



TwinERGY Integrated Solution

D8.3

DECEMBER 2022

Deliverable

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

DELIVERABLE REFERENCE NUMBER AND TITLE

D8.3 TwinERGY Integrated Solution

Revision: v1.0

AUTHORS

Ana Isabel Martínez García	Moisés Antón García
ETRA I+D	ETRA I+D



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

DISSEMINATION LEVEL

✓ P Public

C Confidential, only for members of the consortium and the Commission Services

Version History

REVISION	DATE	AUTHOR	ORG...	DESCRIPTION
V0.1	15.12.2022	Moisés Antón, Ana Isabel Martínez	ETRAID	Version to be reviewed by partners
V0.2	22.12.2022	Konstantinos Kotsalos; Apostolos Kapetanios	ED	Corrections within the document
V0.3	22.12.2022	Alexandros Tsitsanis	Suite5	Corrections within the text
V0.4	28.12.2022	Athanasios Chassiakos	UoP	Corrections within the text
V0.5	29.12.2022	Moisés Antón, Ana Isabel Martínez	ETRA ID	Final version with comments and corrections integrated sent to PC
V1.0	10.01.2023	Athanasios Chassiakos, Stylianos Karatzas, Vasiliki Lazari, Antonis Papamanolis	UoP	Final version sent to EC by the PC.

Statement of Originality

This deliverable contains original unpublished work except where clearly indicated otherwise.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

Legal Disclaimer

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

© 2022 by TwinENERGY Consortium

Executive summary

The output of Task 8.2 will be tested in real-life conditions in the pilot sites. With the feedback provided in that stage, the technical developers of TwinERGY will refine its individual modules and will also refine the integration of them. Thus, the output of this task will be a complete and mature TwinERGY solution ready to be demonstrated in the pilot sites in order to start collecting data valuable for the assessment of the project.

Index

1. Introduction	9
1.1. Scope of the document	9
1.2. Structure of the deliverable	9
1.3. Abbreviation list	10
2. Final pilot deployment	12
3. Integrated solution	28
3.1. TwinERGY architecture	28
3.2. Physical assets	34
3.3. Software modules developed during WP7	34
3.2. Identity Management Platform & Citizens Platform.....	35
3.2.1 Citizens Platform	36
3.2.2. Identity Management Platform.....	39
3.2.3. Identity Management schema	44
3.3. Interoperability platform.....	46
3.3.1. Messages	48
3.3.2. NATS console.....	49
3.4. Core Data Management Platform	52
3.5. Transactive Energy Blockchain Network.....	53
3.6. Digital Twin & iSCAN	55
4. Conclusions and next steps.....	57
References.....	58
Annex 1. Messages interchanged through Interoperability Platform.....	61

List of Figures

Figure 1. British pilot site diagram	16
Figure 2. German pilot site diagram	20
Figure 3. Greek pilot site diagram	24
Figure 4. Italian pilot site diagram.....	27
Figure 5. Logical schema of the architecture	29
Figure 6. General schema of TwinERGY.....	30
Figure 7. Home page in Citizens Platform	37
Figure 8. Registration form in Citizens Platform	38
Figure 9. Main screen in Citizens Platform	39
Figure 10. Operation of an IAM	41
Figure 11. Workflow with Identity Management Platform.....	45
Figure 12: Example of messages sending in NATS	47
Figure 13. Dashboard view in NATS console.....	49
Figure 14. Connection's view	50
Figure 15. Subscriptions view in NATS console.....	50
Figure 16. Message's view in NATS console.....	51
Figure 17. Window to select one existing subject.....	51
Figure 18. Window to send messages in interoperability platform	52
Figure 19: TwinERGY Core Data Management Platform Conceptual Architecture	52
Figure 20. Workflow in Transactive Energy Blockchain network	55

List of Tables

Table 1: Abbreviation List.....	10
Table 2. Use cases implementation per pilot site	13
Table 3. Components list in British pilot site.....	14
Table 4. Components related to German Pilot	17
Table 5. Components related to Greek Pilot.....	21
Table 6. Components in Italian pilot site	25
Table 7. Compiled list of components in TwinERGY	31
Table 8. Comparison of IAMs open tools (1)	42
Table 9. Comparison of IAMs open tools (2)	42

1. Introduction

1.1. Scope of the document

This document is a summary of the integrated solution for TwinERGY, where modules, services and physical components in pilot sites are working together in order to increase the relevance of the Demand Response optimization tools and incorporate strategies in the new generation of energy management systems

The introduction of a Digital Twin framework incorporates the required intelligence for optimizing that Demand Response at the local level without compromising the well-being of consumers and their daily schedules and operations.

In this deliverable we expose the different components implemented and deployed in each of the pilot sites, being one of the two documents that summarizes the work done during Task T8.2 (TwinERGY system modules integration and lab-testing). This work can be considered as the realization of the design created during Task T4.4 (System's architecture design), and developments made during WP5, WP6 and WP7.

1.2. Structure of the deliverable

The structure of this deliverable is the following one. Section 2. Final pilot deployment is the description of the solution deployed in each pilot site. Section 3. Integrated solution is the description of the integrated solution, including subsections per each component of TwinERGY. Section 4. Conclusions and next steps describes the Next steps to be executed since now to the end of the project and Section includes the conclusions of the work exposed here.

1.3. Abbreviation list

Table 1: Abbreviation List

Acronym	Full Name
AI	Artificial Intelligence
BMS	Building Management System
CAS	Central Authentication Service
CDMP	Core Data Management Platform
DLT	Distributed Ledger Techonology
DoA	Description of Action
DT	Digital Twin
HEMS	Home Energy Management System
IAM	Identification and Access Management
LV	Low Voltage
MFA	Multifactor Authentication
MV	Medium Voltage
SGAM	Smart energy Grid Architecture Model

SSO	Single Sign-On
TEM	Transactive Energy Platform

2. Final pilot deployment

In this section we expose the characteristics of each pilot site from the architectural point of view. This section is strongly linked with the WP9 work, but it is not our aim to replace the information explained in deliverables belonging to that work package. On the other hand, we explained a first version of the architecture in deliverable D4.4 [1], so the starting point of this deliverable is that.

As it is defined in the DoA of TwinENERGY, the four pilot sites are covering nine Use cases, not being considered all of them for all pilot sites:

- UC01 - “Home Energy Management”: Energy management at the residential consumer premises to maximize self-consumption and self-sufficiency.
- UC02 - “RES Generation in domestic and tertiary buildings”: Creation of further renewable sources and infrastructure to increase the RES share in public and private buildings.
- UC03 - “Grid capacity enhancement utilizing e-mobility”: Testing of how the Electric Vehicles (EVs) can be a distributed storage asset, able to stabilize the grid and lead to more decarbonized neighbourhoods.
- UC04 - “Prosumers empowerment in local energy trading markets”: Provision of solutions to transactive energy uses cases and enables grid decentralization and democratization by connecting the micro-grid operators to the DER managers and their customers.
- UC05 - “Enhance grid’s flexibility through DER Management”: Improving of grid flexibility and grid stability and to increase the share of RES through the management.
- UC06 - “Consumers engagement in Demand Side Management Programs utilizing feedback mechanisms”: Feedback-based demand-side intervention strategies applied at the residential level.
- UC07 - “Consumers engagement in Demand Side Management Programs utilizing a socio-economic context”: Enable Social context drivers for energy-related behaviour changes, by utilization of social interactions and cultural values to influence energy exchanges between households, and consumer attitudes towards benefit and comfort
- UC08 - “Consumer’s engagement in demand response programs utilizing personalized/health-oriented services”. Obtain the consumers’ realistic comfort/wellbeing level with minimum intervention.

- UC09 – “Consumer’s engagement in demand response programs utilizing digital twins’ prediction capabilities for dynamic VPPS”: Considering a virtual power plant (VPP) as a distributed power plant that aggregates the capacities of heterogeneous distributed energy resources (DER), this use case includes a human-machine interface of the digital twins to manage it by both consumers and communities, in an interactive way.

The next table depicts what use cases are going to be implemented in each pilot site.

Table 2. Use cases implementation per pilot site

Use Case	Bristol (United Kingdom)	Hagedorn (Germany)	Athens (Greece)	Benetutti (Italy)
UC01	X	X	X	X
UC02	X	X		X
UC03		X	X	
UC04	X			X
UC05		X		X
UC06			X	X
UC07	X	X		X
UC08		X	X	X
UC09	X	X	X	X

As it can be deduced, the physical assets as well as the TwinERGY software components in each pilot site are different, depending on the use cases that are to be implemented per each of them.

Next, we describe the items considered per each pilot site. Later on, in 3. Integrated solution we will expose the final configuration per TwinERGY system, where those components are detailed.

British pilot

The pilot located in Bristol, is focused on South Bristol areas as well as other potential areas of the city including the University Campus in Clifton. Working with a selected sample of households in Bristol, as well as within University of Bristol campus

infrastructure, local partners utilise TwinERGY modules to understand the function of digital twins as part of new models of energy resource management and consumer engagement.

Considering that this pilot site is participating in UC01, UC02, UC04, UC07 and UC09, we can enumerate these components as those presented in this pilot site.

Table 3. Components list in British pilot site

Component	Component Type
Energy asset/appliance	Physical asset
Energy smart plugs	Physical asset
Environmental sensors	Physical asset
Home battery storage	Physical asset
LV Grid	Physical asset
PV Panel	Physical asset
PV smart meter	Physical asset
Raspberry Pi	Physical Asset
Raspberry Pi (gateway)	Physical asset
Smart Battery	Physical Asset
Smart Meter	Physical Asset
Comfort Well-being module	TwinERGY application
Consumer Demand Flexibility Profiling Module	TwinERGY application
Core Data Management Platform	TwinERGY application
Digital Twin Platform	TwinERGY application
Home & Tertiary Energy Monitoring Module	TwinERGY application
Interoperability Platform	TwinERGY application
iSCAN	TwinERGY application
Identity Management Platform	TwinERGY application

Citizens Platform	TwinERGY application
RES Integration & DER Management Module	TwinERGY application
Risk Management Module	TwinERGY application
Social Network Module	TwinERGY application
Social Network Module GUI	TwinERGY application
Transactive Energy Blockchain Network	TwinERGY application
Transactive Energy Platform	TwinERGY application
ENTSO-E transparency platform	External Application
HEMS	External Application
HEMS gateway	External application
Raspberry Pi Oracle	External application

The relationship among components in this pilot site can be depicted in next Figure 1 which contains a schema extracted from the SGAM diagrams explained in D4.4.

The TwinERGY software components are placed on the top of the figure, while physical assets and gateways to transmit information from them to the Core Data Management platform are placed at the bottom. Red lines represent physical connections such as the electrical distribution network, the connection between a sensor and its controller or the connection between a PV panel and the LV grid. On the other hand, blue lines represent connections related to logic connections. This can be a wireless connection between a controller in a home and the Home Energy Management system or the internet link between a module (e.g., the Social Network Module) and another one (e.g., the Interoperability Platform).

Due to the fact that this pilot site is oriented to use cases mainly related to energy management, Demand Response and RES maximization, assets like community and home batteries or PV panels have a great presence.

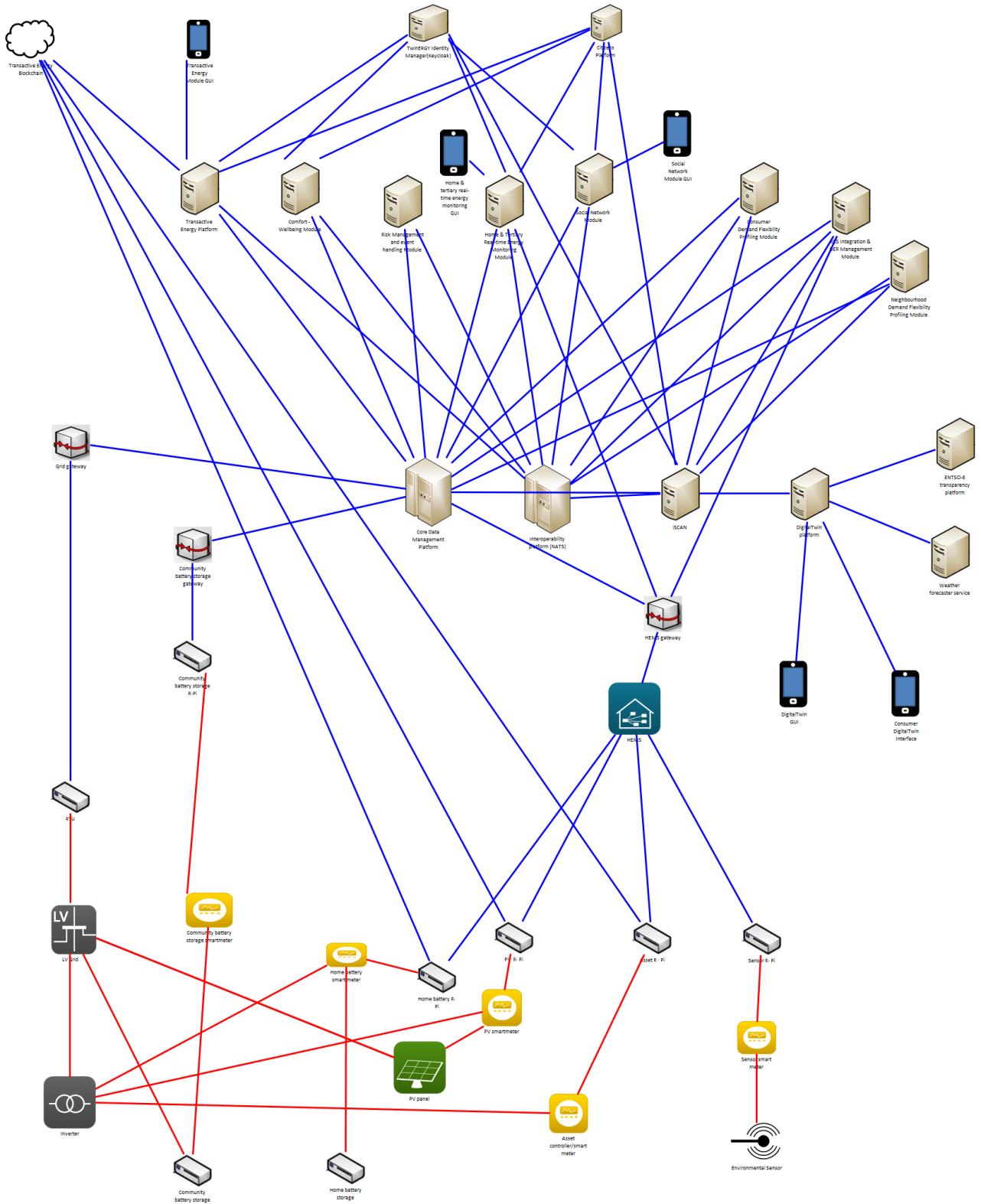


Figure 1. British pilot site diagram

German pilot

The village of Hagedorn in Steinheim is located at the district of Höxter, North Rhine-Westphalia, where the medium voltage grid is supplied with electricity by two local network stations. This pilot is oriented to a climate protection plan, in the way of achieving efficient energy use and a higher degree of integration of renewable energies as well as a significant increase in consumer interest in energy-related measures. Besides, the use of electric vehicles with intelligent charging and VehicleToGrid (V2G) technologies allow to demonstrate the capacity increase of the network.

