



D2.15 IANOS System Architecture

Authors:

P. Tzallas, N. Foutakis, N. Bezas, I. Moschos, D. Ioannidis, D. Stefanitsis
[CERTH]

L. Pons Bayarri [ETRA]

D. Ziu [ENG]

E. Werkman [TNO]

C. Patrao [CWD]

L. Meijer [NEROA]

R. Lopes [UNINOVA]

A. Bachoumis [UBE]



H2020-LC-SC3-2018-2019-2020 / H2020-LC-SC3-2020-EC-ES-SCC

EUROPEAN COMMISSION

Innovation and Networks Executive Agency

Grant agreement no. 957810

PROJECT CONTRACTUAL DETAILS

<i>Project title</i>	IntegrAted SolutioNs for the DecarbOnization and Smartification of Islands
<i>Project acronym</i>	IANOS
<i>Grant agreement no.</i>	957810
<i>Project start date</i>	01-10-2020
<i>Project end date</i>	30-09-2024
<i>Duration</i>	48 months
<i>Project Coordinator</i>	João Gonalo Maciel (EDP) - JoaoGoncalo.Maciel@edp.com

DOCUMENT DETAILS

<i>Deliverable No</i>	D2.15
<i>Dissemination level</i>	Public
<i>Work Package</i>	WP2 Requirements Engineering & Decarbonization Roadmapping
<i>Task</i>	T2.5 – System Architecture
<i>Due date</i>	30 th November 2022
<i>Actual submission date</i>	8 th December 2022
<i>Lead beneficiary</i>	CERTH

Version History

v	Date	Beneficiary	Changes
0.1	31/10/2022	EDP	Contributions and proposed changes
0.2	7/11/2022	ETRA	Contributions and updates
0.3	11/11/2022	TNO	Contributions and updates
0.4	24/11/2022	CERTH	Changes regarding Functional and Non-Functional Requirements
0.9	28/11/2022	CERTH	Consolidated version for review
1.0	5/12/2022	CERTH	Final version



Executive Summary

This document is one of the deliverables of the IANOS project, in the context of Work Package 2 (WP2) “Requirements Engineering & Decarbonization Roadmapping”. Specifically, it presents the third version of IANOS project’s system architecture and the technical specifications of the project’s ICT components. By providing a holistic view of the overall architecture of the system, the goal of the deliverable is to define the system’s functionalities, describe the several components of the system and the data flows between them in the previously defined scenarios in D2.3 and present the requirements of the system.

As a first step, the model adopted to describe the architecture of the IANOS system is defined, that is the 4+1 architectural view model, separating the overall architecture in different views. The Conceptual Architecture and the Logical view of the system are presented, in which the different layers of IANOS system are identified. These layers correspond to the Business Layer, the Decision-Making Layer, the User Interface Layer, the Communication Layer and the Physical Layer. The components of each layer are introduced, and their functionalities are presented.

Subsequently, for each component of the system, the different functional and non-functional requirements are presented, as well as the priority of each requirement. The functional requirements provide descriptions on what the system must and must not do, thus providing the necessary features that describe the intended behaviour of the system.

With the use of Unified Modelling Language functionality diagrams, the Development View of the system provides information about the use of the existing software, considering the several legal and accessibility issues, as well as the technical requirements and the dependencies of each component.

The next step of the architecture’s definition is the presentation of the Dynamic View of the system that provides a detailed workflow and data exchange between the different components and actors of the system for each defined Use Case in D2.3. UML sequence diagrams are used for a detailed representation of the dynamic view of the system.

Lastly, the deployment of software into hardware (physical aspects of the system) is presented in the Deployment View of the system, with the use of UML deployment view diagrams, as well as detailed data interfaces and the system’s general deployment state.

Table of contents

EXECUTIVE SUMMARY	3
TABLE OF CONTENTS.....	4
LIST OF FIGURES	8
LIST OF TABLES	10
1 INTRODUCTION	12
1.1 OBJECTIVES AND SCOPE OF THE DELIVERABLE.....	12
1.2 STRUCTURE OF THE DELIVERABLE	12
1.3 RELATION TO OTHER TASKS AND DELIVERABLES	13
1.4 IANOS SYSTEM ARCHITECTURE SECOND VERSION CHANGELOG	13
2 METHODOLOGY.....	15
2.1 DESIGN PRINCIPLES	15
2.2 ARCHITECTURE VIEW MODEL	15
2.3 IANOS ACTORS.....	17
2.4 RELATED DATA MODELS, PROTOCOLS AND STANDARDS	20
2.4.1 Open Automated Demand Response (OpenADR).....	20
2.4.2 Universal Smart Energy Framework (USEF).....	21
2.4.3 Energy Flexibility Interface (EFI) and EN 50491-12-2 (S2).....	22
2.4.4 Smart Appliances REference Ontology (SAREF).....	24
2.4.5 Message Queuing Telemetry Transport (MQTT).....	25
2.4.6 Advanced Message Queuing Protocol (AMQP).....	26
2.4.7 IEC 61850 Standard.....	27
2.4.8 CHAdemo V2X protocol.....	29
2.5 UML DIAGRAMS	30
3 CONCEPTUAL ARCHITECTURE – LOGICAL VIEW	32
3.1 BUSINESS LAYER.....	33
3.1.1 VERIFY-D platform tool (LCA and LCC module).....	33
3.1.2 CrowdEquity Platform.....	34
3.1.3 System Modeller.....	35
3.1.4 Grid-Oriented Optimizer - INTEMA.grid.....	38
3.1.5 CBA component.....	39
3.2 DECISION MAKING LAYER.....	42
3.2.1 Aggregation and Intelligent Segmentation.....	42
3.2.2 Forecasting Engine.....	42
3.2.3 Centralized Dispatcher.....	43
3.2.4 DLT-based Transactive Platform	46



3.2.5	GOPACS.....	48
3.3	USER INTERFACE LAYER	48
3.3.1	Virtual Energy Console.....	49
3.4	COMMUNICATION LAYER.....	49
3.4.1	IANOS Secured Enterprise Service Bus.....	49
3.4.2	dEF-Pi.....	52
3.4.3	Back-end Agent	53
3.5	PHYSICAL LAYER.....	53
3.5.1	Non-intrusive characterization and use of energy flexibility in water heating systems	53
4	SYSTEM REQUIREMENTS	55
4.1	FUNCTIONAL REQUIREMENTS	55
4.2	NON-FUNCTIONAL REQUIREMENTS	63
5	DEVELOPMENT VIEW.....	74
5.1	BUSINESS LAYER	74
5.1.1	VERIFY-D platform tool (LCA and LCC module).....	74
5.1.2	CrowdEquity Component.....	74
5.1.3	System Modeller	75
5.1.4	Grid-Oriented Optimizer - INTEMA.grid	76
5.1.5	CBA component.....	77
5.2	DECISION MAKING LAYER.....	78
5.2.1	Aggregation and Intelligent Segmentation.....	78
5.2.2	Forecasting Engine.....	79
5.2.3	Centralized Dispatcher.....	79
5.2.4	DLT-based Transactive Logic.....	82
5.3	USER INTERFACE LAYER	83
5.3.1	Virtual Energy Console.....	83
5.4	COMMUNICATION LAYER.....	84
5.4.1	IANOS Secured Enterprise Service Bus.....	84
5.4.2	dEF-Pi.....	90
5.4.3	Back-end Agent.....	91
6	DYNAMIC VIEW.....	92
6.1	TERCEIRA PILOT ISLAND DYNAMIC VIEW	92
6.1.1	Use Case 1: Community demand-side driven self-consumption maximization	92
6.1.2	Use Case 2: Community supply-side optimal dispatch and intra-day services provision.....	95
6.1.3	Use Case 3: Island-wide, any-scale storage utilization for fast response ancillary services.....	97
6.1.4	Use Case 4: Demand Side Management and Smart Grid methods to support Power quality and congestion management services.....	99
6.1.5	Use Case 5: Decarbonisation of transport and the role of electric mobility in stabilizing the energy system.....	100
6.2	AMELAND PILOT ISLAND DYNAMIC VIEW	102



6.2.1	Use Case 1: Community demand-side driven self-consumption maximization	102
6.2.2	Use Case 2: Community supply-side optimal dispatch and intra-day services provision.....	106
6.2.3	Use Case 3: Island-wide, any-scale storage utilization for fast response ancillary services.....	108
6.2.4	Use Case 4: Demand Side Management and Smart Grid methods to support Power quality and congestion management services.....	110
6.2.5	Use Case 5: Decarbonisation of transport and the role of electric mobility in stabilizing the energy system.....	112
6.2.6	Use Case 6: Decarbonizing large industrial continuous loads through electrification and locally induced generation.....	114
6.2.7	Use case 7: Circular economy, utilization of waste streams and gas grid decarbonization.....	116
6.2.8	Use case 8: Decarbonization of heating network.....	117
6.2.9	Use case 9: Active Citizen and LEC Engagement into Decarbonization Transition	119
7	DEPLOYMENT VIEW	120
7.1	BUSINESS LAYER.....	126
7.1.1	VERIFY-D platform tool (LCA and LCC module).....	126
7.1.2	CrowdEquity Component.....	126
7.1.3	System Modeller.....	127
7.1.4	Grid-Oriented Optimizer - INTEMA.grid.....	127
7.1.5	CBA component.....	128
7.2	DECISION MAKING LAYER.....	128
7.2.1	Aggregation and Intelligent Segmentation.....	128
7.2.2	Forecasting Engine.....	129
7.2.3	Centralized Dispatcher.....	129
7.2.4	DLT-based Transactive Platform	131
7.3	USER INTERFACE LAYER	132
7.3.1	Virtual Energy Console.....	132
7.4	COMMUNICATION LAYER.....	133
7.4.1	IANOS Secured Enterprise Service Bus.....	133
7.4.2	dEF-Pi.....	135
7.4.3	Back-end Agent.....	135
8	CONCLUSIONS.....	137
	ANNEX I: IANOS COMMON DATA MODEL.....	138
	MEASUREMENT DATA MODEL.....	138
	STATUS DATA MODEL.....	139
	CLUSTER DATA MODEL.....	140
	ANNEX II: SECURED ESB COMMUNICATION DATA MODEL	141
	RATIONALE.....	141
	DATA FORMAT FOR FIELD DEVICES MESSAGES.....	141
	DATA MODEL MEASUREMENTS AND STATES NAMES	144
	Data model.....	144
	Measurements names.....	147
	States names.....	147

<i>Storage specific measurements names</i>	151
<i>Storage specific status names</i>	151
DATA COMMUNICATION FOR FIELD DEVICE MESSAGES	152
PLATFORM SECURITY	152
ANNEX III: SECURED ESB WEATHER INFORMATION COMMUNICATION DATA MODEL	154
RATIONALE	154
DATA MODEL FOR WEATHER INFORMATION MESSAGES	154
<i>Real time weather messages data model</i>	154
<i>Forecasted weather messages data model</i>	155
REQUEST API	156
SUBSCRIBE TO REAL TIME WEATHER DATA	157



List of figures

Figure 2-1: IANOS 4+1 architectural view.....	16
Figure 2-2: Example of OpendADR interactions.....	21
Figure 2-3: The USEF interaction model.....	22
Figure 2-4: EFI: a common language for energy flexibility.....	23
Figure 2-5: Standard EN 504910-12-2 is positioned at the S2 interface in the M/490 Flexibility Functional Architecture between the CEM and the Resource Manager.....	23
Figure 2-6: Overview of concepts and relationships in SAREF.....	24
Figure 2-7: MQTT Architecture.....	25
Figure 2-8: IEC 61850 Data Model Hierarchy.....	28
Figure 2-9: CHAdeMO V2X.....	29
Figure 2-10: UML Functionality Diagram Template.....	30
Figure 2-11: UML Sequence Diagram Template.....	31
Figure 2-12: UML Deployment Diagram Template.....	31
Figure 3-1: IANOS Conceptual Architecture.....	33
Figure 3-2: IANOS CrowdEquity Platform Main Use Case.....	34
Figure 3-3: ESSIM.....	35
Figure 3-4: Left: A system that appears to have no net yearly energy imbalance; Right: Hourly data showing imbalance over the year.....	35
Figure 3-5: ESDL MapEditor provides a geographical front-end to design new scenarios for ESSIM simulations.....	37
Figure 3-6: The ESDL MapEditor describing a piece of the energy system of Ameland, used as input for an ESSIM simulation.....	38
Figure 3-7: High level diagram of the position of CBA component in the IANOS Energy Planning & Transition Support Toolset.....	40
Figure 3-8: High level diagram of the position of CBA component in the IANOS Energy Planning & Transition Support Toolset and its interconnection with the rest components.....	41
Figure 3-9: Reflex.....	44
Figure 3-10: OptiMEMS internal system architecture.....	45
Figure 3-11: General platform architecture for CWD platforms.....	46



