp. 1–7, 2010, doi: 10.1109/ISGTEUROPE.2010.5638857.

[41] 'Bamboo Energy website'. https://bambooenergy.tech/en/platform/.

[42] 'InfluxDB'. https://influxdb-python.readthedocs.io/en/latest/.

[43] 'AVEVA Edge Resources'. https://sw.aveva.com/monitor-and-control/hmi-supervisory-and-control/intouch-edge-hmi/resources.

[44] M. H. Shamsi, W. O'Grady, U. Ali, and J. O'Donnell, 'A generalization approach for reduced order modelling of commercial buildings', *Energy Procedia*, vol. 122, pp. 901–906, 2017, doi: 10.1016/j.egypro.2017.07.401.

[45] H. Harb, N. Boyanov, L. Hernandez, R. Streblow, and D. Müller, 'Development and validation of grey-box models for forecasting the thermal response of occupied buildings', *Energy Build.*, vol. 117, pp. 199–207, 2016, doi: 10.1016/j.enbuild.2016.02.021.

[46] S. Jenu, I. Deviatkin, A. Hentunen, M. Myllysilta, S. Viik, and M. Pihlatie, 'Reducing the climate change impacts of lithium-ion batteries by their cautious management through integration of stress factors and life cycle assessment', *J. Energy Storage*, vol. 27, no. October 2019, p. 101023, 2020, doi: 10.1016/j.est.2019.101023.

[47] ESIOS (Red Eléctrica de España), 'Término de facturación de energía activa del PVPC'. https://www.esios.ree.es/es/pvpc?date=17-07-2020.

[48] ESIOS (Red Eléctrica de España), 'Precio de la energía excedentaria del autoconsumo para el mecanismo de compensación simplificada (PVPC)'. https://www.esios.ree.es/es/analisis/1739?vis=1&start_date=17-07-2020T00%3A00&end_date=19-07-2020T23%3A50&compare_start_date=16-07-2020T00%3A00&groupby=hour.

[49] N. Andreadou, 'Performance Measurement & Verification methodology report, DRIMPAC project'. 2019.

[50] M. Minou, G. Thanos, M. Vasirani, T. Ganu, M. Jain, and A. Gylling, 'Evaluating Demand Response Programs : Getting the Key Performance Indicators Right', *Int. Energy Work.*, no. i, pp. 1–10, 2011.

[51] 'openleadr webpage'. https://openleadr.org/docs/index.html (accessed Feb. 01, 2020).

[52] 'EMISIONES Y FACTOR DE EMISIÓN DE CO2 EQ. DE LA GENERACIÓN'. https://www.ree.es/es/datos/generacion/no-renovables-detalle-emisiones-CO2.

[53] Selectra, 'Tarifas eléctricas: precios y condiciones', [Online]. Available: https://tarifasgasluz.com/pymes/tarifas-luz/30.