This schema is covered by the participation of the site in the use cases UC01, UC02, UC03, UC05, UC07, UC08, UC09. For them, the components included are the following ones:

Table 4. Components related to German Pilot

Component	Component Type
Charging Point	Physical asset
Community battery storage	Physical asset
Electric Vehicle	Physical asset
Energy asset/appliance	Physical asset
Energy smart plugs	Physical asset
Environmental sensor	Physical asset
Home battery storage	Physical asset
Inverter	Physical asset
LV Grid	Physical asset
MV Grid	Physical asset
MV-LV Transformer	Physical Asset
Physiological sensor	Physical asset
PV Panel	Physical asset
PV smart meter	Physical asset
Raspberry Pi	Physical asset
Raspberry Pi (gateway)	Physical asset

RTU	Physical asset
Smart Meter	Physical asset
Wearable device	Physical asset
Wind Farm	Physical asset
Consumer Demand Flexibility Profiling Module	TwinERGY application
Transactive Energy Platform	TwinERGY application
Comfort Well-being module	TwinERGY application
Consumer Comfort/Well-being Module	TwinERGY application
Consumer Digital Twin platform	TwinERGY application
Core Data Management Platform	TwinERGY application
Digital Twin Platform	TwinERGY application
Home & Tertiary Energy Monitoring Module	TwinERGY application
Interoperability Platform	TwinERGY application
iSCAN	TwinERGY application
Neighbourhood Demand Flexibility Profiling Module	TwinERGY application
RES Integration & DER Management Module	TwinERGY application
Risk Management Module	TwinERGY application
Social Network Module	TwinERGY application
Transactive Energy Blockchain Network	TwinERGY application
TwinEV Module	TwinERGY application
Charging Point Operator Gateway	External application
ENTSO-E transparency platform	External Application
Grid gateway	External application
HEMS	External Application
HEMS Gateway	External application
Weather Forecast Service	External application

Charging Point Operator	Organization
-------------------------	--------------

The relationship among components in this pilot site can be displayed in Figure 2, which contains a schema extracted from the SGAM diagrams explained in D4.4.

The TwinERGY software components are placed on the top of the figure, while physical assets and gateways to transmit information from them to the Core Data Management platform are placed at the bottom. Red lines represent physical connections such as the electrical distribution network, the connection between a sensor and its controller or the connection between a PV panel and the LV grid. On the other hand, blue lines represent connections related to logic connections. This can be a wireless connection between a controller in a home and the Home Energy Management system or the internet link between a module (e.g., the Social Network Module) and another one (e.g., the Interoperability Platform).

We remark here that charge points and components linked to the RES generation are displayed since this pilot site is focused on use cases primarily related to the renewable energy and the electromobility services.

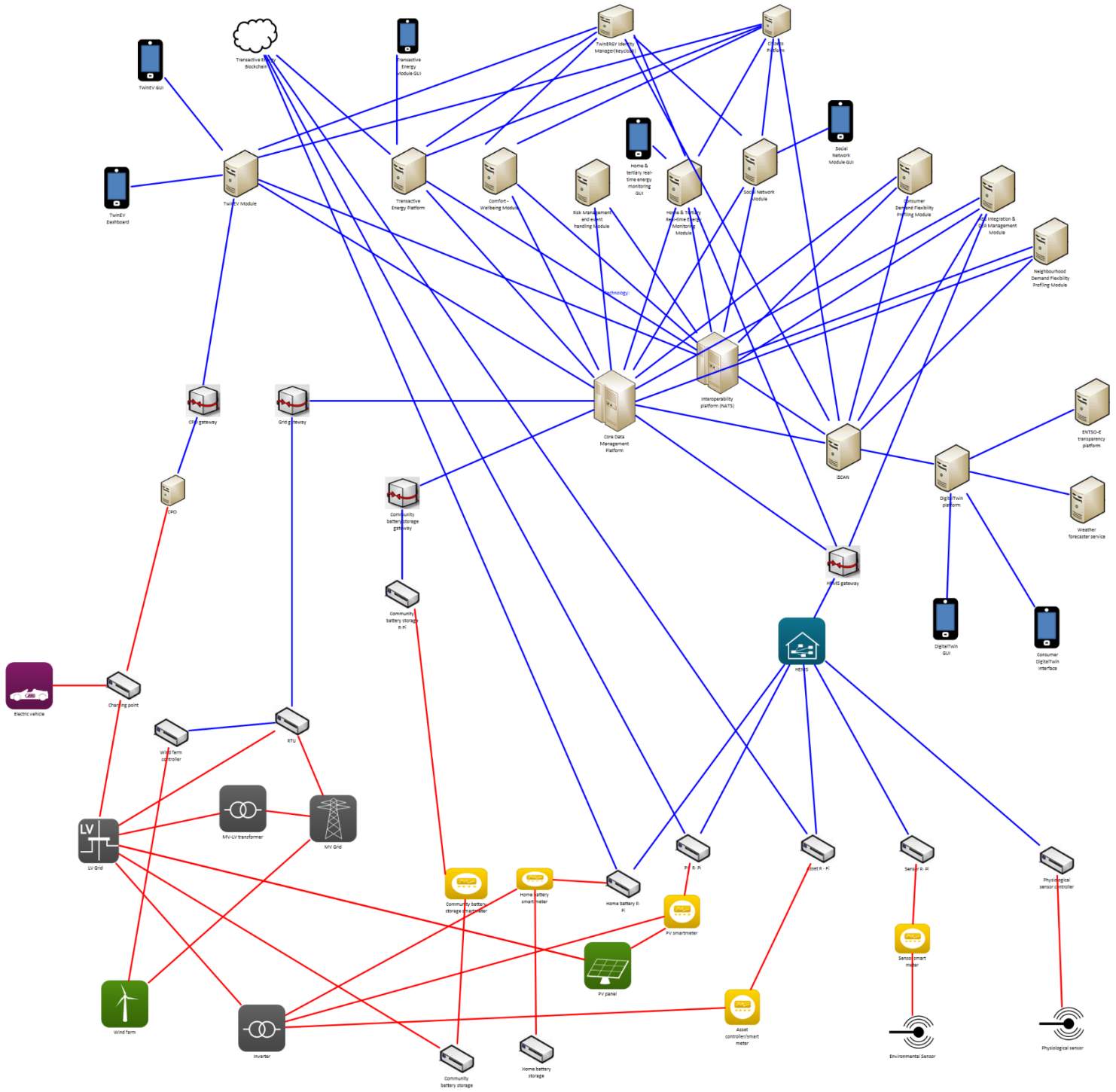


Figure 2. German pilot site diagram

Greek pilot

The Greek pilot site involves a group of residential and commercial buildings located in Athens, Greece. These buildings are already – partially - equipped with a variety of sensors and smart meters/ actuators, including temperature, humidity, luminance and CO2 sensors, smart thermostats, smart dimmers and plug meters enabling the measurement of electricity consumption at device level and allowing for the accurate profiling of their energy behaviour and (sub-sequently) their flexibility, in a non-intrusive and highly effective and engaging manner.

The Greek demo case is oriented to the implicit demand response programs, dynamic pricing schemes, feedback mechanisms and human-centric features that allow consumers to alter their energy consumption patterns and provide flexibility to the electricity retailer, without compromising their comfort and wellbeing. This pilot site considers as well a set of public Charge points for electrical vehicles.

The Athens demo site is participating in the next use cases: UC01, UC03, UC06, UC08, UC09, that includes the components indicated in *Table 5*.

Table 5. Components related to Greek Pilot

Component	Component Type
Energy smart plugs	Physical asset
Environmental sensors	Physical asset
PV Panel and Inverter	Physical asset
Home battery storage	Physical asset
Home storage smart meter	Physical asset
Raspberry Pi (gateway)	Physical asset
LV Grid	Physical asset
Charging Point	Physical asset
Community battery storage	Physical asset
Community battery storage smart meter	Physical asset
Electric Vehicle	Physical asset
Home battery storage	Physical asset

Home storage smart meter	Physical asset
Inverter	Physical asset
LV Grid	Physical asset
MV Grid	Physical asset
MV-LV Transformer	Physical Asset
PV Panel	Physical asset
Raspberry Pi	Physical asset
RTU	Physical asset
Energy Light	Physical asset
Home Battery Storage	Physical asset
Physiological sensor	Physical asset
Environmental sensor	Physical asset
Wearable device	Physical asset
Smart Meter	Physical asset
Energy asset	Physical assets
Core Data Management Platform	TwinERGY application
Consumer Demand Flexibility Profiling Module	TwinERGY application
Digital Twin Platform	TwinERGY application
iSCAN	TwinERGY application
Interoperability Platform	TwinERGY application
Home & Tertiary Energy Monitoring Module	TwinERGY application
Risk Management Module	TwinERGY application
Consumer Demand Flexibility Profiling Module	TwinERGY application
RES Integration & DER Management Module	TwinERGY application
Identity Management Platform	TwinERGY application
Consumer Demand Flexibility Profiling Module	TwinERGY application

Interoperability Platform	TwinERGY application
Neighbourhood Demand Flexibility Profiling Module	TwinERGY application
RES Integration & DER Management Module	TwinERGY application
Transactive Energy Blockchain Network	TwinERGY application
Transactive Energy Platform	TwinERGY application
TwinEV Module	TwinERGY application
Interoperability Platform	TwinERGY application
Consumer Comfort/Well-being Module	TwinERGY application
ENTSO-E transparency platform	External Application
HEMS	External Application
Charging Point Operator Gateway	External application
ENTSO-E transparency platform	External Application
Grid gateway	External application
HEMS	External Application
HEMS Gateway	External Application

The relationship among components in this pilot site can be displayed in Figure 3, which contains a schema extracted from the SGAM diagrams explained in D4.4.

The TwinERGY software components are placed on the top of the figure, while physical assets and gateways to transmit information from them to the Core Data Management platform are placed at the bottom. Red lines represent physical connections such as the electrical distribution network, the connection between a sensor and its controller or the connection between a PV panel and the LV grid. On the other hand, blue lines represent connections related to logic connections. This can be a wireless connection between a controller in a home and the Home Energy Management system or the internet link between a module (e.g., the Social Network Module) and another one (e.g., the Interoperability Platform).

In the physical assets, we can highlight that the sensor meters in buildings and the use of renewable energy where it is possible are included in the architecture. The schema also includes the electrical vehicle and the infrastructure for charging.