Figure 3-12: Local P2P energy trading based on blockchain technology.....	47
Figure 3-13: GOPACS: Grid Operator Platform in the Netherlands for market-based congestion solutions.....	48
Figure 3-14: Interoperability definition.....	50
Figure 3-15: Interoperability categories.....	50
Figure 3-16: Common data exchange format interoperability.....	52
Figure 5-1: VERIFY-D platform tool (LCA and LCC module) functionality diagram.....	74
Figure 5-2: CrowdEquity component's functionality diagram.....	75
Figure 5-3: ESSIM development view.....	76
Figure 5-4: INTEMA.grid – grid-oriented optimizer's functionality diagram.....	77
Figure 5-5: CBA component's functionality diagram.....	77
Figure 5-6: Aggregation and Classification functionality diagram.....	78
Figure 5-7: Forecasting Engine functionality diagram.....	79
Figure 5-8: KIPLO Core Platform functionality diagram.....	80
Figure 5-9: OptiMEMS functionality diagram.....	81
Figure 5-10: ReFlex functionality diagram.....	82
Figure 5-11: DLT-based Transactive component functionality diagram.....	83
Figure 5-12: Virtual Energy Console Functionality Diagram.....	84
Figure 5-13: CITRIC's architecture.....	85
Figure 5-14: Information exchanges matrix.....	87
Figure 5-15: IANOS Secure Enterprise Service Bus (ESB) Functionality Diagram.....	90
Figure 5-16: dEF-Pi component functionality diagram.....	91
Figure 5-17: Ivpp Back-end Functionality Diagram.....	91
Figure 7-1: VERIFY – LCA/LCC toolkit's deployment view.....	126
Figure 7-2: CrowdEquity component deployment diagram.....	127
Figure 7-3: ESSIM Deployment diagram.....	127
Figure 7-4: INTEMA.grid – grid-oriented optimizer's deployment view.....	128
Figure 7-5: CBA component deployment diagram.....	128
Figure 7-6: Aggregation and Classification deployment diagram.....	129
Figure 7-7: Forecasting Engine deployment diagram.....	129
Figure 7-8: Kiplo Core Platform deployment diagram.....	130
Figure 7-9: OptiMEMS deployment view.....	131

Figure 7-10: ReFlex deployment diagram.....	131
Figure 7-11: DLT-based transactive platform deployment diagram.....	132
Figure 7-12: Virtual Energy Console deployment diagram.....	132
Figure 7-13: iVPP ESB deployment diagram.....	134
Figure 7-14: dEF-Pi deployment view.....	135
Figure 7-15: Back-End Agent Deployment View.....	136

List of tables

Table 2-1: IANOS Actors.....	17
Table 2-2: IEC 61850 standard documents.....	27
Table 4-1: LCA/LCC Toolkit functional requirements.....	55
Table 4-2: CrowdEquity Platform functional requirements.....	56
Table 4-3: Cost Benefit Analysis Component functional requirements.....	57
Table 4-4: Grid Oriented Optimizer (INTEMA) functional requirements.....	57
Table 4-5: Forecasting Engine functional requirements.....	58
Table 4-6: Centralized Dispatcher functional requirements.....	59
Table 4-7: Aggregation and Intelligent Segmentation functional requirements.....	60
Table 4-8: Virtual Energy Console functional requirements.....	61
Table 4-9: DLT-Based Transactive Logic functional requirements.....	61
Table 4-10: Secure Enterprise Service Bus (ESB) functional requirements.....	62
Table 4-11: Back-end Agent Functional Requirements.....	62
Table 4-12: LCA/LCC Toolkit non-functional requirements.....	63
Table 4-13: CrowdEquity platform non-functional requirements.....	64
Table 4-14: Cost Benefit Analysis Component non-functional requirements.....	65
Table 4-15: Grid oriented optimizer (INTEMA) non-functional requirements.....	66
Table 4-16: Forecasting Engine non-functional requirements.....	67
Table 4-17: Centralized Dispatcher non-functional requirements.....	68
Table 4-18: Aggregation and Classification non-functional requirements.....	69
Table 4-19: Virtual Energy Console non-functional requirements.....	70
Table 4-20: DLT-Based Transactive Logic non-functional requirements.....	71



Table 4-21: Secure Enterprise Service Bus (ESB) non-functional requirements.....	72
Table 4-22: Back-End Agent Non-Functional Requirements.....	73
Table 5-1: CITRIC platform suggests technologies for bulk data ingestion	85
Table 5-2: Exchanged Information types	87
Table 5-3: Technical constrains per asset.....	88
Table 7-1: IANOS system detailed data interfaces.....	120
Table 7-2: IANOS system general deployment state.....	124



1 Introduction

1.1 Objectives and Scope of the deliverable

The main objective of this deliverable is to present a complete description of the third version of the Architecture of IANOS system and mainly the ICT and software components that will be deployed in the Lighthouse Islands. In this deliverable, IANOS IEPT and iVPP modules are presented in detail. In addition, their relations with the overall system components and the middleware that enables this communication (e.g. the Enterprise Service Bus) is presented. Furthermore, the data models and protocols that are going to be used in IANOS are defined and described.

The System Architecture is presented from various views, namely the Logical or Conceptual View, the Development View, the Dynamic or Process View and the Deployment View. Each view serves a different purpose and combined they provide the user with the necessary information regarding the functionalities, the functional and non-functional requirements, the communication protocols, the hardware and the data models utilized by the system components. Furthermore, IANOS's system is broken down into Different Layers namely the Business Layer, the Decision-making Layer, the User Interface Layer and the Communication Layer.

IANOS project is part of BRIDGE initiative, which is a cooperative group involving multiple projects in the areas of Smart Grid, Energy Storage, Islands, and Digitalisation Projects. BRIDGE aims at creating a structured view of issues encountered in the projects and providing necessary field experience, feedback and lessons learned, aiming to overcome the barriers to effective innovation.

1.2 Structure of the deliverable

The deliverable consists of seven chapters that are structured as follows:

- **Chapter 1** provides an introduction of the document, presenting the objectives and scope, the structure, the relation to other tasks and the abbreviation of the document. In addition, in this deliverable, which corresponds to the third version of IANOS System Architecture, a section containing the changes, additions and revisions in relation to the second version (D2.14).
- **Chapter 2** provides a detailed description of the methodology that the deliverable follows. More specifically, the design principles, the architecture view models, the different IANOS actors, related data models, protocols, and standards, as well as the UML diagrams that are used in the document are presented.
- **Chapter 3** presents the Conceptual Architecture and the Logical View of the system. The different components of the system are presented, with a short introduction to their functionality.

- **Chapter 4** provides the functional and non-functional requirements of IANOS system.
- **Chapter 5** introduces the Development view, where the inputs, outputs and subcomponents of each component and the interrelations and dependencies between components are presented with the use of UML functionality diagrams.
- **Chapter 6** provides the Dynamic View of the system, which focuses on describing the workflow and data exchange between the different components and actors of the system in each pilot island, with regards to the previously defined Use Cases, with the use of UML sequence diagrams.
- **Chapter 7** presents the deployment of software into hardware in the Deployment View of the system, with the use of UML deployment view diagrams, as well as detailed data interfaces and the system's general deployment state.

1.3 Relation to other Tasks and Deliverables

This deliverable extracts information from the defined Use Cases in D2.3. Based on the defined use cases, the inner modules of the iVPP and the IEPT toolkit are analysed according to an Architecture view model. In addition, the role of the iVPP's modules and their interactions amongst themselves and the external system is explicitly defined in this deliverable. Finally, based on the UCs the Process or dynamic view of the system is created and represented in detail with the Use of UML diagrams, giving insight to the workflow of IANOS's iVPP internal modules. Additionally, the deliverable considers the security standards defined in T1.4 Ethics and Cyber Security Management report.

The system architecture is used as input for tasks T.3.1-T.3.5 that are associated with the creation and deployment of the IEPT toolkit and its various modules. Furthermore, this deliverable is used as input for tasks T.4.1-T.4.5 that are associated with the creation and deployment of the iVPP and its various modules.

1.4 IANOS System Architecture second version changelog

This deliverable constitutes the third version of IANOS System Architecture. The changes from the second version deliverable (D2.14) are presented in this section. The changes in the respective chapters are as follows:

- Chapter 1: Minor changes
- Chapter 2:
 - Section 2.3:
 - The asset names and/or descriptions regarding the Water Taxis, Auto Generative High-Pressure Digester and Electrolyzer were changed to more generic names/descriptions as they are going to be finalized after the submission of this deliverable. Another version regarding the System Architecture might be written if necessary.
- Chapter 3:
 - Section 3.4.:
 - Added section 3.4.3 which describes the Backend Agent Component



- Chapter 4:
 - Section 4.1:
 - Added the responsible partner for each tool
 - Added/removed functional requirements
 - Added a new table for the Backend Agent
 - Section 4.2:
 - Added the responsible partner for each tool
 - Added/removed functional requirements
 - Added reliability and robustness requirements for the Centralized Dispatcher and the ESB
 - Added a new table for the Backend Agent
- Chapter 5:
 - Section 5.4:
 - Added Section 5.4.3 for the Backend Agent Component
- Chapter 6:
 - Changes to the Dynamic Views to reflect the alignment with D2.3
- Chapter 7:
 - Section 7.4:
 - Added Section 7.4.3 for the Backend Agent Component
- Annex 1:
 - Minor changes
- Annex 2:
 - Changes to the Section of the Data Model to reflect most recent developments regarding IANOS Data Model

2 Methodology

2.1 Design Principles

While designing the architecture of a system, there are certain principles that need to be kept in mind to handle the complexity, reduce the effort needed and provide all the potential users with the ability to easily make use of the project's functions. These design principles allow for a less complex and error-free development of the system, while mitigating potential problems of future maintenance or enhancement. In IANOS project a set of key design factors have been identified to assist in the development process of the system.

The first principle that is going to be followed is the **Separation of Concerns (SoC)**, which suggests that the system needs to be separated into distinct sections, each of them designed for a separate concern (feature). It also suggests that each section needs to be as independent as possible, with minimum overlap between features. The second principle to be followed is the **Law of Demeter (LoD)** or Principle of least knowledge that proposes that each architectural element of the system should have limited knowledge of the internal details of the other elements and interact only with the elements related to it. The **Single Responsibility Principle (SRP)** is another similar principle. SRP dictates that each system's element needs to be responsible for one and only specific functionality. For a new functionality, another element needs to be developed.

Keep it Short and Simple (KISS Principle), is a principle that suggests the more complicated systems tend to perform worse than the simpler ones, thus the simplicity of the systems is preferred. Respectively, **Don't repeat yourself (DRY)**, is another principle that aims to reduce the repetition of the same functionality from different elements. Based on this principle when a functionality is commonly used throughout the system, it must be implemented in more generic components that can be accessed by different elements.

2.2 Architecture View Model

In order to describe a system's architecture, a framework called view model is used. A view model defines representations of the whole system, called views, each of them related to the viewpoint of the different stakeholders, which vary from developers, system engineers, users, and project managers.

The model adopted to describe the architecture of the IANOS system is the **4+1 architectural view model**. In this model, each component of the system is described based on multiple views, in order to standardize the system's design, making it easier to be comprehended by each interested party. As its name suggests, there are four main views of the system, namely the Logical, Process, Development, and Physical View, with an additional level being the different scenarios that highlight examples of the



functionalities of the system. In detail, the different views of the IANOS system are described below:

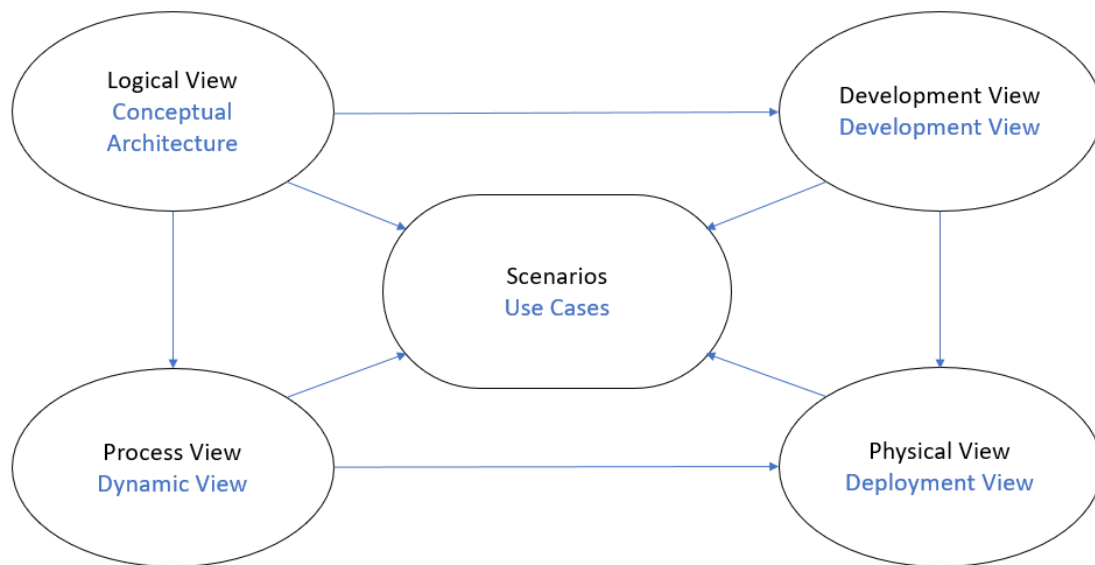


Figure 2-1: IANOS 4+1 architectural view.

- The **Logical view** is used to present the different components of the systems and their main functionalities. This view provides descriptions about the various layers of the system, the modules in each layer and the responsibilities of each component. Thus, a decomposition of the system is provided in order to factorize each element's functionality, making it clear for each party involved. Additionally, the interactions between the different components are provided. In IANOS system, this view is represented as the **Conceptual Architecture – Logical View** and its description is provided in Chapter 3.
- The **Development View** provides a description of the architecture for each element, containing information about the usage of existing developments, varying from access to legal constraints, the dependencies used, and the different tools used for the final implementation of each component. It offers the view of the software project managers and the programmers. In IANOS system, the internal architecture for each component and subcomponent and the information about their interrelations and dependencies are provided in Chapter 5.
- The **Process View** focuses on the dynamic view of the system, providing information about the workflow or process rules of the system and about the intercommunication of these processes. It includes interrelations, interactions and dependencies with other system components, showing the information flow between the components and different actors. The **Dynamic View** of IANOS system is presented in Chapter 6.
- The **Physical View** presents the deployment of software into hardware, together with its installation and configuration. It presents the different physical equipment that is used to assist the systems functionality. It offers the view of the system engineer. This view is presented in Chapter 7 as **Deployment View**.

- The final view of the architecture is known as **Scenarios**. Each scenario highlights the combination of the previous views in different use cases. It is usually used as a starting point, due to its effectiveness in finding and understanding all the architectural elements needed, but it is also used to validate the implementation. The Scenarios are named **Use Cases** in IANOS project and is extensively described in the implementation of D2.3.

2.3 IANOS Actors

In this chapter, a list of actors involved in IANOS project is presented, together with the type of each actor and a general description. An actor specifies a role played by human users, external hardware, or other subjects that interact with the system. While the role of several different actors can be played by a single physical entity, actors do not necessarily represent explicit physical instances but merely specific roles of some entities. The table includes different types of actors, varying from systems, modules, assets, individuals, or communities.

Table 2-1: IANOS Actors

Actor Name	Actor Type	Description
IANOS iVPP	System	The IANOS iVPP sets up a virtual network of decentralized renewable energy resources, both non-dispatchable such as wind, solar, tidal resources and dispatchable ones such as geothermal and green gas CHP plants. Moreover, the iVPP comprises Energy Storage Systems (ESS), integrated as a single unit, providing flexibility services and fostering island renewable energy self-consumption.
Power Production Dispatchable Assets	System	Power generation assets (geothermal power plant, waste incineration plant, fossil fuels generators), from which power can be dispatched on demand at the request of grid operators when needed.
Power Production Non-Dispatchable Assets	System	Local power generation assets from which power cannot be controlled by grid operators such as wind, solar and tidal power generators.
Production-Side Assets	System	Residential PV panels and assets from community owned areas such as small wind turbines, fuel cells and micro-CHP systems.
Storage Assets	System	Assets of any-scale that can store energy for later use such as flywheels, distributed and centralized electrochemical batteries. Other means of very flexible production units such as Fuel Cells are also included. These assets are aggregated and controlled by the IANOS iVPP.
Other Large-Scale Storage Systems	System	Other large-scale storage systems such as large-scale systems producing alternative fuels (electrolyzers). In Ameland, an electrolyzer will be installed
Large-Scale BESS	System	Large-scale battery technology system (10.5MWh in Terceira and 3 MWh in Ameland) which stores energy to be used later. It is connected to distribution/transmission networks.

Actor Name	Actor Type	Description
Energy Demanding Assets	System	Energy demanding assets of the island.
Electricity Grid	System	Power systems including power generation, transmission and MV/LV distribution.
Gas Grid	System	Network where gas is transported, to feed CHPs and fuel cells.
District Heating Network	System	Pipe network which provides heating and hot water from a central power plant for connected consumers.
Electric Vehicle	System	A vehicle with an electric drive and a battery which can be charged at a charging station.
Electric Charging Station	System	System that connects an electric vehicle (EV) to a source of electricity to recharge vehicle's battery.
V2G Charging Station	System	Bidirectional system that connects an electric vehicle (EV) to a source of electricity. Besides recharging the vehicle's battery, it enables to provide balancing services.
H2 fuelled vehicles	System	Hydrogen fuelled vehicles .
Natural Gas Platform	System	Power intensive energy consumers from the industrial sector. In the case of Ameland, there is a Natural gas platform located on Ameland's coast which has been operated by the Nederlandse Aardolie Maatschappij (NAM) since 1986. Current natural gas production is close to 1 million m ³ /day, of which 100k m ³ /day is used as fuel to power the platform (mainly compression).
Hydrogen Demanding Assets	System	Assets which consume hydrogen.
Hybrid Heat Pumps	System	Hybrid heat pumps run on both electricity and natural gas and are composed of a 20 kWth boiler and a 1.1kW _e /5kW _{th} heat pump, thereby allowing to a switch between gas and electricity operation. Hybrid heat pumps can also run on biogas.
Home Energy Management System (HEMS)	System	Energy management system used for real time monitoring of energy consumption/generation, controlling domestic devices and electric circuits, accessing smart meter data and real time energy consumptions. HEMS is responsible for gathering flexibilities within the customer premises and providing them to the iVPP platform. Briefly, the system is composed by the hardware (Smart Meters, Sensors and Actuators), Data Management (Communication, Data Processing and other modules) and User Interfaces (UI).
GOPACS	System	Grid Operation Platforms for Congestion Solutions interface (GOPACS) is a unique initiative in Europe and has resulted from active collaboration between the Dutch TSO and the DSOs. This platform is consistent with key European directives to mitigate grid congestion, while offering large and small market parties an easy way to generate revenues with their available flexibility and contribute to solving congestion situations.
Fuel Cells	System	Assets with the ability of offering electricity, when necessary, while also supporting the synergy between energy grids (NG and electricity in the specific case).



Actor Name	Actor Type	Description
Electrolyzer	System	An electrolyzer will be installed in Ameland.
Digester	System	The digester solution to be implemented is now being decided by the LH managers and, if needed, a further version of this deliverable will describe it and its specifications.
Local Energy Community	Role	Decentralized cooperatives of local communities and citizens that promote the production and consumption of local energy. Local energy communities share a common long-term goal for a sustainable future of energy and work to advance the transition through active citizenship engagement. In Terceira there is not a LEC yet, while in Ameland already exists.
Citizens	Role	Citizens who live in the community.
Local Stakeholders	Role	Local stakeholders present in the community.
Mediator (E.G. Local Authority)	Role	Person who helps to connect the community and the project.
Potential Project Developers	Role	Project Developers interested in applying biomass technologies to reduce waste streams.
Prosumer	Role	End-user of electricity, gas, water or heat that can also generate energy using a Distributed Energy Resource.
Forecast Service Provider	Role	Monitors energy data from prosumers and provides an overproduction report based on forecast performed for prosumer's energy consumption and production.
Weather Forecast Provider	Role	Provides generation, consumption and weather-related operational risks, for a given location and a specific time horizon for non-dispatchable generation assets.
Supply And Demand-Side Assets	Device	Private end-user's energy assets such as electrochemical and heat batteries, electric water heaters, electric vehicles, smart home appliances and smart plugs. Additionally, it also comprises assets from community owned areas, for instance hybrid heat pumps and biobased saline batteries.
Fog Enabled Intelligent Device Plus (Feid-Plus)	Device	Device that performs resource-intensive functionalities such as computation, communication, storage, and analytics locally next to the end-user assets instead of forwarding data to cloud-based servers to be processed. The FEID-Plus is a fog-enabled computing device equipped with special functions to control I/O, phase width modulation and analog signals. It employs enough processing capacity for applying distributed computing such as information capturing and storing, algorithms execution and control over the installation. Additionally, it also has the capacity to interface with several field elements for instance controllable building loads, storage and EV charging stations through appropriate protocols.
Hybrid Transformer	Device	Device with the capability of regulating the voltage during operation. This hybrid transformer is able to implement a dynamic voltage regulation actuation, in each phase, with unlimited number of operations and with the addition of other features such as the contribution to reactive power compensation, unbalance correction and improvement in the voltage profile quality.
Smart Energy Router	Device	Power electronic device that provides grid-to-grid communication, load management and integration of multiple generation &

Actor Name	Actor Type	Description
		storage units, heterogeneous appliances and existing distribution grid. Moreover, it also allows to provide ancillary services to the grid.
Smart home appliances	Device	Devices which are interconnected through the internet, allowing the user to control functions remotely using a mobile or other networked device.
Smart Plugs	Device	Plugs which can be controlled remotely through a mobile and allow to control and automate small appliances and home devices.

2.4 Related Data Models, Protocols and Standards

In this section, an analysis of the relevant data models, protocols and standards are presented. More specifically, the concepts that are going to be reviewed, to be used in IANOS project are the following:

- Standards for information communication for Demand Response (OpenADR)
- Frameworks describing the market for flexibility (USEF)
- Communication protocols for energy flexibility (EFI and EN 50491-12-2 (S2))
- Data models for IoT devices (SAREF, MQTT)
- Power Utility Automation, Hydro Energy Communication, and Distributed Energy Resources Communication standard that applies mainly to energy systems (IEC 61850)
- Vehicle to Everything (V2X) protocol (CHAdemo)

2.4.1 Open Automated Demand Response (OpenADR)

Effective implementation of Demand Response (DR) services requires rapid, secure, reliable, and consistent two-way communication between a wide variety of participants, varying from power plants, possibly aggregators, and consumers. The latter is a necessity in explicit DR, where the automation of decision is of key importance. The standardization in this communication is necessary, due to specific strict requirements of the energy supply and the larger number of stakeholders.

OpenADR (Open Automated Demand Response) is an open-source information exchange model and global Smart Grid standard used in DR applications. Its most common use is for insufficient demand scenarios when signals are sent to devices to cause them to shut down during periods of higher demand. It determines the exchange of information between energy suppliers and energy control systems.

The two basic elements of OpenADR architecture are Virtual Top Nodes (VTNs) and Virtual End Nodes (VENs). These are the entities between which interactions take place. VTN is the server that transmits OpenADR signals to end devices, and works on the utility or the load aggregator side. A VEN is the client that recites on the device side and accepts the OpenADR signal from a server (VTN). In many cases a VTN and VEN can be the same

device, for example a DR aggregation server can act both as a VTN for end devices, and as a VEN for a utility DR signal.

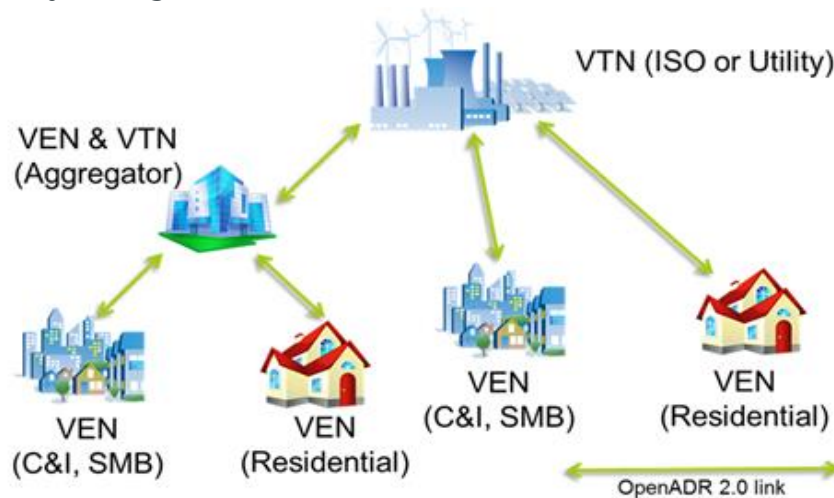


Figure 2-2: Example of OpenADR interactions.

The OpenADR standard allows a response signal to the DR event to return from VENs to the VTNs, together with other information related to DR events, varying from the name, identification, status, and other enumeration data describing the event, data showing the operating mode, signals for reliability and emergency, details about the renewable generation status, data related to market participation and test signals. The implementation of the services is based on a common language over any existing TCP/IP based communications network.

Currently the OpenADR 2.0 specification contains two main profiles. Profile A was developed for low-end embedded systems with restraints for resources, using only basic DR services and markets. Profile B is designed for fully functional control systems and devices and enables feedback and additional services. OpenADR 2.0b provides services for notifying VENs of upcoming DR events and sending demand response signals from the VTN to the VEN (EiEvent), for opting in/out of DR events by VEN (EiOpt), for reporting information from VEN to the VTN (EiReport), in order to predict and monitor the behaviour of the demand-side loads associated with the VEN and finally for establishing communications between VEN and VTN (EiRegisterParty).

2.4.2 Universal Smart Energy Framework (USEF)

USEF is a framework that is able to deliver the market model for the buying and selling of energy flexibility, and the architecture, tools, and rules to assure its efficient work. It has been developed to provide specifications for a faster and more cost-effective route to an integrated smart energy future. USEF delivers one standard with which it is possible to assemble flexibility from business customers as well as consumers, combine them and define specific flexibility solutions.

USEF can integrate new and existing energy markets, by extending existing processes, making it a perfect fit to most of the energy models. The framework is created

to provide all international stakeholders with fair access to energy market and benefits by introducing a new market-based coordination mechanism, which optimizes the value of flexibility across all roles in the system.

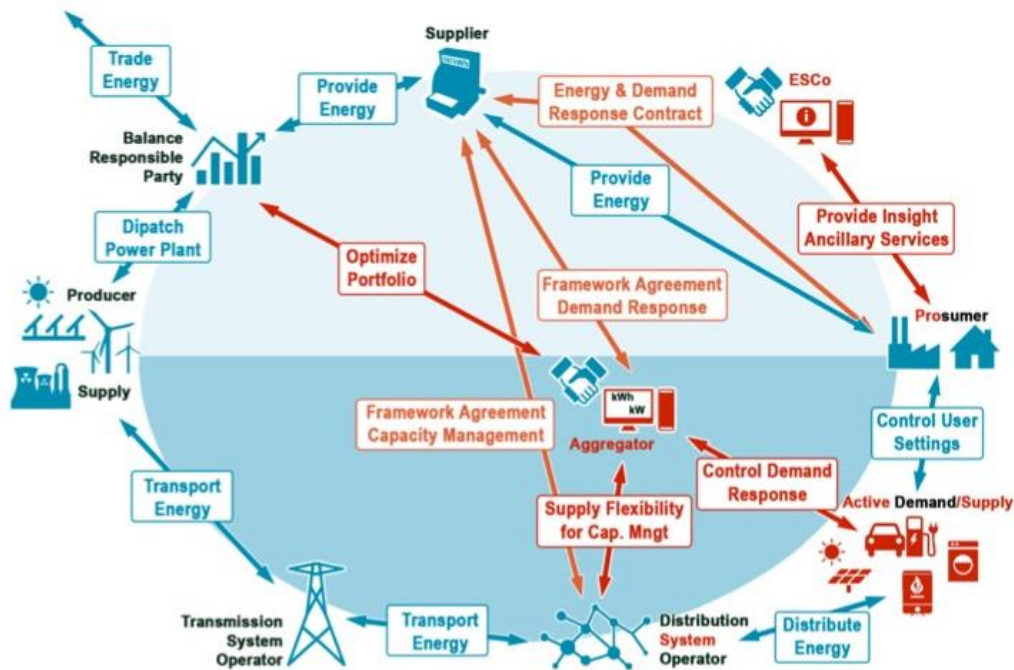


Figure 2-3: The USEF interaction model.