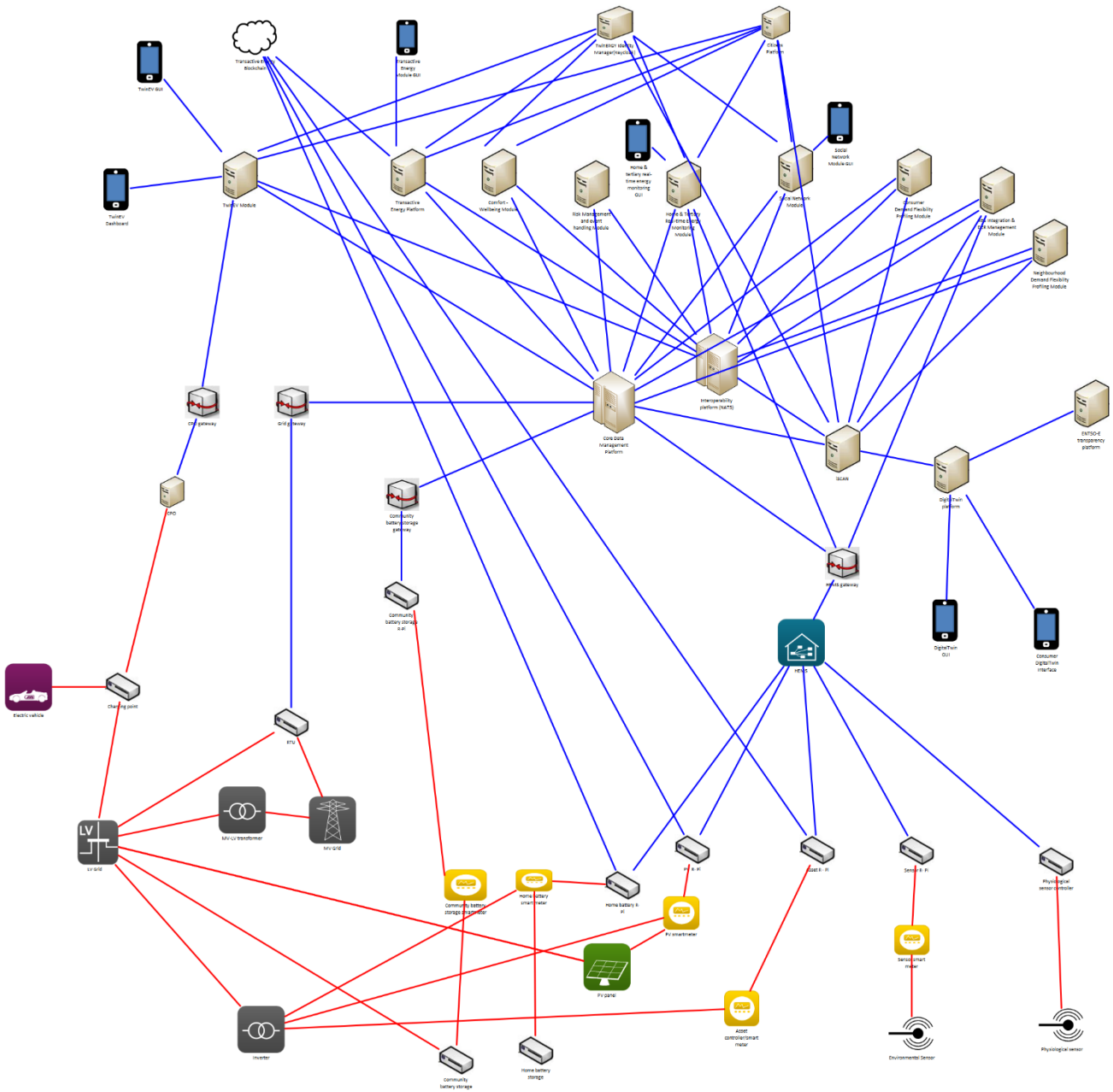


Figure 3. Greek pilot site diagram

Italian pilot

Benetutti is a smart energy community with concession for the distribution of electricity on medium and low voltage distribution networks, being the electricity delivered to final

customers. The place can generate renewable energy through PV panels. This pilot site is focused on RES generation, Demand Response programmes and the exchange of energy between different actors.

Benetutti is participating in all use cases, except UC03, since the community does not have charging points for electrical vehicles in the town. The list of components is included in the next table:

Table 6. Components in Italian pilot site

Component	Component Type
Community Battery Storage	Physical asset
Energy asset/appliance	Physical asset
Energy Light	Physical asset
Energy smart plugs	Physical asset
Environmental sensor	Physical asset
Home Battery Storage	Physical asset
Smart meter	Physical asset
LV Grid	Physical asset
Physiological sensor	Physical asset
PV Panels	Physical asset
Raspberry Pi	Physical Asset
Raspberry Pi (gateway)	Physical asset
Smart Battery	Physical Asset
Wearable device	Physical asset
Wind Farm	Physical asset
Core Data Management Platform	TwinERGY application
Transactive Energy Market (TEM) Connected	TwinERGY application
Consumer Comfort/Well-being Module	TwinERGY application
Digital Twin Platform	TwinERGY application

Home & Tertiary Energy Monitoring Module	TwinERGY application
Interoperability Platform	TwinERGY application
iSCAN	TwinERGY application
RES Integration & DER Management Module	TwinERGY application
Risk Management Module	TwinERGY application
Social Network Module	TwinERGY application
Social Network Module GUI	TwinERGY application
Transactive Energy Blockchain Network	TwinERGY application
Transactive Energy Platform	TwinERGY application
Identity Management Platform	TwinERGY application
ENTSO-E transparency platform	External Application
HEMS	External Application
HEMS gateway	External application
Raspberry Pi Oracle	External application
Weather Forecast Service	External application

The relationship among components in this pilot site can be displayed in Figure 4, which contains a schema extracted from the SGAM diagrams explained in D4.4.

The TwinERGY software components are placed on the top of the figure, while physical assets and gateways to transmit information from them to the Core Data Management platform are placed on the bottom. Red lines represent physical connections such as the electrical distribution network, the connection between a sensor and its controller or the connection between a PV panel and the LV grid. On the other hand, blue lines represent connections related to logic connections. This can be a wireless connection between a controller in a home and the Home Energy Management system or the internet link between a module (e.g., the Social Network Module) and another one (e.g., the Interoperability Platform).

When it comes to the physical assets, we may emphasize the usage of renewable energy sources that are included into the architecture.

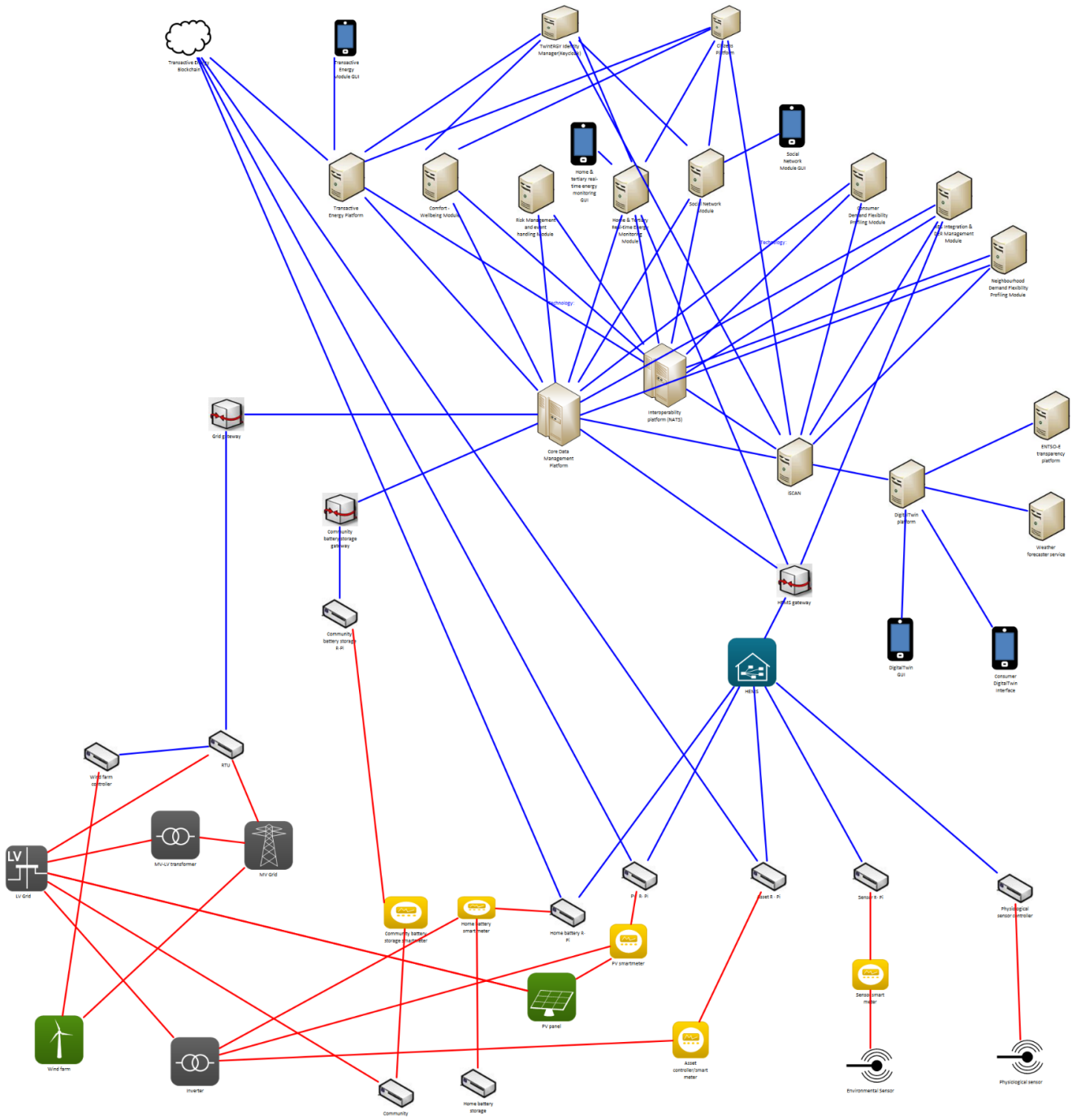


Figure 4. Italian pilot site diagram

3. Integrated solution

3.1. TwinERGY architecture

Considering that each pilot site presents differences with respect to the other sites, the TwinERGY architecture is a complete solid solution where software modules and physical assets are communicating among them through common tools. Users only communicate to TwinERGY services that apply in their pilot site, and these modules are in charge of communicate with the rest of components. In this way, users are not concerned about services that they do not need to use, even they do not need to know about the existence of these other services.

This is logical represented in the next Figure 5. On the other side, the following Figure 6 depicts the real architecture of the TwinERGY project, regardless of the pilot site.