USEF Foundation, which develops, maintains and audits USEF, is a non-profit partnership founded by seven key players, all active across the smart energy chain. More specifically, the partners are Alliander, Stedin, ABB, DNV-GL, IBM, ICT and Essent.

2.4.3 Energy Flexibility Interface (EFI) and EN 50491-12-2 (S2)

EFI has been specifically developed as a standard for communication methods between smart devices and Demand-Side Management (DSM) solutions. EFI enables end users to control various smart energy devices, such as EV chargers or solar panels, thus unlocking the opportunities of flexible energy. EFI is an open-source standard that allows communication between smart grids and smart devices, allowing the allocation of flexible energy which helps the transition to affordable and sustainable energy supply.

Initially, DSM solutions that exploit the flexibility of energy devices, such as OpenADR or USEF, developed their own language to communicate with the energy devices in order to model flexibility. Due to the large number of energy devices the latter does not scale well, thus creating a need of a common language between devices and DSM solutions. EFI provides a common language, with which DSM services are allowed to communicate with all devices without having to develop custom adapters for each combination.

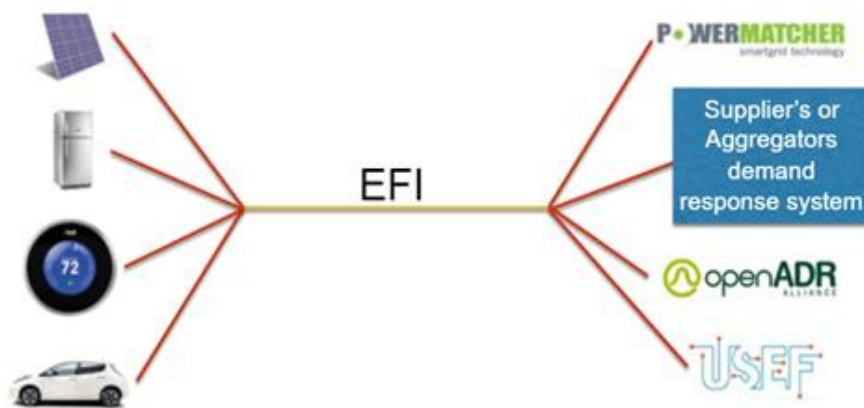


Figure 2-4: EFI: a common language for energy flexibility.

The interface specification of EFI is freely available, while the model is also available to developers in its GitHub repository <https://github.com/flexiblepower> including a reference implementation.

EFI is the starting point for a European standard for defining and communicating energy flexibility. As of this writing its successor has just been published as EN 50491-12-2. It is positioned within the M/490 European Flexibility Architecture as the S2 interface, where it defines the communication between the Customer Energy Manager (CEM) and the Resource Manager that represent devices that provide energy flexibility and provide conversion to native protocols of those devices.

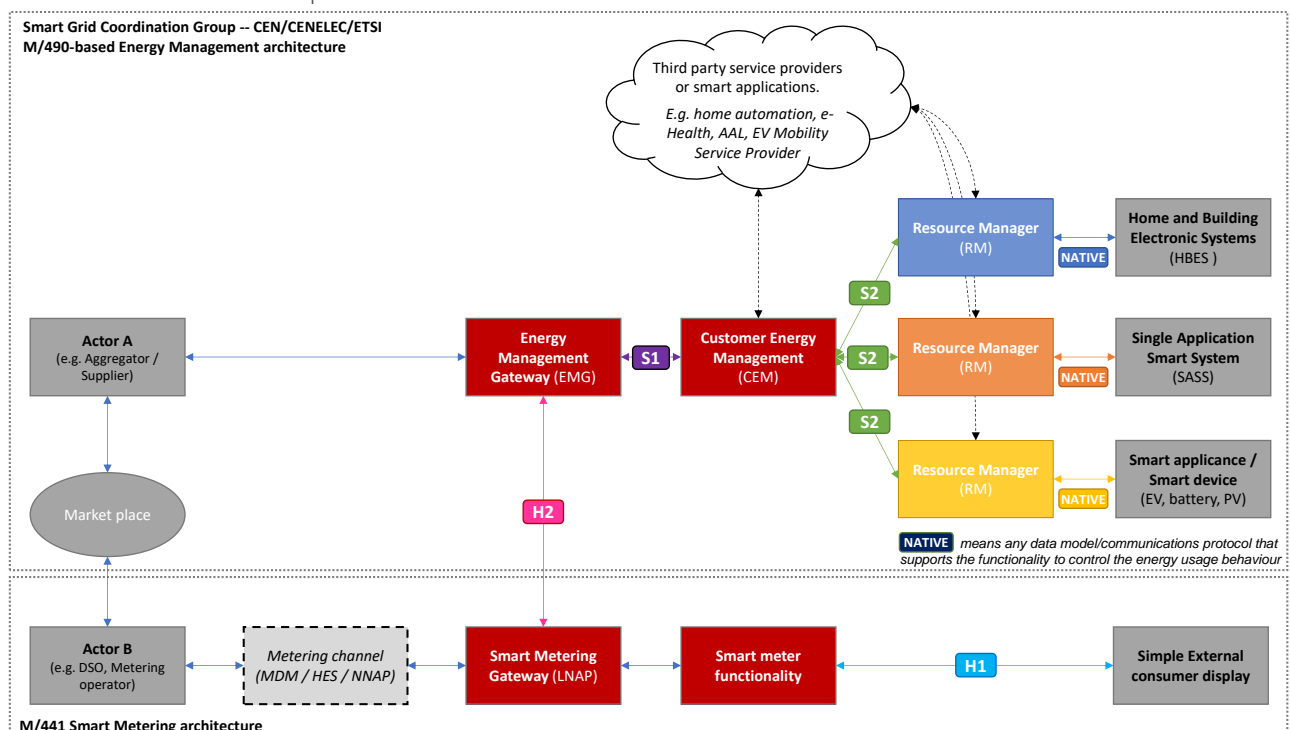


Figure 2-5: Standard EN 504910-12-2 is positioned at the S2 interface in the M/490 Flexibility Functional Architecture between the CEM and the Resource Manager.

The design philosophy of this interface is to divide the energy management responsibilities between the CEM and the Resource Manager. The Resource Manager

describes the ‘technical’ flexibility capabilities and constraints, whereas the CEM focuses on how to utilize the value of that flexibility by using e.g. incentive schemes.

The EN 50491-12-2 interface specification supports a clear separation of concerns between devices and CEMs by focusing on a generic description of energy flexibility. The flexibility capabilities provided by smart devices can be mapped onto this generic description of flexibility, while it does not impose any limitations on CEM algorithms to exploit this flexibility in a specific way.

EN 50491-12-2 will be implemented into IANOS architecture for the Ameland-related UCs through the dEF-Pi open platform, for exchanging information regarding energy flexibility to the Decision Making Layer of the iVPP (ReFlex component).

2.4.4 Smart Appliances REference Ontology (SAREF)

The majority of the household devices that consume energy in modern applications consist of highly intelligent and networked devices. Based on the latter, it has become possible to manage and optimize the energy usage at a device level in order to reduce the overall energy usage. Due to the heterogeneity of the device systems, a standardization of the interfaces on device and sensor level is needed. Therefore, a common architecture is needed, together with other relevant standards.

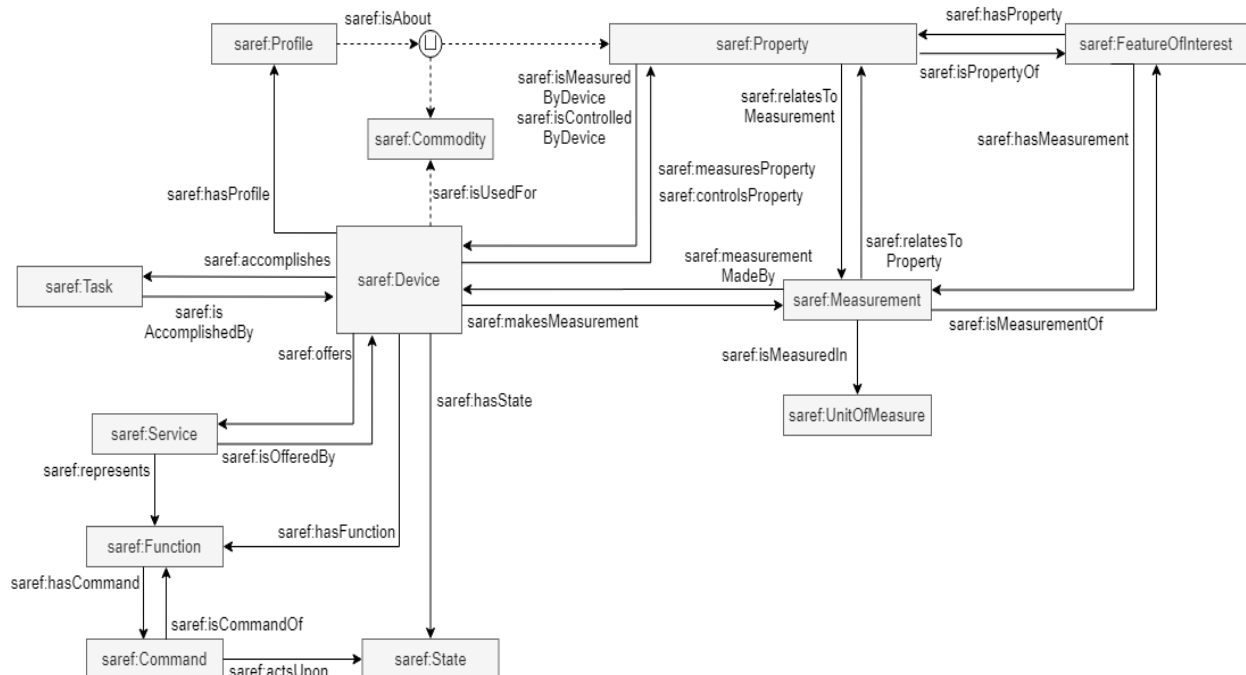


Figure 2-6: Overview of concepts and relationships in SAREF.

The Smart Applications REference ontology (SAREF) aims to enable semantic interoperability between intelligent devices from different providers and among various activity sectors on the Internet of Things (IoT), by providing a reference ontology and thus a unified architecture for appliances. An ontology, such as SAREF, provides the ability to create methods for device abstraction, without the need for the energy-saving application developers to know the exact specifics of the various standards.

SAREF provides the language to specify repetitive key concepts in the smart appliances' domain, alongside with the relationships between them and rules to determine the usage of the concepts and the relationships. Additionally, SAREF deliver mappings to different concepts used by other standards.

2.4.5 Message Queuing Telemetry Transport (MQTT)

MQTT is a standard for communication between smart meters that provides a messaging protocol for the IoT. MQTT delivers bi-directional communication between the cloud and the smart meters, establishing an easy broadcasting of messages.

Two types of network entities are defined based on the MQTT protocol, the MQTT broker and the MQTT client. The client can be any smart meter or device that is connected and transmits messages to a MQTT broker through a network. The messages transmitted by the client are received by the broker and rerouted to the appropriate destination.

When a new data item of a certain topic needs to be distributed, a message containing the data is sent to all connected brokers. Next, the information is published from the broker for all the clients that the topic concerns or it is discarded if there are no clients subscribed to that topic, unless it is flagged by the sender as retained message. Retained messages are stored by the broker until a client subscribed to a topic pattern similar to the retained message topic is found or a new retained message of the same topic is stored.

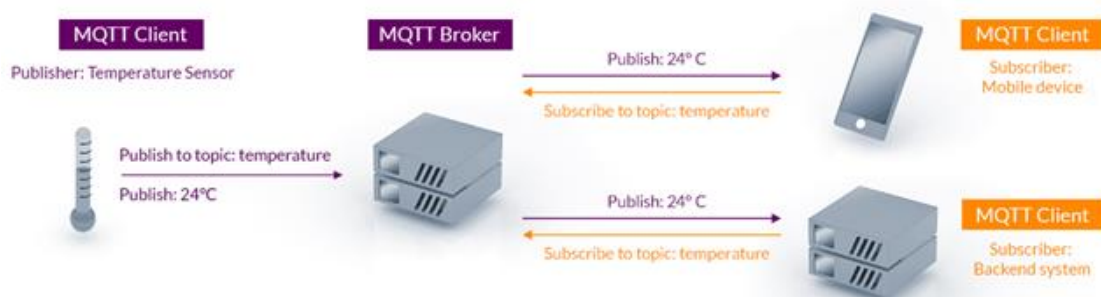


Figure 2-7: MQTT Architecture.

The sender and subscriber do not have information about the location or the IP address of each other, and they only interact with the broker. In this way the devices are decoupled, which makes the communication more secure. Additionally, MQTT can encrypt messages with the use of Transport Layer Security (TLS) encryption that requires clients to use connections protected by username and password. Optionally, clients can be obliged to provide a certification that matches with the certification of the server. MQTT has a Quality of Service (QoS) measure to assure the reliability of the message publishing. Each message send is characterized by one of three quality of service levels: "at most once", where the broker takes no action to confirm the delivery of the message that is only sent once, "at least once", where the message is re-send multiple times until

acknowledgement is received and “exactly once”, where it is ensured that only one copy of the message is published.

2.4.6 Advanced Message Queuing Protocol (AMQP)

Advanced Message Queuing Protocol (AMQP) is an application-level open standardization protocol for messaging-oriented intermediate software, based on TCP/IP. It enables the communication between different message-oriented middleware, providing business processes with the required data and transmitting instructions to achieve the objectives.

One of the main components in AMQP are **queues**. Each message always ends in a queue, even if it is a private memory queue that feeds a customer directly. The queues are like mailboxes at the destination or in the intermediate booking areas at the sorting office. Queues can store messages in memory or on disk, as well as search and rearrange messages and can participate in transactions. Another component is the **exchanges** that provide sorting and delivery services. The concept of exchange enables AMQP to use different mid-term delivery models. Exchanges may vary from instant exchange that will send a message directly to a single queue, subject exchange, where the message is copied and sent to all customers who have expressed interest in a subject, and headers exchange, where all the headers in a message are examined by evaluating them based on the query templates provided by the interested customers. The final component is the **bindings** that are the parameters provided in exchanges to enable message routing. Bonds differ depending on the nature of the exchange.

The AMQP protocol uses frames to establish connections, transfer data and eliminate connections between two peers. To exchange messages, links must be established. To establish and tear down a new link, the attach and detach frames are used, respectively, and transfer frame is used to send messages through an existing link. Flow frames are used to manage the volume of messages in the process. A disposition frame is used to communicate the changes of the state of a message, using reliability guarantees, such as at-most-once, at-least-once, and exactly-once. If a conversation between of peers is initiated, by the exchange of multiple messages between peers, then the links can be grouped into a session, which starts and stops by a begin and end session, respectively.

In AMQP the message is divided into two conceptual parts. The first part is the **Bare Message**, which is the main message that cannot be altered during transport from intermediate nodes. The second part is the **Annotated Message**, which includes bare message and adds additional information regarding the routing and transmission of the message as well as its identification. The annotations can be added by any intermediaries during the transit. They may include a header that contains the message transfer features, such as message priority or durability, delivery and message annotations, message properties, such as the message-id or the subject of the message, application-properties that any intermediate node can acquire, application-data and a footer containing details to the message or its delivery.

2.4.7 IEC 61850 Standard

IEC 61850 has been developed as international standard that defines protocols for communication between smart devices at electrical substation. As a part of the IEC TC 57 reference architecture for electric power systems, its main goal is to improve the automation of the electrical substation. It provides definitions for the smart devices in an electrical substation and for their bidirectional communication.

IEC 61850 has been in development since 1995, aiming to provide a common standard incorporating many incompatible standards for device communication in an electrical substation. The first version, IEC 61850 ed. 1.0, was developed in 2004 and covers not only substation, but also battery systems, electrical vehicle supply equipment, distributed energy resources and more. The second version, IEC 61850 ed. 2.0, has been under development since 2005, in order to provide control remotely and data acquisition in real-time, while being independent of technology and platform used in the applications. It contains several standard documents, an example of which is presented in the following table.

Table 2-2: IEC 61850 standard documents

Standard	Title
IEC 61850-1	Introduction
IEC 61850-2	Glossary
IEC 61850-3	General requirements
IEC 61850-4	System and project managements
IEC 61850-5	Communication requirements for services and functions
IEC 61850-6	Configuration description language
IEC 61850-7-1	Communication Structure - Principles and models
IEC 61850-7-2	Communication Structure - Basic models, abstract services and basic types
IEC 61850-7-3	Communication Structure - Common Data Classes
IEC 61850-7-4-xx	Communication Structure - Domain specific Logical Nodes and Data Classes
IEC 61850-7.5-xx	Communication Structure - Modelling concepts and guidelines
IEC 61850-8-xx	Specific communication service mapping
IEC 61850-9-xx	Specific communication service mapping (from Sampled Values)
IEC 61850-10	Conformance testing



The architecture of IEC 61850 is based on the client-server model, where the communication is established between servers, which include protection or control devices, and clients that include gateways. At the same time, independent communication is achieved between smart devices via multicast messages.

A hierarchical data model is defined by IEC 61850 which contains the following levels. **Servers** that provide connection points for physical intelligent electronic devices (IED), are first in the hierarchy of the data model of IEC 61850. The information about a physical smart electronic device that performs several functions is organized by the **Logical Devices (LD)**, by splitting them into different categories, such as Control (CTRL), Disturbance Record (DR), Extended (EXT), Measurement (MEAS), or Protection (PROT).

The functions of the devices, related to control, measurement, protection and more, are represented by **Logical Nodes (LN)**. There are currently 92 logical nodes available for each device separated into 19 categories, depending on its functionality. Each logic node is identified by four letters, the first of which indicates the category. For example, the logical node MHA1 is referred to Measurements HARmonics and Inter-harmonics.

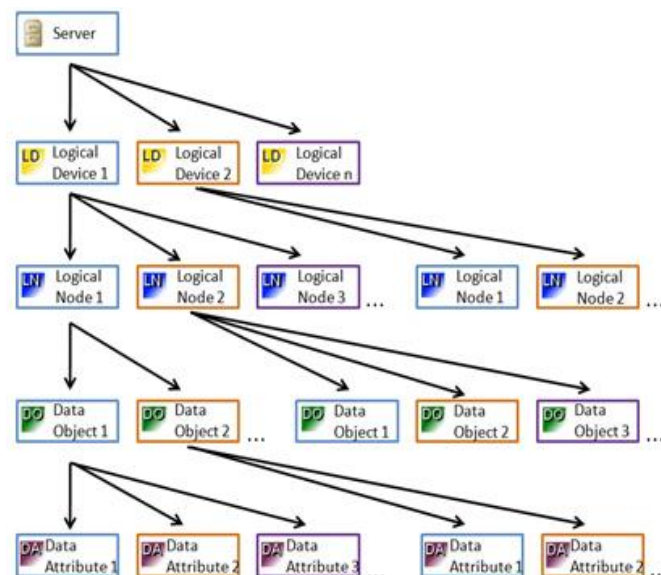


Figure 2-8: IEC 61850 Data Model Hierarchy.

In each LN there are **Data Objects (DO)**, which contain the mandatory and optional variables of the specific node. The standard defines a table of data objects for each node that can contain **Data Attributes (DA)**, such as measurements, information, status and more.

IEC 61850 offers several services. The **Control** service enables clients to command the operation of a switch on the primary equipment. With the **Report** service, clients receive information from servers for each new event or change of status that takes place within the substation. The **Logging** service essentially collects and sorts system events in chronological order, thus creating historical files. The **Substitution** service allows the operator to substitute the value of a variable for the actual value. This feature is used in cases of scheduled maintenance. With the **Setting Group** service, the operator from the substation control centre, or from the remote-control centre, can change the set of settings that exist in the protected computers. The **Sampled Value** service provides the

ability to transmit analogue sampled sizes between devices through a Publisher/Subscriber relationship. **Time Synchronization** is the service that offers international UTC time to the devices that make up the IEC61850 network as well as to the time stamped messages. The **File Transfer** service provides the ability to transfer files, such as computer configuration, transfer of fault records from the protection computers to the main computer, etc. Finally, the **GOOSE - Generic Object-Oriented Substation Event**, provides the ability for reliable and fast decentralized distribution of information between devices, offering the simultaneous delivery of information to one or more physical devices via multicast & broadcast.

2.4.8 CHAdeMO V2X protocol

Vehicle to everything (V2X) term refers to a series of systems that will allow an electrical vehicle (EV) to discharge in order to supply electricity to the grid (Vehicle to Grid, V2G), at home (Vehicle to Home, V2H) or any load (Vehicle to Load, V2L). Using V2X technology, the EVs can be utilized not only as vehicles but also as portable batteries, saving costs for the EV owners, by optimizing the energy usage, while providing flexibility services to the grid. In order to achieve this, the charging station and the vehicle needs to be supplied with the required technology, namely bi-directional charging of the EV must be established and standardized, as well as communication protocols to enable communication between the charger and the car and control systems.

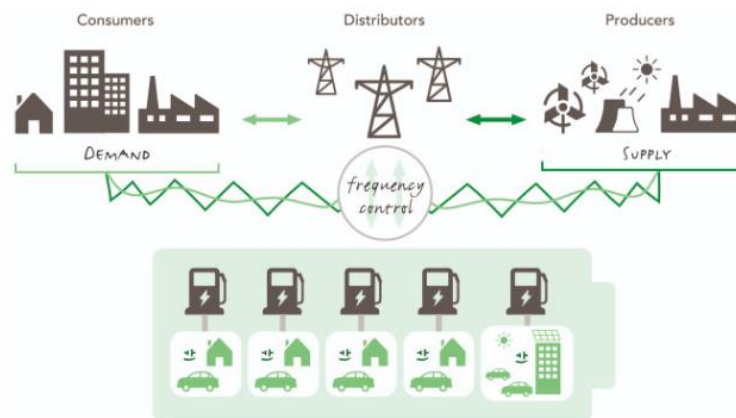


Figure 2-9: CHAdeMO V2X.

CHAdeMO is an association formed by five major automakers based in Japan that is attempting the standardization of fast charging methods for EVs as well as vehicle to everything protocols. CHAdeMO provides the only available protocol detailing V2X extension for EVs and also certified V2X chargers and vehicles. The protocol not only dictates strict guidelines for the design of EV chargers, providing a guaranteed electrical safety, but also enables bi-directional charging capabilities making it ideal for smart grid projects. CHAdeMO association has already established several demonstration projects since 2012, mainly in Europe and Japan.

2.5 UML Diagrams

Unified Modeling Language (UML) is a graphical language for visual representation, specification, and software-based systems documentation. Thus, to visually represent the system, as well as its components and the data flows between them, different types of UML Diagrams were created. The system's components that are represented in the diagrams include the different functional elements of the system, external entities, interfaces, and connectors. There are four UML diagram templates created for the representation of the system, more specifically information view diagrams, activity diagrams, sequence diagrams and deployment diagrams.

The data structure and the information flow between the components of the system are represented via UML functionality diagrams. This type of diagrams provides information about the types of data in each component, information about the ownership, generation, and storage of data, as well as information exchange details. They have been proven very useful in systems with many components, highlighting the communication and data exchange between different components. By developing the correct functionality diagrams, the detailed architecture of the system can be presented, and the potential inefficiencies and risks can be identified. The template of the UML functional diagrams used in IANOS project is shown in Figure 2.10

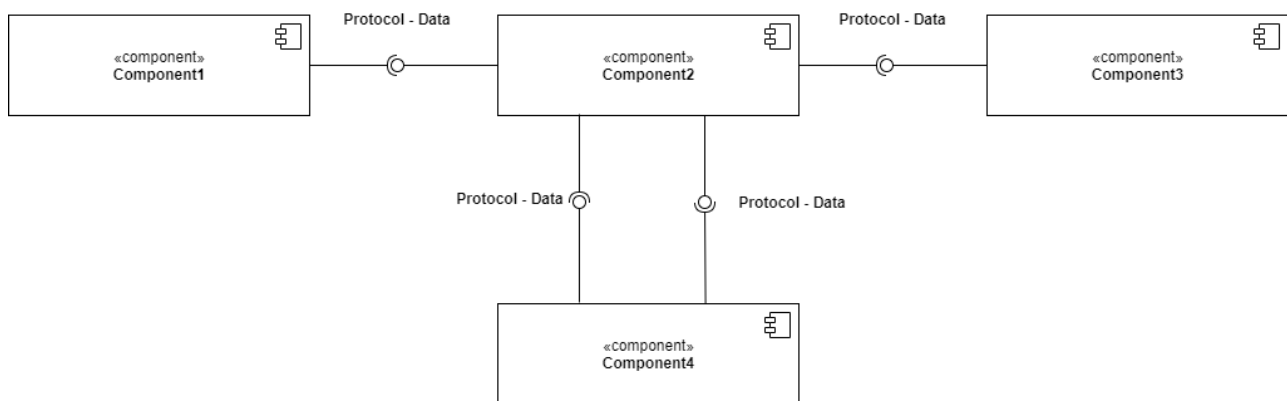


Figure 2-10: UML Functionality Diagram Template.

Another type of UML diagrams used in IANOS are UML sequence diagrams. As their name suggests, sequence diagrams describe the sequence of the information exchanged between the actors, objects, and components of the system. As Figure 2.11 shows, in the horizontal axis there is a representation of the entities of the system and in the vertical axis a representation of the time. The communications between the entities are represented in a chronological manner, thus focusing on the order that the events are carried out. The correct use of UML sequence diagrams is very important in order to document the exact processes that the system executes and its requirements.

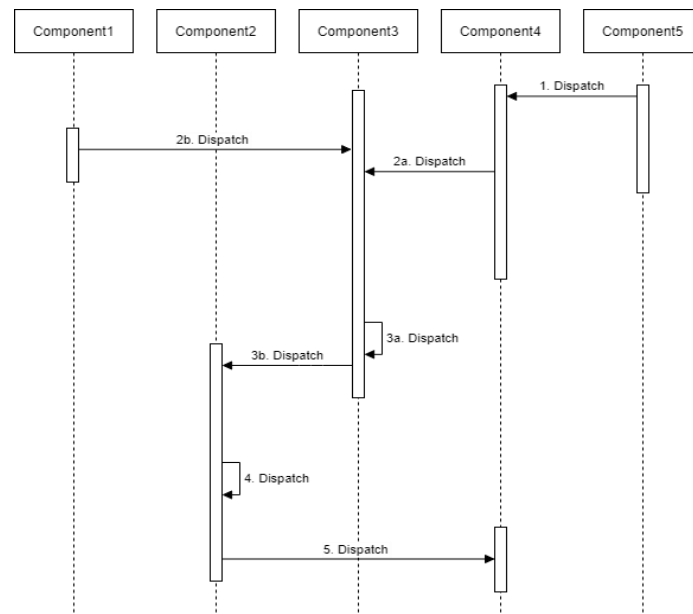


Figure 2-11: UML Sequence Diagram Template.