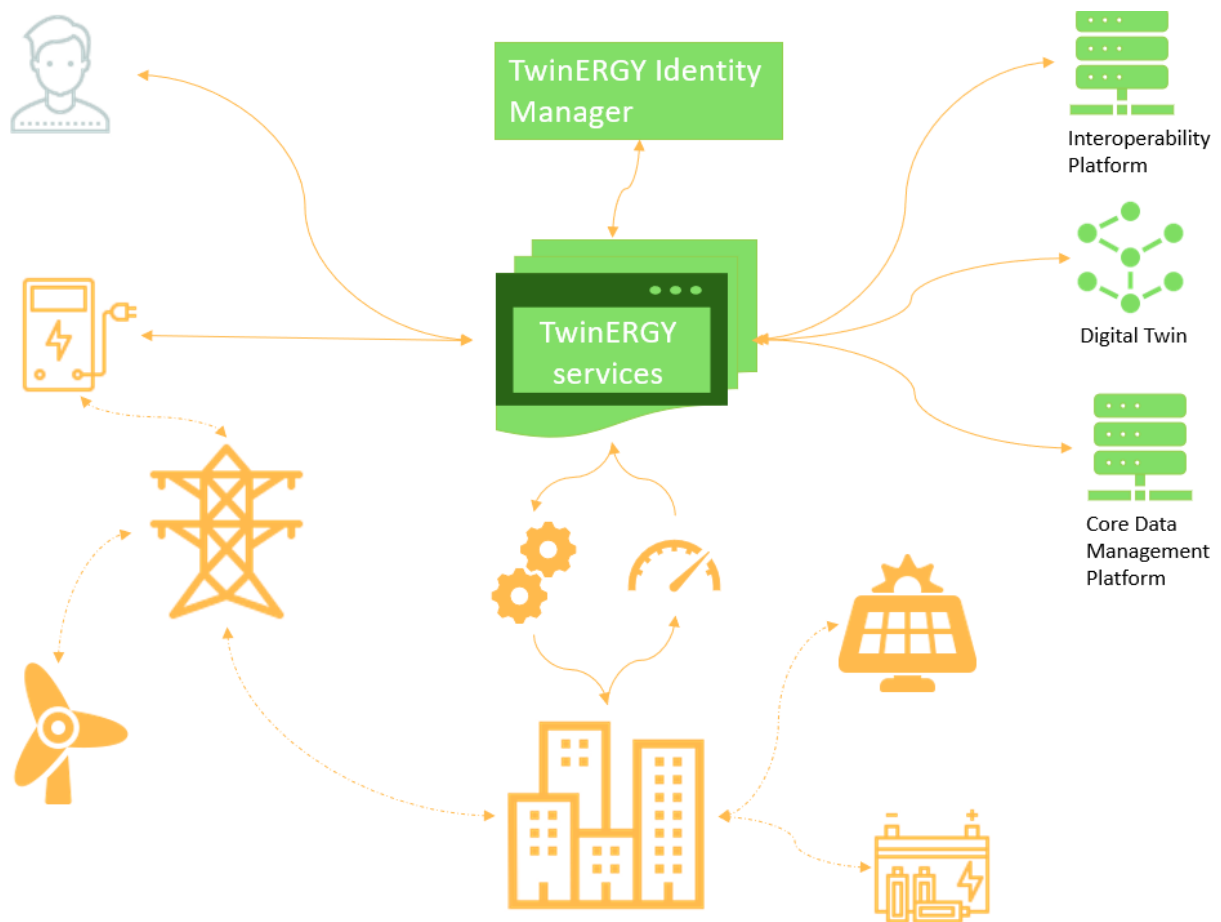


Figure 5. Logical schema of the architecture

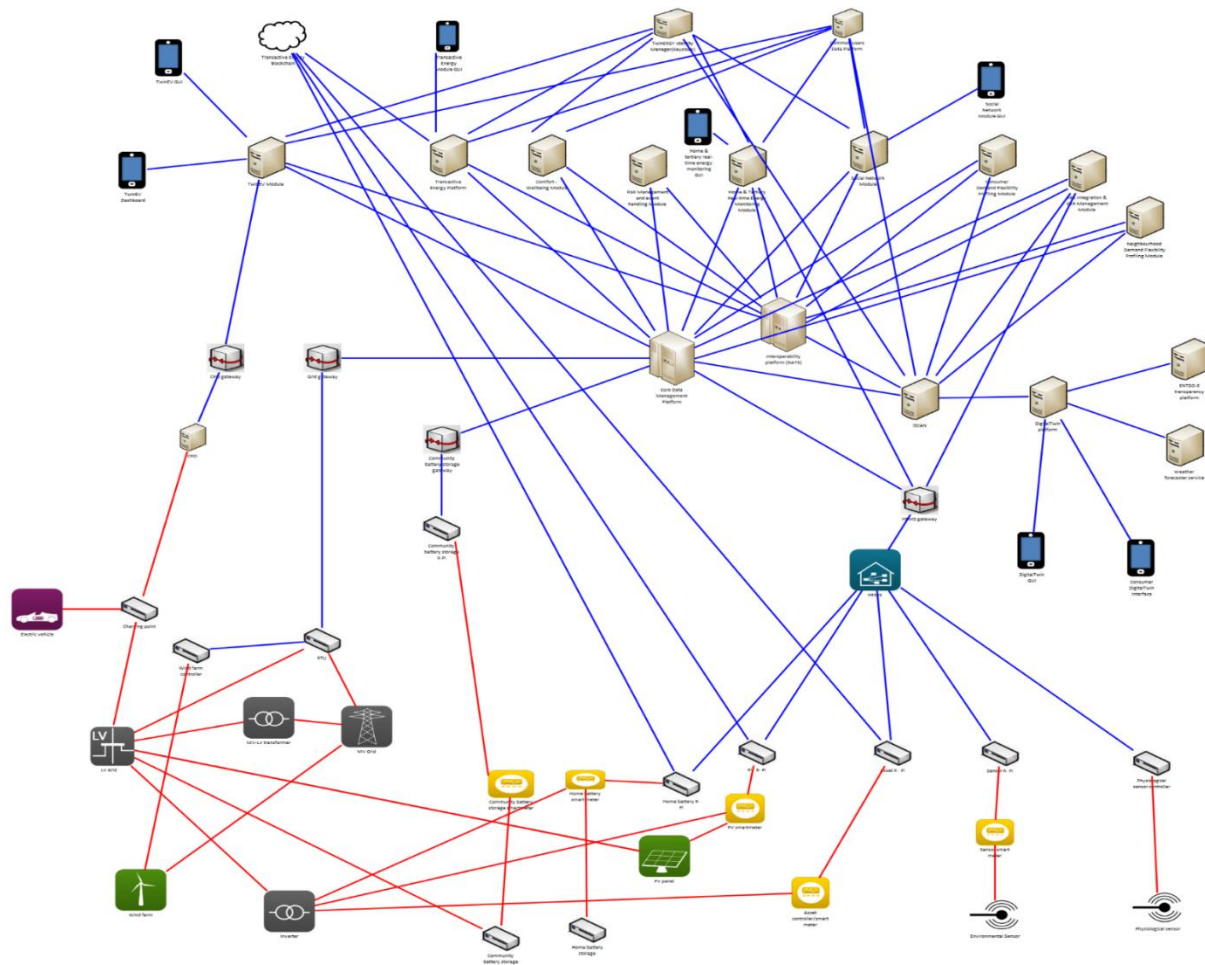


Figure 6. General schema of TwinERGY

As it has been commented in the previous section 2. *Final pilot deployment*, the TwinERGY software components are placed on the top of the figure, while physical assets and gateways to transmit information from them to the Core Data Management platform are placed at the bottom. Red lines represent physical connections such as the electrical distribution network, the connection between a sensor and its controller or the connection between a PV panel and the LV grid. On the other hand, blue lines represent connections related to logic connections. This can be a wireless connection between a controller in a home and the Home Energy Management system or the internet link between a module (e.g., the Social Network Module) and another one (e.g., the Interoperability Platform).

As commented before, a detailed report of this architecture was presented in D4.4. However, some minor variations have appeared during the development and integration of components. The complete list of elements participating in TwinERGY project is summarized in the following *Table 7*. Compiled list of components in TwinERGY, and the description of each of them is done in the next subsections, being them the next ones:

- Physical assets: Sensors, smart meters, PV panels, etc.
- Software modules developed during WP7
- Common software components: Interoperability Platform, Identity Management Platform or Citizens Platform
- External resources, such as the external service of weather forecasting.

Table 7. Compiled list of components in TwinERGY

Component	Component Type
Charging Point	Physical asset
Community battery storage	Physical asset
Community battery storage smart meter	Physical asset
Electric Vehicle	Physical asset
Energy asset/appliance	Physical asset
Energy Light	Physical asset
Energy smart plugs	Physical asset
Environmental sensor	Physical asset
Environmental sensors	Physical asset
Home battery storage	Physical asset
Home storage smart meter	Physical asset

Inverter	Physical asset
LV Grid	Physical asset
MV Grid	Physical asset
MV-LV Transformer	Physical Asset
Physiological sensor	Physical asset
PV Panel	Physical asset
PV Panel and Inverter	Physical asset
PV Panels	Physical asset
PV smart meter	Physical asset
Raspberry Pi	Physical Asset
Raspberry Pi (gateway)	Physical asset
RTU	Physical asset
Smart Battery	Physical Asset
Smart Meter	Physical Asset
Wearable device	Physical asset
Wind Farm	Physical asset
Energy asset	Physical assets
Transactive Energy Market (TEM) Connected	TwinERGY application
Citizens Platform	TwinERGY application
Comfort Well-being module	TwinERGY application
Consumer Comfort/Well-being Module	TwinERGY application
Consumer Demand Flexibility Profiling Module	TwinERGY application
Consumer Digital Twin platform	TwinERGY application
Core Data Management Platform	TwinERGY application
Digital Twin Platform	TwinERGY application

Home & Tertiary Energy Monitoring Module	TwinERGY application
Identity Management Platform	TwinERGY application
Interoperability Platform	TwinERGY application
Interoperability Platform	TwinERGY application
iSCAN	TwinERGY application
Neighbourhood Demand Flexibility Profiling Module	TwinERGY application
RES Integration & DER Management Module	TwinERGY application
Risk Management Module	TwinERGY application
Social Network Module	TwinERGY application
Social Network Module GUI	TwinERGY application
Transactive Energy Blockchain Network	TwinERGY application
Transactive Energy Platform	TwinERGY application
TwinEV Module	TwinERGY application
Charging Point Operator Gateway	External application
ENTSO-E transparency platform	External Application
Grid gateway	External application
HEMS	External Application
HEMS gateway	External application
Raspberry Pi Oracle	External application
Weather Forecast Service	External application
Charging Point Operator	Organization

3.2. Physical assets

This kind of components refer to components that are physically installed on a demo site, and can be used for different tasks.

In addition to batteries and/or PV panels, pilots responsible have installed different components to measure the used, generated or stored energy and/or the conditions in the environment. This is the case of Smart meters, environmental sensors or physiological sensors.

In the opposite way, the controllers of the physical assets are needed, in order to implement the Demand Response feature. Here we can find Raspberry Pi and other controllers.

Besides, if we are speaking of energy generation and consumption, the communication with the grid is important, so the architecture includes components like inverters or transformers MV-LV. In relation to the grid, charge points are included in order to allow the use of electromobility.

Finally, all these components are not isolated in the schema, but they are communicating with the rest of components through gateways. We can enumerate here the Home Energy Management System (HEMS), grid gateways and community batteries gateways.

3.3. Software modules developed during WP7

The smart control of the physical assets goes through several tools implementing the different services offered by TwinERGY. Those tools have been developed during WP7 by different partners, being called TwinERGY Modules:

- M1. Consumer Comfort / Well-being module: Based on user profiles and supported by autonomous wearable devices, this module not only depicts the comfort / wellbeing level of the consumers, but it notifies about optimal energy usage while preserving comfort and well-being levels is feasible.
- M2. Consumer demand flexibility profiling Module: This module has been developed to enable the calculation of flexibility for individual buildings/consumers, for both implicit and explicit demand response. Beyond the estimated amount of flexibility itself, the controllability and predictability features are also estimated to allow better informed decisions to be taken from the aggregators/prosumers.
- M3. Neighbourhood demand flexibility profiling Module: This module is similar to the previous one, but it is focused on the whole neighbourhood.

- M4. Home & Tertiary real-time Energy Monitoring Module: This module leverages real-time data from buildings and devices to optimize energy consumptions of them
- M5. RES integration & DER management Module: It generates a real-time image of the electrical grid that is as accurate as possible. Internal algorithms control each storage unit in the grid individually, so that a Virtual Power Plant (VPP) can be operated with the lowest possible storage losses.
- M6. Risk Management and event handling Module: This module helps to identify the security gaps, supports the security operation planning and management and helps draw up the adequate preventive actions and countermeasures.
- M7. TwinEV Module: It supports the set of functionalities related to the electromobility, from smart charging of an electric vehicle in a charge station, to limit the supply power in those stations, as well as including the electric vehicle into the VPP when problems in grid appear.
- M8. Social Network Module: It is oriented to prosumers engagement and their participation in programs of rewards, in order to maximize energy efficiency.
- M9. Transactive Energy Module: Based on Hybrid Blockchain technologies, this module solves current intractable optimization problems and creates a premiere Transactive Energy (TE) protocol layer settlement process, marketplace, and governance framework to allow energy-related apps to be written and interoperate with each other.

3.2. Identity Management Platform & Citizens Platform

Since the services/modules of TwinERGY are offered to the end users and they are all deployed in different servers or on the cloud, a traditional schema implies that a user has to register and identify in each of these services, even if the modules are sharing information and communicating among them.

It is more an issue of usefulness than just seeming as though the modules are unconnected to one another. For a user that has to access various project services, the solution involves a single registration and a single login. The solution pass by a unique register and a single login for a user that needs to access to different services of the project. Besides, the same schema is needed in order to execute the log-out and the authorization process.

Therefore, we have created the Identity Management Platform and the Citizens Platform, which present an environment where end-users participating of TwinERGY project only have to register once for accessing to all services or modules in the project and provide the possibility of login for all the modules in an integrated manner.

On one side, the Identity Management Platform is the way where the applications can authenticate what user is trying to access to them. Based on Keycloak technology, it is a server where an application can ask for the validation of the identity of a user and the permission of accessing to restricted resources.

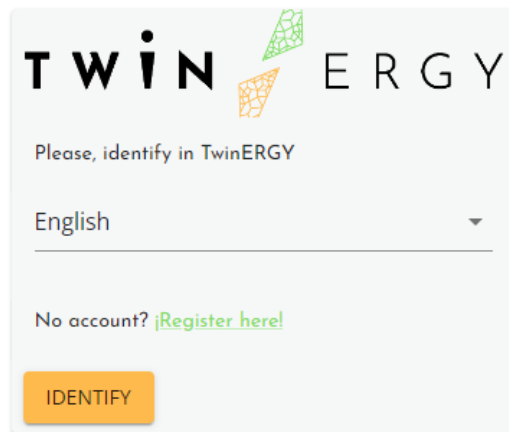
On the other side, the Citizens Platform is the place where an end-user can self-register in TwinERGY and maintain information that is going to be used by the modules. Such information can be the electric vehicle that a person uses, or the batteries installed at home. Besides, each user can go out of the project, so the self-deletion of accounts is present too. In the next subsections we detail each aspect necessary for maintaining this ecosystem.

3.2.1 Citizens Platform

The Citizens Platform is a multilanguage and responsive web site where users can register for their participation into TwinERGY and manage their data. Internally, the same Keycloak server is used for verification of identities and registrations. The application manages data related to end-users that are going to be shared by modules developed during WP7. These data are stored in a MongoDB placed in ETRA servers. The platform is accessible in this URL: https://twinergy_citizens.tec.etra-id.com/

Since the fact that end-users are the owner of their data, the web site will allow self-register, self-management of their data and self-deletion of their accounts. Therefore, the access to the platform is through a home page, showed in Figure 7. This home page allows two actions: the login for identifying and enter into the platform (button "Identify") and the self-registration for creating an account (link "Register here!").

The login page consists of a form launched by Keycloak while the registration page is a form with the minimal information required to create the account (Figure 8).



The image shows a login interface for TwinEnergy. At the top, the logo 'TWIN ENERGY' is displayed with a stylized lightning bolt icon. Below the logo, the text 'Please, identify in TwinENERGY' is centered. Underneath, there is a language selection dropdown menu currently set to 'English'. Below the dropdown, the text 'No account? [Register here!](#)' is displayed. At the bottom of the form is an orange button labeled 'IDENTIFY'.


 THIS PROJECT HAS RECEIVED FUNDING UNDER THE EUROPEAN UNION'S HORIZON 2020 RESEARCH AND INNOVATION PROGRAMME - NO 957736.