Lastly, the UML deployment diagrams depict the hardware of the system and its relations with the corresponding software. In UML deployment diagrams, the hardware devices or application server and database server are presented as three-dimensional boxes, called nodes, and inside the nodes the different components are represented as rectangles, as shown in Figure 2.12.

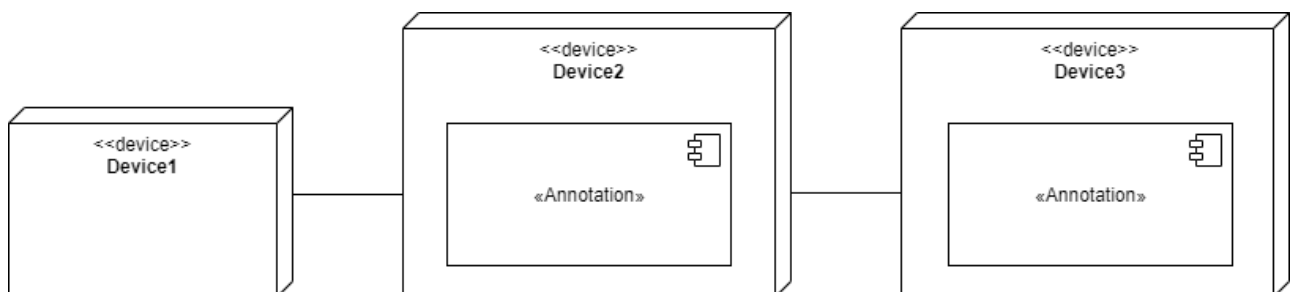


Figure 2-12: UML Deployment Diagram Template.

3 Conceptual Architecture – Logical View

In this chapter, a detailed overview of IANOS system's architecture is provided. The architecture is divided into separate layers, corresponding to the general functionality of the components in each layer. The overview includes the different layers of the system, as well as the components of each layer. In addition, information about the main functionalities of the components is provided, which corresponds to the logical view of the system.

The vision of the project is to provide solutions for the decarbonization of the energy system and the energy independence of the islands. This will be implemented via researching novel functionalities, services and assets, such as renewable energy systems, storage systems, Internet of Things (IoT) devices and Demand Response programs.

In the following figure, the conceptual architecture of the IANOS system is presented. The system is separated into five different layers: the Business Layer, the User Interface Layer, the Decision Making Layer, the Communication Layer and the Physical Layer. In the following chapter, a description of the layers is provided, together with an introduction of each layer's components. The Physical Layer, which contains the various physical assets and devices used by IANOS, was introduced in D2.1, "Report on Islands requirements engineering and Use Cases definitions", and therefore is not further analysed in the current document, except the inclusion of "Non-intrusive characterization and use of energy flexibility in water heating systems".

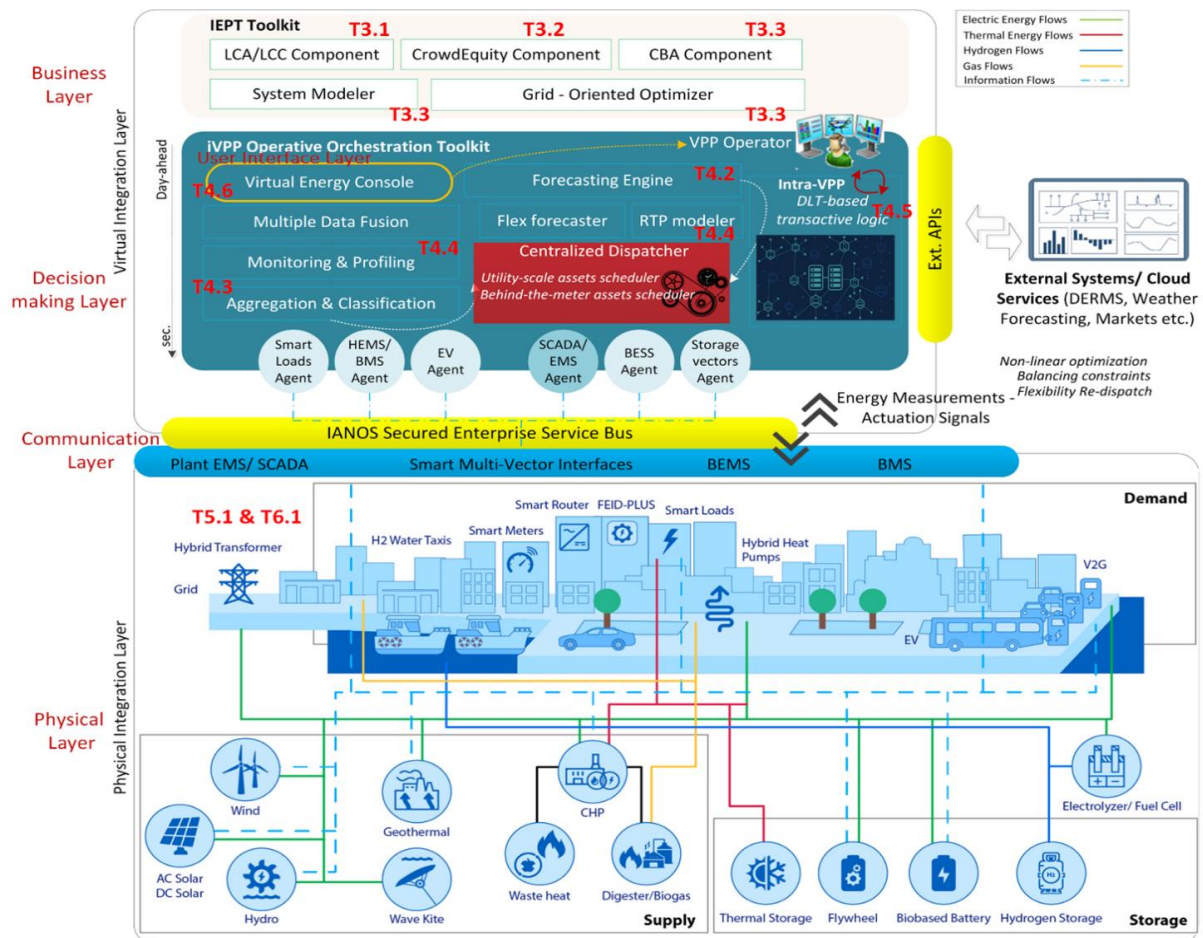


Figure 3-1: IANOS Conceptual Architecture.

3.1 Business Layer

IANOS Business Layer represents the Island Energy Planning and Transition Suite (IEPT) Toolkit. The toolkit's main focus is to assist all the stakeholders to develop an effective renewable energy portfolio and decarbonization plan. This layer includes all the components that are needed to boost new collaborative sustainable investments and promote co-ownership of assets, provide Life Cycle Assessment (LCA) analysis to optimize environmental costs and act as long-term decision-making aggregator. In conclusion, the business layer reflects IANOS business models and its general business logic.

3.1.1 VERIFY-D platform tool (LCA and LCC module)

The VERIFY-D platform tool belongs to the Island Energy Planning and Transition Suite (IEPT) of IANOS. The VERIFY-D tool will perform life cycle environmental and cost assessment of the proposed implemented RE technologies in the islands as well as of the storage devices:

The Lifecycle Assessment (LCA) evaluates parameters such as:

- Energy savings due to the implementation of the IANOS interventions
- Reduced fossil fuel consumption

- c) Reduced Greenhouse Gas Emissions
- d) Primary Energy Demand and Consumption assessment

The Lifecycle Cost (LCC) module is part of the VERIFY-D platform dedicated to evaluate the direct, indirect, internal, and external costs of the implemented technologies at all stages during a project's lifetime (capital, operation and maintenance and end-of-life costs).

The evaluation of the implemented technologies will be conducted based on the definition of the KPIs in T2.3.

3.1.2 CrowdEquity Platform

The term crowdfunding describes the collective funding of a project or company by a broad group of investors, usually via the Internet. There are several models of crowdfunding, one of them is the equity crowdfunding, or crowd equity, where the return of the findings is in the form of shares, or equity in that company or project, making each individual investor a beneficial shareholder.

CrowdEquity Platform will provide all the stakeholders, such as project investors and islanders, with the opportunity to fund future projects in exchange for shares to the project or register their future projects to receive funding. The component will make the members of the community shareholders of the renewable energy assets, aligning with the community and Islander-centric approach that IANOS adopts.

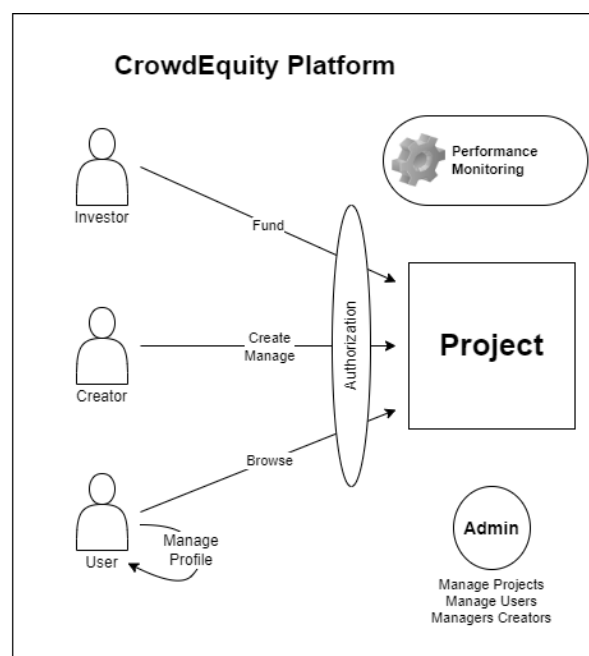


Figure 3-2: IANOS CrowdEquity Platform Main Use Case.

The component will allow transactions made with normal currency or tokenized energy based on the Distributed-Ledger based energy Transactions. It will include secure decentralized dApps, such as token wallets, in order to support the management of the end-user portfolio, and the processing of the transactions. The oversight of the end-user portfolio and the transactions is also going to be supported by visual analytics.

3.1.3 System Modeller

The Energy System Simulator (ESSIM) is a discrete-time simulation tool and collection of models that calculates energy flows in assets and the effects thereof, in an interconnected hybrid energy system over a period of time. With the help of the energy flows ESSIM calculates, one can get insights into how well the assets in a network are dimensioned, if there is overloading in any given transport asset (like pipe, cables, etc.) and what the effect of storage is in any part of the network.

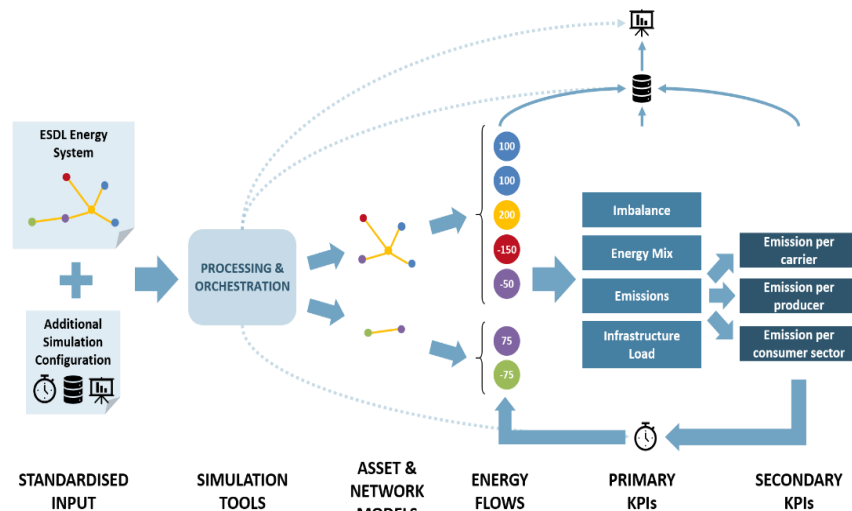


Figure 3-3: ESSIM.

Traditionally, dimensioning of assets was assessed by looking at yearly values of energy supply and demand. In a simple network where a producer supplied 10TJ per year and a consumer consumed 10TJ, the system, under such a calculation, is in balance and the energy producer is properly dimensioned to meet the demands of the consumers in the network.



Figure 3-4: Left: A system that appears to have no net yearly energy imbalance; Right: Hourly data showing imbalance over the year.

However, in reality, the producers do not produce a constant amount of energy every hour of every day throughout the year. Neither do demands have a constant flat line consumption pattern. These variations, when captured as a profile, highlight the mismatch despite the annual sums adding up.

ESSIM helps the user identify such nuances as it repeatedly calculates energy flows per time step for a simulation period. It takes as inputs the energy system defined in the

ESDL specification language and calculates the optimal schedule of flexible producers and the effect of this schedule in terms of emissions, costs, grid load, etc.

3.1.3.1 Energy System Description Language (ESDL)

ESDL is a modelling language created to describe complete (hybrid) energy systems in one uniform data format. It provides tools for the description of the individual components of the energy system, namely how they are connected, how they are utilized (energy consumption or production profiles), where they are actually located (on the map), how much they cost (in current or future time). Additionally, it allows a description of the information such as the energy potential and KPIs of areas, buildings of an area, or any asset).

ESDL can be applied to publish open data on energy systems or assist the interoperability between different energy transition models.

In the ESDL model, the components of a system are described by their basic functionalities, named *Energy Capabilities*. Five abstract categories are used to model the functionalities, namely Production, Consumption, Storage, Transport and Conversion.

The language provides energy modelers with a generic way to model complex energy systems. It is a machine-readable language that supports the development of GIS (geographic information system) applications and energy transition calculation tools with the intention to implement interoperability of their products.

Typical usages of ESDL are:

- In energy transition calculation tools, it can be used as a common language for energy transition calculation tools in order to describe the inputs and outputs of those tools. This allows for integration of multiple tools.
- In Energy Information Systems, it can be utilized as the core of a central energy information framework where the energy system of a certain region is registered.
- It can be used as a language, with which (local) governments model and communicate their (local) energy system.
- Multiple ESDL snapshots of a certain area over time can be utilized to provide insight and monitoring of the evolution of an energy system.
- Finally, ESDL can be used as a format to share information related to the energy transition or generally to the energy systems. Examples:
 - CO2 emissions per energy system
 - Technology factsheets for specific components, brands, types of assets (e.g. a heat pump factsheet that defines its standard parameters)
 - Cost information or expected cost developments of assets
 - Standard configurations or templates of standard parts of the energy system (for example a house with solar panels, a heat pump, and an EV charging station)

ESDL is released open source under the Apache 2.0 License. More information about ESDL can be found at its documentation page <https://energytransition.gitbook.io/esdl> and data model reference site at <https://energytransition.github.io/>.



IANOS benefits from the use of ESDL as a common energy system definition, as it allows the LCA/LCC, CBA and ESSIM toolset in the IETP suite to share a common view of the different islands' energy systems that are analysed. When using the same input, their outputs are easier relatable and comparable.

To support the end user, there are several tools available to create, view and edit ESDLs for all kinds of use cases:

- ESDL MapEditor is Google Maps on steroids for energy modelers. The ESDL MapEditor has been developed to allow one to create, view and edit an ESDL energy system description by just dragging and dropping energy system components on a map.
- ESDL bindings to Python and Java. This allows programmers to create their own software that is compatible with the ESDL language.
- Advanced simulators for heat, gas and electricity.
- A first implementation of a public energy data hub, named Energy Data Repository (EDR), where detailed information can be found on several aspects of the energy transition. Some examples include energy consumption profiles and technology factsheets. An API to access this information is provided by the EDR.

3.1.3.2 ESDL + MapEditor + ESSIM

Thanks to the easy configuration of the energy system with the help of the ESDL MapEditor, the user can use ESSIM to perform "what if?" scenario analyses on current and future energy systems. Along with the primary KPIs (Key Performance Indicator) of ESSIM (Energy mix, network imbalance, emissions, etc.), external KPI modules connected to ESSIM also allow for post-processing ESSIM data to get measurable insights into the energy system variations.

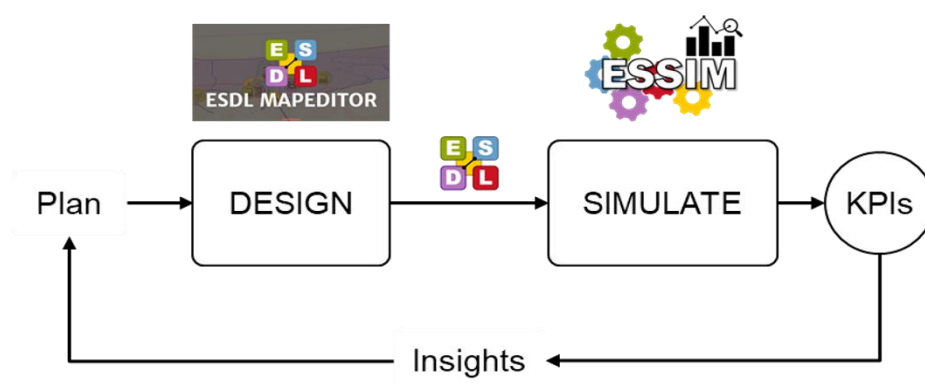


Figure 3-5: ESDL MapEditor provides a geographical front-end to design new scenarios for ESSIM simulations.

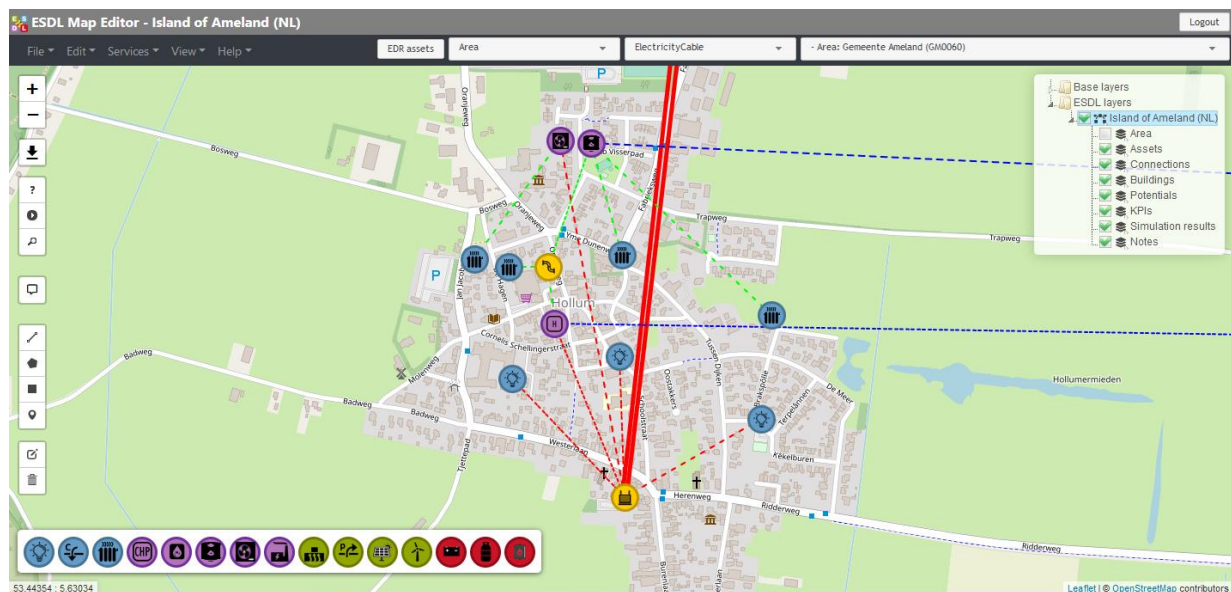


Figure 3-6: The ESDL MapEditor describing a piece of the energy system of Ameland, used as input for an ESSIM simulation.

At the heart of the tool are:

- A simulation engine that provides an evaluation in fixed discrete timesteps of each service network's balance in a fixed order,
- An algorithm that matched demand with supply based on flexibility. The algorithm uses the marginal costs of energy production as a tool to grade suitability of producers,
- A tree-based transport network solver used in the calculation of the load on various transport elements based on the results of the aforementioned algorithm.

A time-series database (InfluxDB) is used to store the data outputs generated during the simulation, and a dashboard (Grafana) is generated to visualize the results. ESSIM allows for connecting external asset models (via an MQTT interface) and external network (solver) models (via a REST interface), alongside its internal models and while using its own orchestration mechanism. This makes ESSIM a viable candidate for a co-simulation orchestrator.

The tooling is open sourced and for everybody available at GitHub:

- Downloading and deploying the tool suite on your local PC or cloud infrastructure <https://github.com/ESDLMapEditorESSIM/docker-toolsuite>
- MapEditor and ESSIM tutorials: <https://github.com/ESDLMapEditorESSIM/essim-tutorials>
- ESDL MapEditor documentation: <https://esdl-mapeditor-documentation.readthedocs.io/>
- ESSIM documentation: <https://essim-documentation.readthedocs.io/>

3.1.4 Grid-Oriented Optimizer - INTEMA.grid

The INTEMA.grid component belongs to the Island Energy Planning and Transition Suite (IEPT) of IANOS. INTEMA.grid is a grid modelling and simulation tool, able to support

long-term planning, while it can evaluate several energy management and operational strategies, accounting for multiple RES and storage integration scenarios as well as fostering decarbonization of current energy mixture. It also can promote energy networks synergetic operation. The tool can perform the following calculations, for any grid topology, including electrical, heating/cooling, gas networks and storage solutions:

- a) Power flow calculation at each grid node
- b) Optimal power flow calculation, minimising the generation cost at each timestep
- c) Frequency stability study, which requires the comprehensive electromagnetic transient (EMT) modelling of each asset
- d) Contingency plans analysis (N-1 criterion, short-circuit etc.), accounting also for forecasting algorithms (in real-time, if needed)
- e) Renewables power generation and load

Besides plots of the grid response, the tool is also able to calculate specific aggregated energy-related KPIs, as defined in D2.3.

3.1.5 CBA component

The Cost Benefit Analysis (CBA) component belongs to the Island Energy Planning and Transition Suite (IEPT) of IANOS. The objective of this component is to create a holistic tool able to assess the overall benefits expected from the green energy/ smart grid interventions in the demonstrators of IANOS. The CBA tool will be based on the JRC's and ENTSO-E's CBA methodologies and will offer an analytical approach for the stakeholders/ investors that provides a quantifiable insight regarding whether a smart grid intervention exceeds the existing baseline scenario in terms of cost and benefits. To do that and depending on the activities and vision of the various stakeholders, the factors of interest will be selected and respective KPIs will be calculated to assist them in setting strategic priorities, aligning horizontally in all cases with the priorities set by the Green Deal.

As the cornerstone of the CBA component is considered the CBA methodology proposed by the European Network for System Operators (ENTSOe). This methodology strives to assess smart grid projects rigorously, objectively, and consistently by not only considering the benefits on an economic basis, but also incorporating environmental and social impacts. The CBA is based on three pillars: Benefits, Costs and Residual Impacts. The complementary components of IANOS IEPT will be leveraged in order to calculate those indicators, by utilizing the KPIs defined in T2.3. In addition, the CBA component has the capability of providing CBA results for each demonstrator and the involved stakeholders in order to justify the benefits of the smart grid intervention for various perspectives.

Both JRC's and ENTSO-E's methodologies are comprehensive frameworks to assess costs and benefits of Smart Grid Interventions inserted into the electricity infrastructure from the system planning perspective. The analysis accounts at same time for technical, economic, reliability, environmental, and security benefits. This framework aims at evaluating its effectiveness in the IANOS project demonstrators, i.e., LH islands of Terceira and Ameland, and the defined UCs.



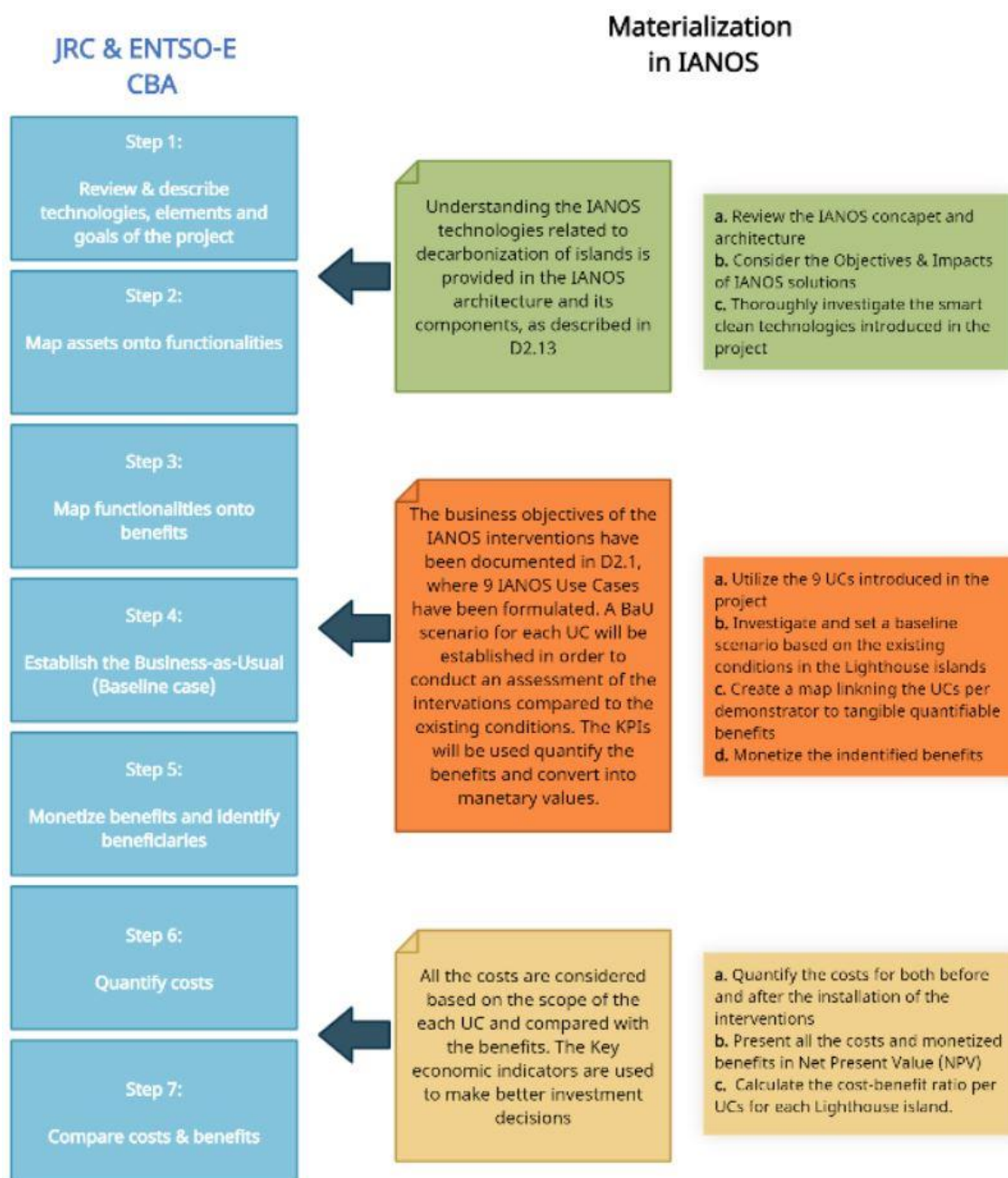


Figure 3-7: High level diagram of the position of CBA component in the IANOS Energy Planning & Transition Support Toolset.