Figure 7. Home page in Citizens Platform


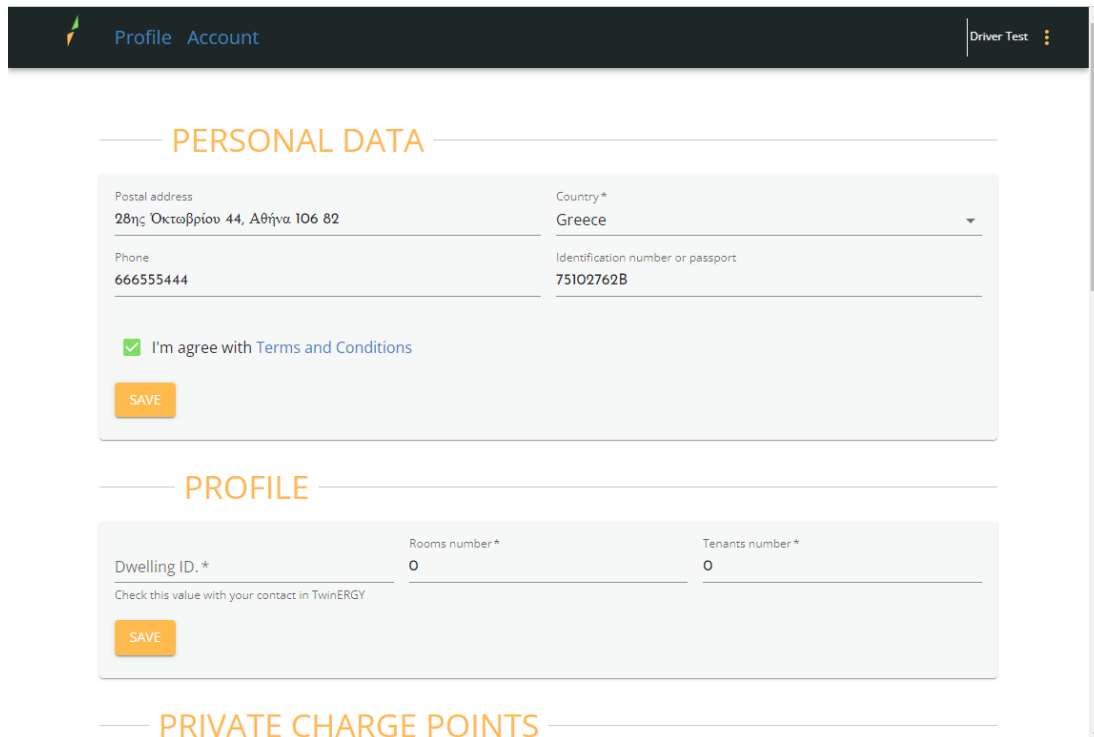
 THIS PROJECT HAS RECEIVED FUNDING UNDER THE EUROPEAN UNION'S HORIZON 2020 RESEARCH AND INNOVATION PROGRAMME - NO 957736.

Figure 8. Registration form in Citizens Platform

After filling out the registration form, the user clicks on the button “Create an account”. If all information is correct and the email or username have not been registered before by another user, the account is created but the user needs to activate it before next 24 hours, or it will be automatically removed. This is done so in order to avoid automatic registrations and accounts that never are used.

In order to activate the account, right after the creation, the system sends an automatic email with a code that the user has to insert into the same form so that the account becomes active. As it has been commented, if the user does not insert the code, the account is removed after 24 hours from the request of registration.

On the other side, once users are logged into the platform, they can manage their profile data. This is presented in several sections, as shown in Figure 9. Some of this data include: Dwelling characteristics, private charge points, or domestic energy storage system.



The screenshot shows the main screen of the Citizens Platform. At the top, there is a navigation bar with 'Profile' and 'Account' links, and a 'Driver Test' button. The main content is divided into three sections:

- PERSONAL DATA:** This section contains a form with the following fields:
 - Postal address: 28ης Οκτωβρίου 44, Αθήνα 106 82
 - Country*: Greece
 - Phone: 666555444
 - Identification number or passport: 75102762B
 Below the form, there is a checkbox labeled 'I'm agree with Terms and Conditions' which is checked, and a 'SAVE' button.
- PROFILE:** This section contains a form with the following fields:
 - Dwelling ID.*: (empty)
 - Rooms number*: 0
 - Tenants number*: 0
 Below the form, there is a note 'Check this value with your contact in TwinERGY' and a 'SAVE' button.
- PRIVATE CHARGE POINTS:** This section is currently empty.

Figure 9. Main screen in Citizens Platform

As this tool is used by citizens, who are the owners of their data and can leave the project when they want, the platform includes a button for removing account and related data in the screen "Account". When the user confirms the will to remove the account, the actions taken are:

- The tool removes profile data
- The tool removes the account in Keycloak
- The session ends
- The login page is shown
- User won't be able to log into any of TwinERGY applications.

3.2.2. Identity Management Platform

Being the core of the verification of end users in TwinERGY, the Identity Management Platform is a server based on Keycloak, which manages the access control based on Single Sign-On (SSO) for web apps and RESTful web services [2]. It means that the platform creates a trust relationship between an application (service provider), and an identity provider. This trust relationship is often based upon a certificate that is exchanged.

In a summary, Keycloak is a server to which other applications point to and are secured by it. Browser applications redirect a user's browser from the application to the Keycloak authentication server where they enter their credentials.

This redirection is important because users are completely isolated from applications and applications never see a user's credentials. Applications instead are given an identity token that is cryptographically signed. A token is a unique chain of text (an identifier) that represents, but does not contain, sensitive data, so external applications cannot obtain information of the data by only using this token.

By sensitive data we mean username, address, email, and so on. The token can also hold information about accessing permissions so that applications can make authorization decisions related to protected resources or actions. These tokens can also be used to make secure invocations on REST-based services. Depending on the configuration, the communication between Keycloak and the clients asking it for authentication services happens according to one of the two main supported SSO protocols: OpenID Connect [3] and SAML [4], being the first one the preferred method. Keycloak is based on the concept of realm, which manages a set of users, credentials, roles, and groups. A user belongs to and logs into a realm. Realms are isolated from one another and can only manage and authenticate the users that they control.

Identification and Access Management (IAM)

Identification and Access Management (IAM) is the term that ingroups the structures and processes within an organization that manage the access to IT resources by users. This term deals with network access rights, privileges, group memberships, etc, making easier the access for users to systems, applications, data, etc. In other words, IAM controls who has access to each resource and how to access it. These systems help ensure efficient, secure, and compliant operations [5]. Another typical task for an IAM system is the user account lifecycle. We can summarize the typical features of a IAM as it follows [6]:

- Shared access to the account: Creation of separate usernames and passwords for individual users or resources and delegate access.
- Granular permissions: Restrictions can be applied to requests. For example, allowing to download information, but deny the updating of information through the policies.

- Multifactor authentication (MFA): Users provide their username and password plus a randomly generated number used as an additional authentication factor.
- Identity Federation: If the user is already authenticated, such as through an external account, IAM can be made to trust that authentication method and then allow access based on it. This can also be used to allow users to maintain just one password for both on-premises and cloud environment work.
- Password policy: The IAM password policy allows to reset a password or rotate passwords remotely. Besides, this feature includes to set rules about how to pick a password or how many attempts a user may make to provide a password before being denied access.

In general, we can explain the operation of an IAM like showed in Figure 10. The IAM is composed by two phases [7]: configuration and operation. During the configuration, the information needed for the operation is stored: the user account (identity information) is created, together with the resources that are available to each user and the activities that are permitted for each of those resources.. During the operation phase the identification of the users and the control of accessing for a requested resource is given. Please, note that during the user account lifecycle, credentials or authorization information can be changed, so we cannot say that the configuration phase is executed only once when the account is created. On the other hand, the IAM manages two points: identities and access. Identities management corresponds to the user identity and credentials, and access management corresponds to the authorization processes.

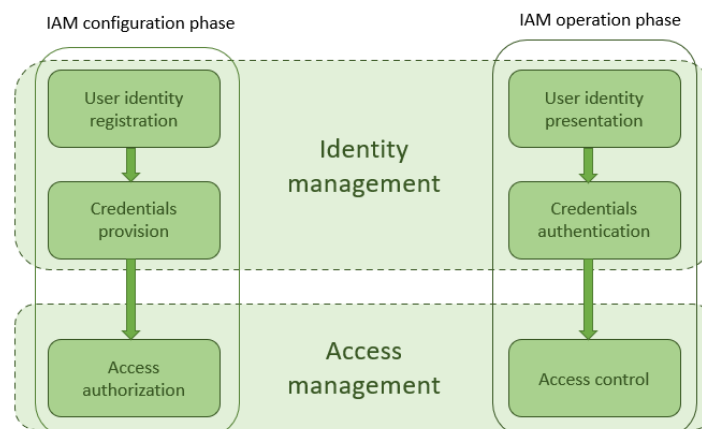


Figure 10. Operation of an IAM

Different standards of IAM have been created by different companies or consortiums, such as: Security Assertion Markup Language 2.0 (SAML 2.0) by OASIS [8], OAuth [9], OpenID [10], Central Authentication Service [11], etc. On the other hand, different

commercial and open source solutions based on these standards has been implemented. This is the case of Azure Active Directory by Microsoft [12], IBM Security [13], AWS Identity and Access Management by Amazon [14] are examples of commercial tools, whilst OpenIAM [15], Apache Syncope [16], Keycloak [17], or Central Authentication Service (CAS) [18] are example of open tools.

Due the high number of implementations we studied the differences among them in order to get the most appropriated for TwinERGY project, considering the features that are summarized in Table 8 and Table 9:

- Inclusion of Single Sign-On/Off
- Support for different protocols
- Extensible in the future, not a close environment
- Supported by a community, that is, that the software is currently maintained by a developers team, so any bug can be solved.
- Management of users' identities
- Management of password policies, such as format of the password, if they must be changed passed X period, and so on.
- Management of user privileges, and
- Compatibility with modern web languages.

Table 8. Comparison of IAMs open tools (1)

	Single Sign-on	Multi-protocol	Extensible	Supported by community	Manage identities
Keycloak	YES	YES	YES	YES	YES
Apache Syncope	YES	NO	NO	YES	YES
OpenIAM	YES	NO	NO	YES	YES
CAS	YES	YES	NO	YES	YES

Table 9. Comparison of IAMs open tools (2)

	Manage password policies	Manage user privileges	Compatible to web	Observations
Keycloak	YES	YES	YES	

Apache Syncope	YES	YES	NO	
OpenIAM	YES	YES	YES	Free project, but not open-source. It can be free used, but we cannot adapt code to our needs
CAS	YES	YES	YES	

As it can be seen in both tables, Keycloak seems to be the best option, considering that OpenIAM and Apache Syncope are not multiprotocol and CAS is not extensible.

3.2.2.1 Actions in IAM configuration phase

The most important actions that can be done during IAM configuration phase are:

1. Managing of realms.
2. Configuration of external storage (if it is needed).
3. Managing of realm's users. In addition to typical actions related to users, Keycloak can act as a third-party authorization server to manage application users, including users who self-register. In this last case, the login page will display a registration link so that user can create an account. Besides, it is possible to configure the personal data that is going to be stored per each user in the realm.
4. Managing of resources.
5. Managing of user sessions: This includes revocation policies, sessions timeouts and offline access (useful if an application needs to perform offline actions on behalf of the user when the user is not online, like regular backups).
6. Managing of permissions, roles and groups. That is, we can establish a permission for accessing to a resource or for executing an action, not only by the user itself, but depending on the role or the group where a user belongs to.
7. Configuration of authentication
8. Managing of the identity provider for a realm. On one hand, Keycloak allows to integrate external identity providers, like Google, Facebook or Twitter, or another cloud-based identity service. On the other hand, Keycloak can be configured as a third-party authorization server to manage application users, including users who self-register. If self-registration is enabled, the login page displays a registration link so that user can create an account.

9. Configuration of themes. Keycloak allows to configure the appearance of login and registration pages to make them looking similar to the services or applications that call Keycloak. That means that the user does not have the sensation of being visiting an external website of validation when he/she is inserting his/her credentials to access to the application

All these operations are executed in the Identity Management Platform.

3.2.2.2. Actions in IAM operation phase

During the operation, other actions are executed, as are indicated here:

Single Sign-On/Single Sing-Out

This feature allows users to:

1. Login the first time he/she access to one application/service and maintain that session for the rest of application/services.
2. Maintain the session and automatically renew it for all requests until a user express log out.
3. Logout once for all applications, so the rest of applications that uses the same IAM will have automatically logged out.

Access to protected resources

The client applications must validate permissions for accessing to protected resources via the IAM. For that, these tools are using:

1. Validation of credentials.
2. Generation of authorization codes.
3. Generation of accessing tokens and identity tokens. The *identity token* contains information about the user such as username, email, and other profile information. The *access token* is digitally signed by the realm and contains access information (like user role mappings) that the application can use to determine what resources the user is allowed to access on the application.
4. Redirection to resources. This means that after getting the access token to a protected resource, the IAM redirects the application to the URI where the token can be accessed.

3.2.3. Identity Management schema

For TwinERGY, the final architecture can be seen in Figure 11. From end-user's point of view, they are unaware of the identity. When an end-user wishes to participate in

TwinERGY, the action has to be undertaken to install the wanted applications into the respective device or access to the web page via a browser. In parallel, users self-register with Citizens platform and can start to use applications. Once the user has registered and/or login, the application will store the information about the session in the device memory.

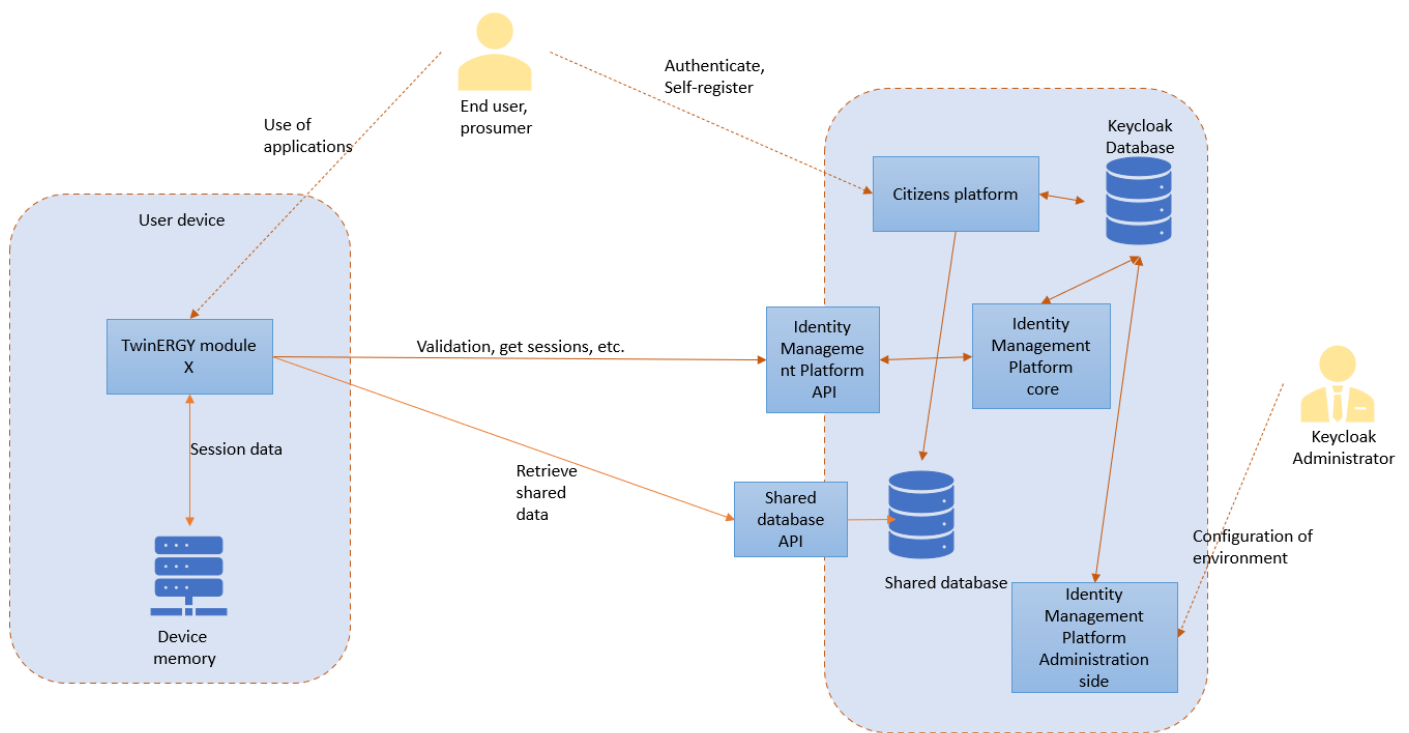


Figure 11. Workflow with Identity Management Platform

On the other hand, the administrator may access the server via a management page, where all points indicated in *As it can be seen in both tables*, Keycloak seems to be the best option, considering that OpenIAM and Apache Syncope are not multiprotocol and CAS is not extensible.

3.2.2.1 Actions in IAM configuration phase can be configured.

Finally, modules belonging to TwinERGY will communicate with the Identity Management Platform (Keycloak) via an API to make requests about validation of credentials, get sessions, get accessing tokens, and so on.

3.3. Interoperability platform

TwinERGY modules conform a distributed ecosystem where all can communicate with others. This means that each of them needs information generated by one or several modules as well as they generate information that can be needed by the rest of the modules. For instance, TwinEV module generates a charging curve for an EV charging session, and this information is used by other modules, such as DER management module.

Moreover, the status of a module can affect the rest. For instance, if a module fails during its execution, the modules that are waiting for inputs may hang. Hence, those waiting modules need to know this situation in order to follow a plan B: using previous data received from the module, making an estimation of what data would be received, or any other action. This point is important, to ensure that the information is interchanged among them, since it is related to the tolerance on failures of the complete ecosystem. Besides, the modules need to receive the information immediately. It is not desirable to receive information with a great delay or corresponding to a previous time period. Therefore, all modules need to work with the latest data, corresponding to a real time schema.

Such a situation implies that modules need a common platform of communication, where they can receive and send messages about their calculations and their statuses in a real time environment. Traditional solutions such as checking the common database for new/modified information and reading the new/modified data can lead to a potential bottleneck, and must be avoided. Besides, a common database do not solve the problem of informing about the status of a module.

On the other hand, each module has its own characteristics, it is implemented in a different language, and it is deployed in different premises. This implies that the communication channel has to be independent from the module characteristics. All these topics or issues have led us to look for an independent and unified framework that also guarantees security.

During the task T7.1, we examined different standards for both messages' formats and communication protocols, being NATS the most appropriated technology for TwinERGY.

NATS [19] is a messaging protocol that enables client applications to communicate with messaging middleware brokers. NATS works in the same way to a mailbox, so a sender sends a message with an "address" and the listeners that are waiting form messages in

that address to receive the message (several listeners can be waiting for messages related to the same address). Therefore, publishers send messages to the NATS server and this server distributes messages among listeners according to that address (here called subject). The server never stores messages, only distributes them.

As commented, the subject is similar to a mailbox address but with some variations. Listeners connected to the server indicate what subjects or set of subjects are interested in, so the server knows which receivers are waiting for each subject. In *Figure 12* we depict an example where a module is sending information about weather forecasting in different cities and there are different modules that want to receive only weather messages related to a city:

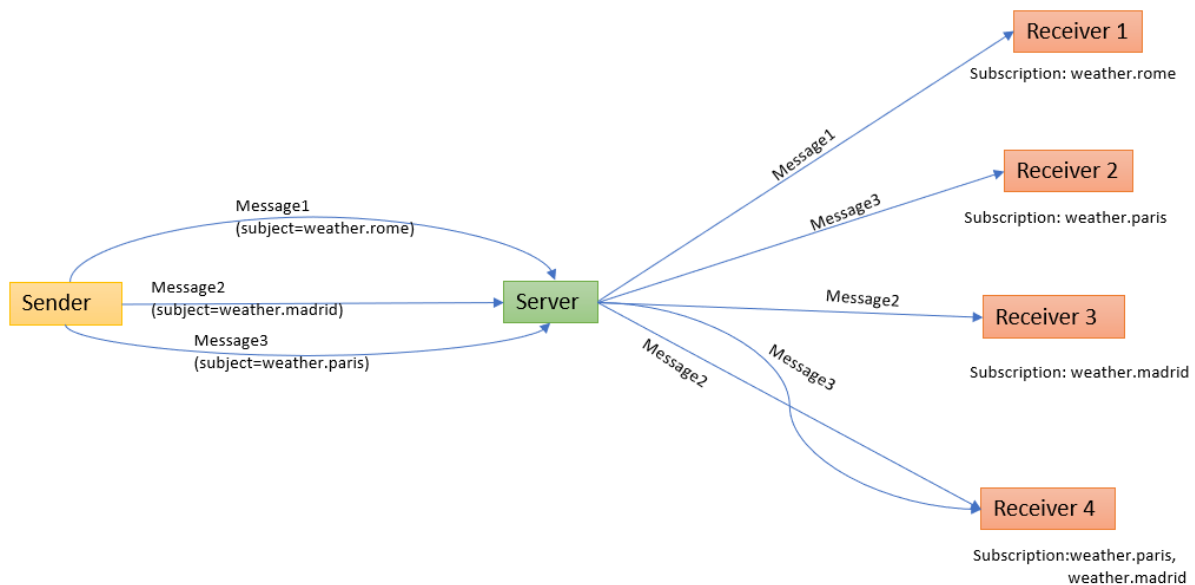


Figure 12: Example of messages sending in NATS

As it can be seen in *Figure 12*, a module can be subscribed to more than one subject. Since a module may need messages related to a several types of subjects, NATS allows to use shortcuts in order to make easier this definition. Continuing with the example of sending weather in cities around the world, if a module wants to receive weather information from all capitals in Europe, the sender can send messages with subjects like “weather.europe.[city_name]”, “weather.asia.[city_name]”, etc. and the receiver can subscribe to “weather.europe.*”. Of course, modules that only wish to get information about a city in Europe still can subscribe to “weather.europe.[city_name]”. In addition to this schema of subscriptions, NATS allows to send a request and waits for a response. This is very helpful in case that a module wishes to request information directly to another

one. In this case, the communication can be synchronous (sender locked until receiving the response or timeout) or asynchronous (sender continues working while waiting for the response).

Regarding security, NATS uses modern Transport Layer Security (TLS) semantics to encrypt clients, route, and monitoring connections. In addition, NATS can be configured to force client authentication in different ways:

- Connection by user/password.
- Connection by token.
- Connection by NKEY: Server has a list of known public keys and clients have to respond to the challenge by signing it with its private key.
- Connection by credentials file.

3.3.1. Messages

For the purposes of TwinERGY, we have defined the data structure in messages and the subject to identify the message type and the sender.

Taking advantage of rules for composing subjects, we structure the subjects in this way:

`{MOD}.{TYPE}.{SUBTYPE}`

Where:

- {MOD} is the module identifier,
- {TYPE} is the message type, and
- {SUBTYPE} is the message subtype.

For example, subject “M3.notify.error” indicates that Module 3 is notifying an error, and “M4.synchronization” indicates that Module 4 needs to synchronize with another module.

Regarding the format of the messages, NATS allows to send messages in plain text, or structured in JavaScript Object Notation (JSON format), which is incorporated in the most of modern languages. JSON is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute–value pairs. After some deliberations among modules responsible, it was agreed to use JSON format in messages and, depending on the message type, have a different structure for the message content.

The complete list of messages can be found in Annex 1. Messages interchanged through Interoperability Platform.

3.3.2. NATS console

Complementary to the Interoperability Platform, this web application has been developed by ETRA, being accessible in this URL <https://twinergy-console.etra-id.com>. It consists of a tool related to monitor the platform and publish/subscribe messages. Since this is a monitoring tool for the Interoperability Platform, only partners can work with it. The application has been created with the aim of supporting the development of modules in WP7 in different ways: a developer can send easily messages to be read by his/her module, check if a message has been sent from his/her modules, or test response time from his/her modules. The application is composed by these screens:

- **Dashboard:** The main page, where it is shown statistics about number of connections, number of subscriptions, memory used, etc.
-

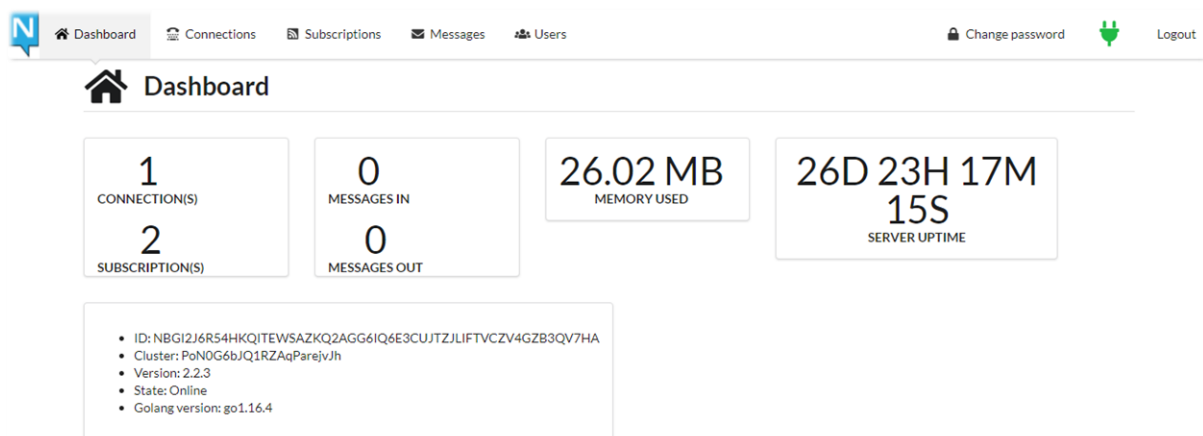
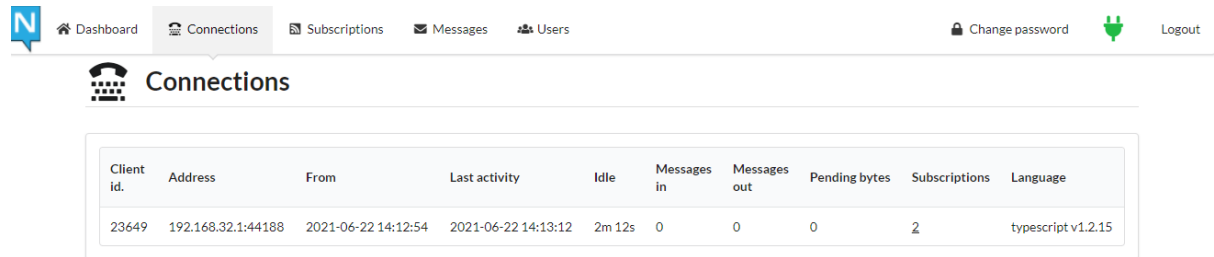


Figure 13. Dashboard view in NATS console

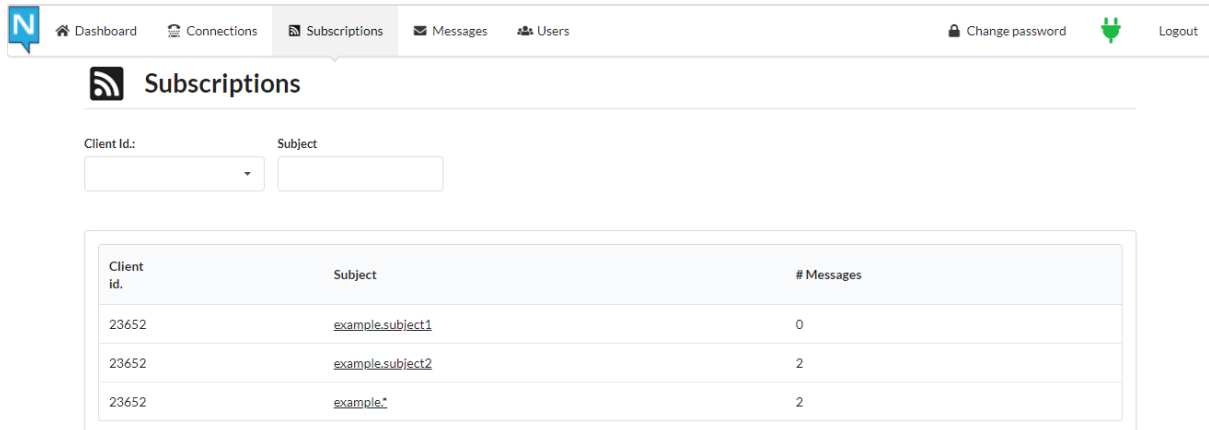
- **Connections:** List of applications (here called clients) connected to NATS server. For each client connected, it shows its address, its activity and other information. Besides, if user clicks over the number of subscriptions of a client, the application goes to the view Subscriptions with the filter client set to the client.



Client id.	Address	From	Last activity	Idle	Messages in	Messages out	Pending bytes	Subscriptions	Language
23649	192.168.32.1:44188	2021-06-22 14:12:54	2021-06-22 14:13:12	2m 12s	0	0	0	2	typescript v1.2.15

Figure 14. Connection's view

- **Subscriptions:** List of active subscriptions, allowing filter by a client and/or a subject. Per each subject, it informs about the client identifier, and the number of messages received with that subject. Besides, when the user clicks on one subject in the table, the application to 'Messages' and start to listen that subject.



Client id.	Subject	# Messages
23652	example.subject1	0
23652	example.subject2	2
23652	example.*	2

Figure 15. Subscriptions view in NATS console

- **Messages:** This screen allows to send messages with a subject, as well as listen to messages related to a subject.

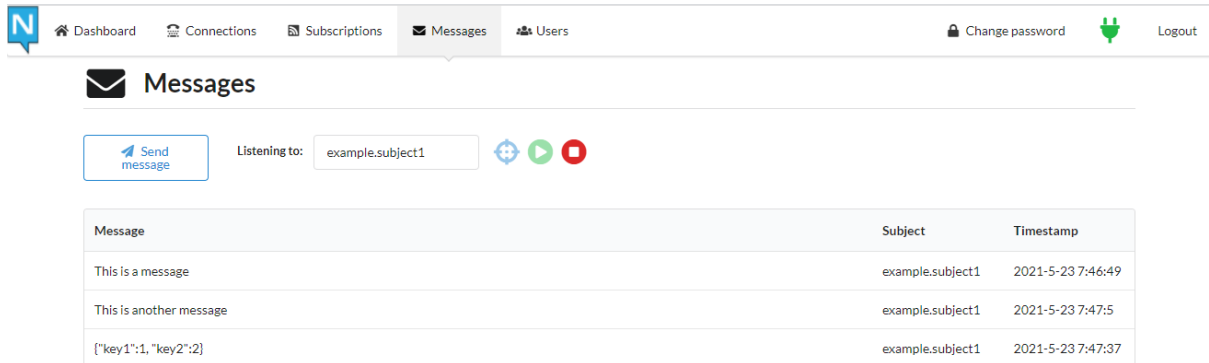


Figure 16. Message's view in NATS console

The application starts to listen when the user inserts the subject in the field "Subject to listen" and click on green icon "play". On the other side, the subscription ends when the user clicks on red icon "stop". Alternatively, the application allows to subscribe to one existing subject, instead of filling that field:

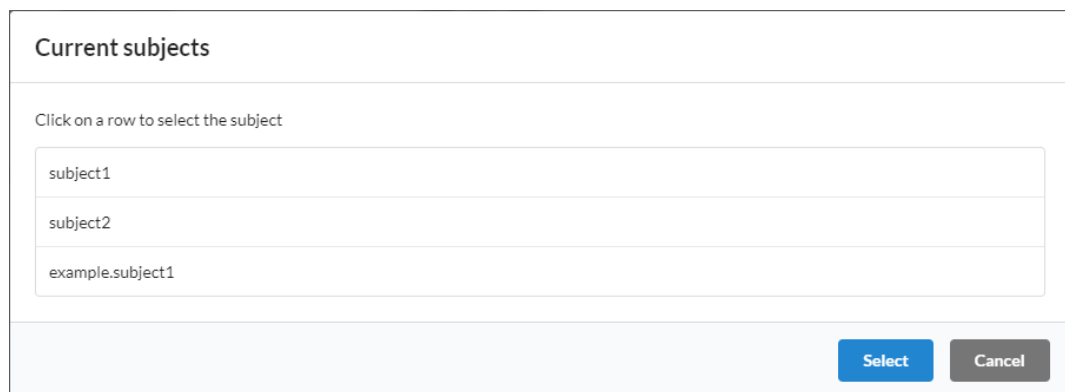


Figure 17. Window to select one existing subject

Finally, the user can send messages with a subject, with the button "Send message". One window to complete the information will appear.



Figure 18. Window to send messages in interoperability platform

3.4. Core Data Management Platform

Developed within WP5 by Suite5, the Core Data Management Platform (CDMP) is an "open", modular and interoperable platform, which enables data collection and management communication, offering the next services:

- Data Collection Service
- Data Security Service
- Data Storage Service
- Platform Management Service

It can be accessed at <https://twinergy.s5labs.eu/>

A schema for this component is presented in the *Figure 19*, where we represent how pilots communicate with the Core Data Management Platform, as well as the connection of the TwinERGY modules and Digital Twins via open APIs to the platform, in order to retrieve the necessary pilot's data for further processing.

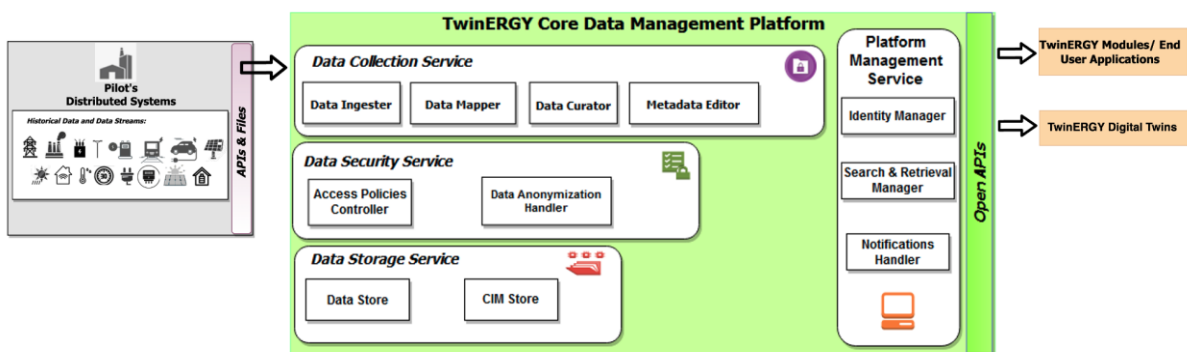


Figure 19: TwinERGY Core Data Management Platform Conceptual Architecture

The internal module that communicates with assets installed in pilot site is the **Data Ingestor**, which introduces proper procedures and methods for ingesting data. It offers the following functions:

- Data ingestion process configuration, that depicts the way that the data will be ingested into the TwinERGY Core Data Management Platform (file uploading, API).
- Ingestion of data from files, that allows data to be retrieved from files in popular file formats (e.g., csv, JSON).
- Ingestion of data via APIs, that allows data to be retrieved from both pilot system APIs and Open Data APIs (e.g., weather data, other local sources).
- Reliable and safe data upload, that enables data to be uploaded to the TwinERGY Core Data Management Platform using reliable and secure processes.

Also, the **Search and Retrieval Manager** allows platform users to search and find data that may be proven valuable, as well as to define the necessary steps that allow the secure retrieval of that data via Open APIs. This component offers the following functions:

- Clear and comprehensive view of the available assets
- Appropriate filters and free text search for the identification of assets that meet the user's criteria
- Intuitive steps by step guide to prepare the API for data retrieval.
- Single point of entry for apps (both TwinERGY modules and Digital Twins) to retrieve data from the CDMP that they're allowed to access

3.5. Transactive Energy Blockchain Network

By definition, Blockchain [20] is a shared, immutable database that is shared among the nodes of a computer network, so the information is stored in a distributed way. That means that the main goal of Blockchain networks is to allow digital information to be recorded and distributed, but not edited, unlike traditional databases, where data can be created, edited and removed. In this way, a blockchain is the foundation for immutable records (ledgers) of transactions that cannot be altered, deleted, or destroyed. Therefore, blockchains are also known as a distributed ledger technology (DLT).

Another main difference between a traditional database and a blockchain is how the data is structured. While traditional databases store information in structured schemes called tables, a blockchain collects information together in groups, known as blocks, that hold sets of information. Blocks have certain storage capacities and, when filled, are closed and linked to the previously filled block, forming a chain of data known as the blockchain. All new information that follows that freshly added block is compiled into a newly formed block that will then also be added to the chain once filled.

In TwinERGY Transactive Energy Platform (TEM) developed by WEC, blockchains cannot interact with data and systems outside their native Blockchain environment. Data outside the Blockchain is considered “off-chain” and data already stored on the Blockchain is considered “on-chain”. Purposely isolated from external data and systems, the Blockchain obtains its most valuable properties. Interoperating with of-chain systems requires additional infrastructure known as “oracle”. Oracles bridge the gap between the two environments.

Input oracles

An input oracle fetches data from the off-chain and delivers it onto a Blockchain network for smart contract consumption. The Transactive Energy Platform utilises different types of input oracles:

- Smart meter oracle: Delivers the meter readings of individual Smart meters to the Blockchain network
- LEM Price oracle: Delivers the calculated LEM price to the Blockchain network
- TEM oracle: Facilitates communication between the prosumers and the Blockchain network for trades, transactions, and rewards.

Output oracles

Opposite of input oracles are “output oracles,” these allow smart contracts to send commands to off-chain systems that trigger them to execute certain actions. The TEM utilises different types of output Oracles:

- Trade oracle: Notifies the TEM about the availability of new trade orders
- Transaction oracle: Instruct the IoT network to charge or discharge batteries when certain conditions are met
- DER oracle: Notifies the DER module to issue vouchers when prosumers have exchanged their reward tokens.

Blockchain consensus

The Transactive Energy Platform utilises a Proof of Authority (PoA) consensus mechanism. Using this consensus mechanism, the Transactive Energy Platform can keep a tighter grip on both coin supply and coin trading and can ensure that all transactions occurring on the blockchain are genuine. In PoA the identities of validator nodes are publicly known, and thus it would be extremely detrimental to the validator to engage in

fraudulent or malicious behaviour. They could be easily found, lose their reputation alongside the loss of validator status.

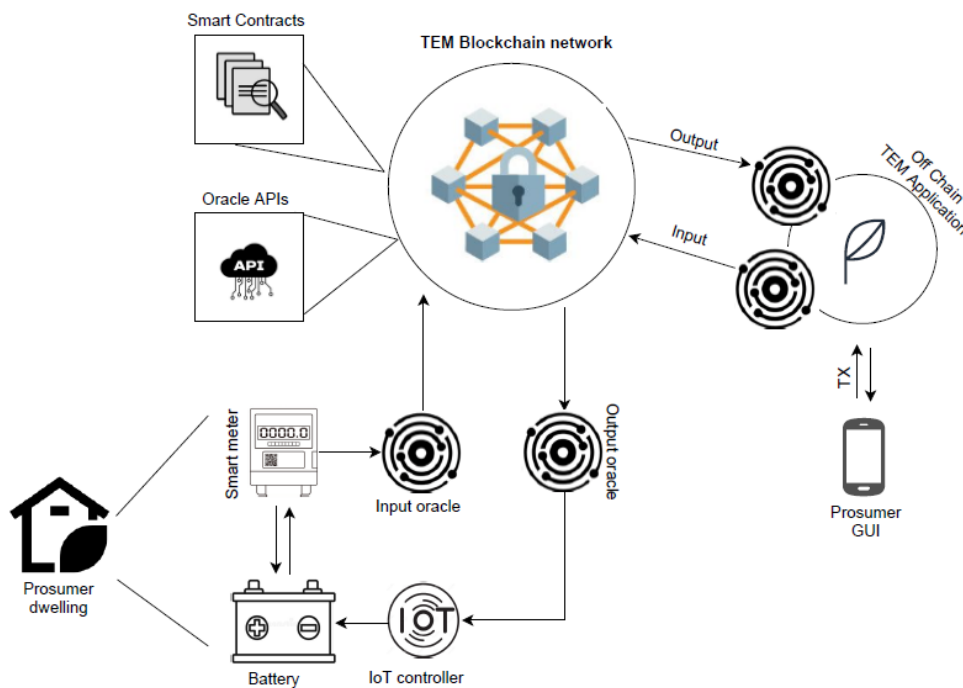


Figure 20. Workflow in Transactive Energy Blockchain network

3.6. Digital Twin & iSCAN

A Digital Twin [21] is a virtual model designed to accurately reflect a physical object. That virtual representation is updated from real-time data, and uses simulation, machine learning and reasoning to help decision-making. For that, the object being studied is outfitted with various sensors related to vital areas of functionality, so they produce the real-time data that is then relayed to a processing system and applied to the digital copy.

Once informed with such data, the virtual model can be used to run simulations, study performance issues and generate possible improvements, all with the goal of generating valuable insights — which can then be applied back to the original physical object.

In TwinERGY, the Digital Twin is deployed to estimate the available flexibility on each combination of load and demand response service. In addition to the flexibility potential, other features are calculated and matched to the required metrics of the demand response services.

We have to remark that three types of digital twins have been developed, one for consumer behaviour, another one for building behaviour and another one for community behaviour. The Consumer DT is an evolving digital profile of the past and current behaviour of the consumer as far as her/his energy consumption is concerned. The aggregation of individual Consumer DTs composes the Building DT and the aggregation of the individual Building DTs on a district modelling tool along with the energy and water networks and mobility infrastructure and assets will constitute the Community DTs. The registered behaviour information is completed through metrics of consumption recordings, personal physiological data, localization info, equipment properties, equipment operating procedures, indoor and outdoor environmental conditions, etc.

Regarding iSCAN, it is a cloud-based data management and interrogation platform that can import time-series data from multiple sources to a single platform, e.g., from the Building Management System (BMS), smart meters, IoT sensors, etc, and it is used in TwinERGY as a gateway between DT and the rest of software components. With iSCAN the user can interrogate the data manually or automatically using Machine Learning and AI. iSCAN has the capabilities to identify how the building is performing in real-time with respect to energy use, indoor environmental quality and running costs. Insights can be generated, and analysis can then be performed on the data, to identify quick wins such as faults or easy to implement energy saving measures. Machine Learning and AI can be used to optimise performance on a day-to-day basis.

Time series data in iSCAN can be exported to DT ecosystem to be exploited by all technologies within the ecosystem. When this is carried out, those DTs become into a hybrid Digital Twin based on both physics data and metered data to create a higher level of accuracy and an ability to carry out more detailed analysis with respect to optimisation and intervention planning. For example, iSCAN can be imported to the Community DT to create a hybrid community model based on real energy end use data and not simulated assumptions.

The DT and the iSCAN are totally described in deliverable *D6.4. Digital Twin Interconnected Platform v2.0*

4. Conclusions and next steps

This deliverable D8.3 is the description of the work done during task T8.2. In it, we have described the integrated final architecture that has been deployed and tested in a lab environment. After describing the implementation for each pilot site, we have described the whole solution planned for TwinERGY, that is, the final implementation of each component and its location in the project.

As it is presented in the deliverable D8.2 (*TwinERGY Pre-trial validation testing scenarios and results*), the testing on the lab environment has been successful, so it proves that this solution is strong enough to be deployed in a real environment. Therefore, the natural next steps will be to deploy the final architecture, presented in this deliverable D8.3, into a real environment and, after a new test in that environment, allow to real users to utilize all services that TwinERGY offer. This will be done during the next months along with task T8.3 and exposed in deliverable D8.4.

References

- [1] T. project, "TwinERGY deliverables," [Online]. Available: <https://www.twenergy.eu/deliverables>.
- [2] Keycloak, "Server Administration Guide," [Online]. Available: https://www.keycloak.org/docs/latest/server_admin.
- [3] Auth0, "The OpenID Connect Handbook," [Online]. Available: https://auth0.com/resources/ebooks/the-openid-connect-handbook?utm_content=oidc-handbook-1&utm_source=google&utm_campaign=emea_spain_esp_all_ciam-dev_dg-ao_auth0_search_google_text_kw_utm2&utm_medium=cpc&utm_term=openid%20connect-c&utm_id=aNK4z0000004HhnG.
- [4] Onelogin by One identity, "SAML Explained," [Online]. Available: <https://www.onelogin.com/learn/saml#:~:text=SAML%20is%20an%20acronym%20used,one%20set%20of%20login%20credentials..> [Accessed 2022].
- [5] Tools 4 Ever, "What is Identity & Access Management?," [Online]. Available: <https://www.tools4ever.com/glossary/what-is-identity-and-access-management/>.
- [6] Simplilearn, "AWS IAM: Working, Components, and Features Explained," August 2021. [Online]. Available: <https://www.simplilearn.com/tutorials/aws-tutorial/aws-iam>.
- [7] Wikipedia, "Identity management," [Online]. Available: https://en.wikipedia.org/wiki/Identity_management.
- [8] OASIS, "Security Assertion Markup Language (SAML) V2.0 Technical Overview," [Online]. Available: <http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0.html>.
- [9] IETF OAuth Working Group., "OAuth 2.0," [Online]. Available: <https://oauth.net/2/>.
- [10] OpenID, "OpenID," [Online]. Available: <http://openid.net/>.
- [11] Apereo, Inc, "CAS Protocol 3.0 Specification," [Online]. Available: <https://apereo.github.io/cas/6.2.x/protocol/CAS-Protocol-Specification.html>.

- [12] Microsoft , “Azure Active Directory (Azure AD),” [Online]. Available: https://azure.microsoft.com/en-us/products/active-directory/?ef_id=Cj0KCQiAwjWdBhCYARIsAjc4idASl0DQRa38eWjLI6oba0cJD7eOGKWlu0MWBedyjXzL6EU0fOMiCMaAojdEALw_wcB%3AG%3As&OCID=AIDcmm68ejnsa0_SEM_Cj0KCQiAwjWdBhCYARIsAjc4idASl0DQRa38eWjLI6oba0cJD7eOGKWlu0MWBBe. [Accessed 2022].
- [13] IBM, “Enterprise security solutions,” [Online]. Available: <https://www.ibm.com/security>. [Accessed 2022].
- [14] Amazon, “AWS Identity and Access Management (IAM),” [Online]. Available: <https://aws.amazon.com/iam/>. [Accessed 2022].
- [15] OpenIAM, “Modern platform to manage all identities,” [Online]. Available: <https://www.openiam.com/>.
- [16] Apache, “Apache Syncope,” [Online]. Available: <https://syncope.apache.org/>. [Accessed 2022].
- [17] Keycloak, “Open Source Identity and Access Management,” [Online]. Available: <https://www.keycloak.org/>.
- [18] CAS, “ Apereo CAS - Identity & Single Sign-On,” [Online]. Available: <https://apereo.github.io/cas/6.6.x/index.html>. [Accessed 2022].
- [19] NATS, “NATS,” [Online]. Available: <https://nats.io/>. [Accessed May 2021].
- [20] A. Hayes, “Blockchain facts: What is it, how it works and how can be used,” Investopedia, September 2022. [Online]. Available: <https://www.investopedia.com/terms/b/blockchain.asp>.
- [21] IBM, “What is a digital twin?,” 2022. [Online]. Available: <https://www.ibm.com/topics/what-is-a-digital-twin>.
- [22] S. Ontology, “SAREF ETSI,” [Online]. Available: <https://saref.etsi.org/>. [Accessed May 2021].

- [23] SAREF, "SAREF for Energy," [Online]. Available: <https://saref.etsi.org/saref4ener/>. [Accessed May 2021].
- [24] SAREF, "SAREF for Buildings," [Online]. Available: <https://saref.etsi.org/saref4bldg/>. [Accessed May 2021].
- [25] O. ADR, "OPEN ADR," [Online]. Available: <https://www.openadr.org/specification>. [Accessed May 2021].

Annex 1. Messages interchanged through Interoperability Platform

Source	Destination	Message	Subject	Payload
M1	ALL	M1 takes a new measurement from the consumer's wrist about the indoor environmental conditions	M1.add.newMeasurement	<pre>{ newMeasurement: { "wearableId":String, "timestamp": "YYYY-MM-DD HH:MM:SS", "measurements": ArrayOf({ "indoor_temperature": float, "indoor_humidity": float, "indoor_co2": float, "indoor_tvoc": float, "user_activity": float }), }</pre>
M1	ALL	M1 user updates the clothing insulation level	M1.add.newClothingInsulation	<pre>{ newClothingInsulation: { "userId":String, "timestamp": "YYYY-MM-DD HH:MM:SS", "clothingInsulation": ArrayOf({ "season": Integer, "clo_value": float, }), }</pre>
M1	ALL	M1 user updates preferences	M1.add.newPreferences	<pre>newPreferences: { "userId":String, "timestamp": "YYYY-MM-DD HH:MM:SS", "m1_preferences": ArrayOf({ "max_temp": Integer, "min_temp": Integer, "thermal_importance_level": Integer "wellbeing_importance_level": Integer }), "electric_de_vices": ArrayOf({ "freezer_importance_level": Integer, "heater_importance_level": Integer, }), "energy_preferences": ArrayOf({ "energy_flexibility_importance_level": Integer, "financial_cost_importance_level": Integer, "financial_cost_up": Integer, }) }</pre>

				<pre> "financial_cost_down": Integer }) } </pre>
M2	All	User appliances flexibility	M2.applianceFlexibility.Building.{PILOT_NAME}	<pre> { "applianceID": "string", "applianceName": "string", "buildingName": "string", "pilotName": "string", "startDate": "YYYY-MM-DD", "endDate": "YYYY-MM-DD", "timeZoneOffset": float, "samplePeriod": int, "profile": ArrayOf(floats), } </pre>
M2	All	Consumer demand forecast	M2.consumerDemandForecast.Building.{PILOT_NAME}	<pre> { "buildingName": "string", "pilotName": "string", "startDate": "YYYY-MM-DD", "endDate": "YYYY-MM-DD", "timeZoneOffset": float, "samplePeriod": int, "profile": ArrayOf(floats), } </pre>
M3	All	Neighbourhood flexibility profile	M2.neighbourhoodFlexibility.{PILOT_NAME}	<pre> { "pilotName": "string", "startDate": "YYYY-MM-DD", "endDate": "YYYY-MM-DD", "timeZoneOffset": float, "samplePeriod": int, "profile": ArrayOf(floats), } </pre>
M3	All	Neighbourhood demand forecast	M3.neighbourhoodDemandForecast.{PILOT_NAME}	<pre> { "pilotName": "string", </pre>

				<pre> "startDate": "YYYY-MM-DD", "endDate": "YYYY-MM-DD", "timeZoneOffset": float, "samplePeriod": int, "profile": ArrayOf(floats), } </pre>
M5	M2	Renewable energy generation profile	M5.renewableGeneration.Building.{PILOT_NAME}	<pre> { "generationID": "string", "generationName": "string", "buildingName": "string", "pilotName": "string", "startDate": "YYYY-MM-DD", "endDate": "YYYY-MM-DD", "timeZoneOffset": float, "samplePeriod": int, "profile": ArrayOf(floats) } </pre>
M4	M2	Appliance electric profile	M4.applianceProfile.Building.{PILOT_NAME}	<pre> { "applianceID": "string", "applianceName": "string", "pilotName": "string", "startDate": "YYYY-MM-DD", "endDate": "YYYY-MM-DD", "timeZoneOffset": float, "samplePeriod": int, "profile": ArrayOf(floats) } </pre>
M4	M3			<pre> { "user": { "uuid": "string" }, "building": { </pre>

				<pre> "uuid": "string", "altitude": float, "longitude": float, "latitude": float, "buildingSpaces": ArrayOf([{ "uuid": "string", "name": "string", "type": "string", "size": integer, "sizeUnitOfMeasure": "string", "physicalObjects": Array, "devices": Array, "buildingSpaces": Array }]) }, "bills": ArrayOf({ "billNum": long, "energyProvider": "string", "offerName": "string", "dateFrom": "YYYY-MM-DD", "dateTo": "YYYY-MM-DD", "totalCharge": integer, "consumedEnergy": integer, "pickPower": integer, </pre>
--	--	--	--	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

				<pre> "readingType": "string", "contractType": "string", "raiFee": integer, "other": "string" }) } </pre>
M5	All	DER Module sends Incentive Curve (24h forecast)	M5.data.icc.{PILOT_NAME}	<pre> { "emission": float, "power_with_slp": float, "pv_power": float, "real_time": datetime object, "res_power": float, "timestamp": string, "wind_power": float }, one dataset is for 15 minutes, the whole one for 24 hours </pre>
M5	All	DER Module sends DR signal (for the current quarter hour)	M5.data.drsignal.{PILOT_NAME}	<pre> { "dr_signal": float } </pre>
M5	All	DER module sends Energy Light signal (for the current quarter hour)	M5.data.energylight.{PILOT_NAME}	<pre> { "energy_light": string } </pre>
M7	All	TwinEV informs about new charge session	M7.notification.sessionStarted.{PILOT_NAME}	<pre> { "stationId":String, "chargePoint":String, "start": datetime("YYYY-MM-DDThh:mm:ss.sssZ"), "end": datetime("YYYY-MM-DDThh:mm:ss.sssZ"), "chargingCurve": ArrayOf({ "from": datetime("YYYY-MM-DDThh:mm:ss.sssZ") "to": datetime("YYYY-MM-DDThh:mm:ss.sssZ") }) } </pre>

				<pre> "power": float, }}, } </pre>
M7	All	TwinEV informs about end of chage session	M7.notification.sessionEnd.{PILOT_NAME}	<pre> { "stationId": String, "chargePoint": String, "timestamp": datetime("YYYY-MM-DDThh:mm:ss.sssZ"), "totalCost": float, "totalEnergySupplied": float "totalEnergyToGrid": float } </pre>
M7	All	TwinEV informs about new restrictions in charge points supply	M7.notification.changeSupply.{PILOT_NAME}	<pre> { points: ArrayOf({chargeStation:string, point:string}), maxPower: float, maxInjection: float, start: datetime("YYYY-MM-DDThh:mm:ss.sssZ"), end: datetime("YYYY-MM-DDThh:mm:ss.sssZ"), } </pre>
M8	M9	Provide update of TwinERGY coins won based on social competition trigger in M8; coins are sent to TEM to be utilized in exchange of available goods	M8.update.TwinCoins	<pre> { "content": { "user_id": "String", "timestamp": datetime("YYYY-MM-DDThh:mm:ss.sssZ") "TwinCoins": "string", } } </pre>
M9	M8	Provide update on balance and TwinERGY coins used	M9.update.TEMbalance	<pre> "content": { "user_id": "String", "timestamp": datetime("YYYY-MM-DDThh:mm:ss.sssZ"), "balanceEu": "string", "TwinCoins_used": "50", } </pre>

M9	M8	Provide status of available goods or services		<pre>{ "content": { "pilotSite": "String", "timestamp": datetime("YYYY-MM-DDThh:mm:ss.sssZ"), "AvailableGoods": "Yes/No", "NrOfGoods": "integer", "AvailableService": "Yes/No", "NrOfService": "integer" } }</pre>
----	----	-----------------------------------------------	--	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------