The economic indicators that are leveraged to perform the CBA are introduced. Firstly, the Discounted Cash Flow (DCF) model is introduced. This model is used to compute the present value of future cash flows, resulting from Smart Grid Interventions for each year of the analysed period. The DCF model is a basic evaluation model for an asset that is expected to generate cash payment in the form of cash earnings, interest and principal payment, or dividends.

After the computation of the DCF for each year, the Net Present Value (NPV) for each benefit is computed. The NPV of a capital budgeting project is the monetary amount of the change in the value of the firms because of undertaking the project. For the case of the

IANOS project, the NPV depicts the monetary amount of the change in the value of the energy infrastructure because of selecting and applying one or more Smart Grid Interventions, according to the business objectives of each introduced IANOS UCs. A positive NPV value showcases that the value of the infrastructure will increase if the new interventions will be adopted. On the other side, a negative NPV means that the infrastructures' value will decrease if the new interventions will be adopted.

The selection of time horizon is of vital importance and has to be defined within the benefits and the costs of each intervention. The time horizon depends on the asset type of each installation, the investment scheme and the benefits for the overall electricity infrastructure (and in general energy infrastructure if benefits can be found in the rest energy carriers).

According to ENTSO-E's CBA methodology, the KPIs have been grouped in 3 distinctive categories, i.e., Benefits, Costs and Residuals, based on which type of impact each KPI can quantify. Each UC of the IANOS project is linked to specific KPIs that quantify the business impact of the interventions. The KPIs that will be used as the outcome of the Energy Planning & Transition Decision Support suite, will be computed using the functionalities of the components that constitute the suite. The following figure provides the position of the CBA component in the suite, and how it uses the outcome of the rest components, i.e., the calculated KPIs values, in order to conduct CBA for the IANOS solutions. The benefits column represents the categories, as identified in the ENTSO-E's CBA methodology. During the pre-validation and refinement activities of the IEPT tool (related to task 3.4), a cross-checking process will be conducted to provide a list of specific benefits for each UC, updating the illustrated ones.

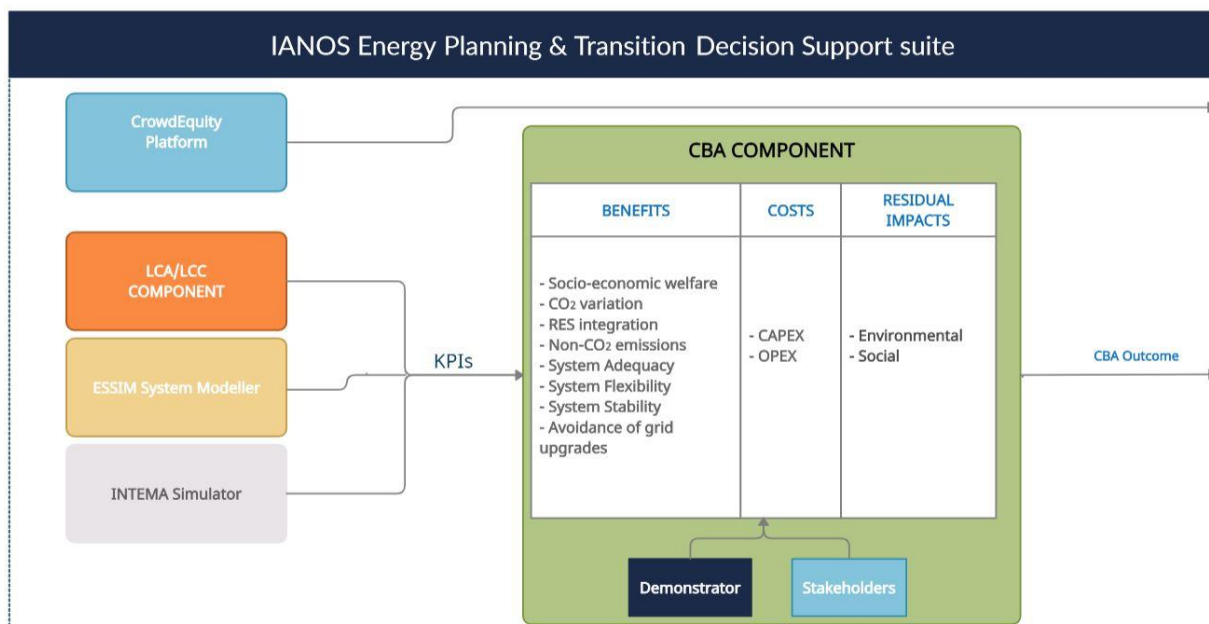


Figure 3-8: High level diagram of the position of CBA component in the IANOS Energy Planning & Transition Support Toolset and its interconnection with the rest components.

The role of the CBA component is to convert into monetary gains (if possible) all the impacts of the benefits for each UC, and to provide a holistic tool for investment planning for the different stakeholders. In case a benefit/ residual cannot be directly converted into

monetary gains, a scaler is introduced in order to provide an auxiliary tool to the decision-makers by showcasing the improvements/ positive impact that a specific IANOS intervention will make.

3.2 Decision Making Layer

The Decision Making Layer, as its name suggests, will contain all the necessary components to optimally plan the dispatch of renewable energy assets. It represents the multi-level decision making intelligence that drives the iVPP Operative Orchestration Toolkit. The iVPP will be responsible for establishing a virtual network of decentralized dispatchable and non-dispatchable renewable energy generation units and energy storage systems, with the aim of promoting the concept of self-consumption. The decision-making process will be complemented by predictive algorithms for optimal use of grid assets.

3.2.1 *Aggregation and Intelligent Segmentation*

Data aggregation is the process of gathering data from multiple heterogeneous sources with the intention of combining these data into a summary for data analysis through fusion techniques. By performing data pre-processing and transformation techniques on the raw data, the volume of the data can be reduced and in addition great insights of significant importance to utilities can be extracted and provided.

Segmentation is a data mining process, where valuable information is extracted by segmenting objects and creating groups with similar features. It can be a useful tool to support the operator's portfolio characterization based on the relevant objectives and strategies. In general, segmentation will assist and provide support to the operator either for formulating market strategies or for exploiting the results for a demand-side management approach. More specifically, in IANOS, the segmentation module will consist of two sub-modules, the classification and the clustering modules. Classification is a supervised learning technique used to assign labels to objects by creating a model based on previous knowledge. In the context of IANOS, this sub-module will classify new prosumers into groups that have been created via the clustering sub-module, in order to assist the aggregator with the creation of demand-side management policies. Clustering will be the main sub-module of the Segmentation module. Clustering is an unsupervised machine learning technique, in which objects are assigned labels and grouped without any prior knowledge. Using clustering, three objectives are going to be addressed 1) the creation of prosumers groups for demand response programs, 2) the provision of data analytics services to the aggregator, 3) assistance with forecasting.

3.2.2 *Forecasting Engine*

This component will provide the necessary inputs to the decision support system by developing forecasting algorithms for both consumption and generation units, contributing to the optimal scheduling and planning of the grid assets. In addition, for



the Ameland Pilot Island, energy market forecasting models will be developed, more specifically; predictions will be made for day-ahead price, intra-day price, imbalance price and Frequency Containment Reserves (FCR).

The forecast horizon will be extended from a short (short-term) to a daily level (day-ahead) using historical consumption/generation/price data and external information (temperature, humidity, wind speed, total day ahead energy generation, etc.) as well. The day-ahead forecast module will be executed every midnight by delivering forecast slots for the whole next day, while the short-term forecast will be triggered at specific time intervals during the day in order to provide more accurate results. Eventually, regarding the development and the training of the models, various techniques will be deployed and examined, including statistical and advanced machine learning techniques such as ensemble methods and deep learning.

3.2.3 Centralized Dispatcher

The Centralized Dispatcher is the main decision-making module of the system. It is responsible for the optimal scheduling of the dispatch of both large-scale and small-scale grid assets. The operation of the Centralized Dispatcher is separated into two main categories. **Before the dispatch activation**, it is responsible to optimally plan the use of the assets. More specifically, it receives inputs from the other components of the iVPP, e.g. from the Forecasting Engine, the Aggregation and Classification, the Flex Forecaster and external services, and it aims to output the optimal dispatch of the assets for different time intervals, either day-ahead or intraday. The latter is going to be achieved through the optimization of the flexibility derived from the data of the multiple energy markets and services. In Terceira, the services will be implemented utilizing the KIPLO core platform and the optiMEMS tool, while in Ameland, the ReFlex software solution is going to be utilized. These components are going to be further analysed in the following chapters.

At the dispatch activation stage, the Centralized Dispatcher is responsible to implement the result of the optimal planning by adapting the decision-making logic, taking into account the most recent system conditions (in terms of assets availability and production/consumption deviations from the planned profiles). At this stage, the module will support the decisions made by its subcomponents and will send the corresponding dispatch signal to the dispatchable assets into the iVPP portfolio. This would correspond to the delivery of the optimal demand, supply, and storage side set points for grid services provision and self-consumption.

In the sections below, the main functionalities of the core decision making platforms (ReFlex, KIPLO, OptiMEMS) used into the Centralized Dispatcher are described.

3.2.3.1 ReFlex

ReFlex is a software solution that empowers aggregators to create a powerful Virtual Power Plant (VPP). This VPP can utilize the flexibility of large quantities of (small) devices effectively for multiple purposes. Enabling the flexibility of DER assets, such as electric



cars, batteries and solar panels, to be utilized in both energy markets and ancillary services markets. In this way, the value of flexibility can be stacked, and the profits of utilizing flexibility increased.

In order to optimize the utilization of flexibility, it is necessary to know what the flexibility of the cluster is. Since there are typically too many devices, of different types, it is not possible to consider each device individually. Instead, the flexibility of the cluster as a whole needs to be considered. The owner of a VPP typically wants to know how much it can ramp up or down at a certain point in time, and what the consequences are of dispatching up or down (since there can be a rebound effect for certain types of flexibility). ReFlex provides that insight.

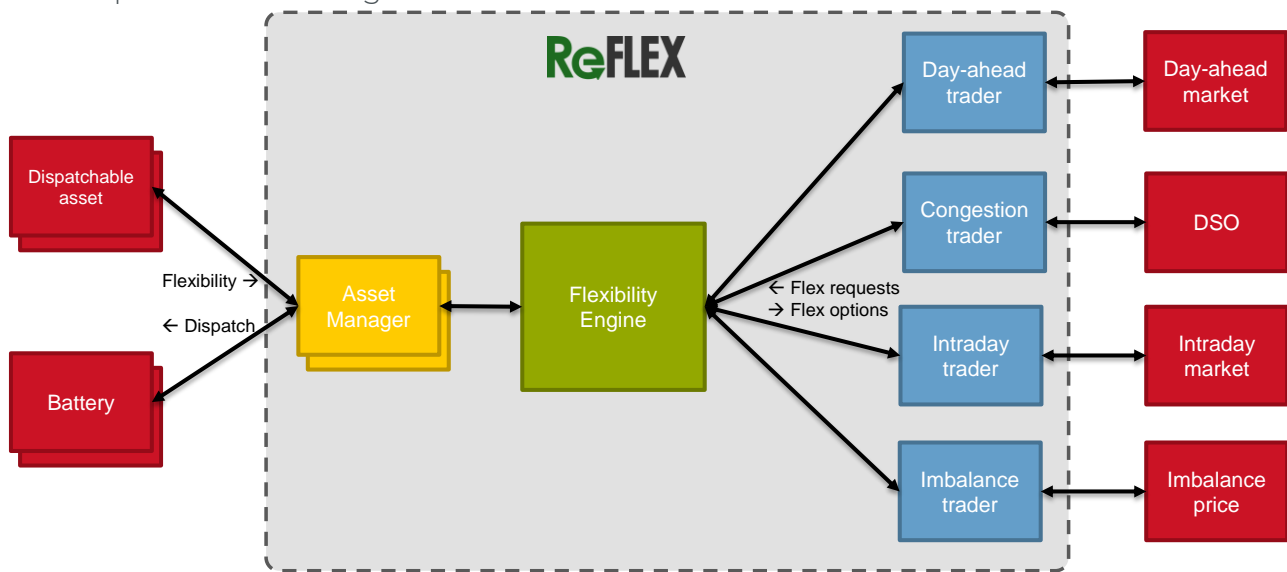


Figure 3-9: Reflex.

ReFlex works with a moving planning window. Typically, ReFlex plans the energy production and consumption for every device for the coming 48 hours with a precision of 15-minute intervals (sometimes referred to as PTU's or ISP's). Each interval has a fixed start and end time, and the first one of the day always starts at midnight. In order to plan the energy consumption and production of each device, ReFlex needs to have knowledge about the future behaviour of the devices. This typically requires forecasts of the behaviour of the devices to be made. ReFlex always has a target for the sum of the energy consumption and production of all the devices in the cluster for each interval. This means that the default behaviour of Reflex is to optimize for self-consumption. The flexibility of the devices can be utilized by changing this target. When the target changes, ReFlex will try to adjust the behaviour of devices in cluster in such a way, that the sum of energy comes as close as possible to the target. Changing the target can also be simulated. In this way, the rebound effect of dispatching flexibility (by changing the target) can be made explicit.

In addition, to change the target for the sum of energy of all devices, it is also possible to put in energy constraints for intervals for a subset of the cluster. In this way, it is possible to take (local) grid congestion constraints into consideration. By placing a constraint on a subset of devices, ReFlex will do its best to respect those constraints.

3.2.3.2 OptiMEMS

OptiMEMS is used to monitor multi-source network power systems, where advanced energy management strategies are deployed. Except for the optimal scheduling of the small and large-scale assets of the grid, the decision tool is responsible for the maintenance of the power balance within the grid, the self-consumption and grid services provision in emergency cases. The main objective of the tool is the minimization of the overall daily cost by optimizing the flexibility of the energy portfolio and extracting the optimal set points in terms of supply/demand/storage. At the current state, OptiMEMS consists of three main sub-components: an optimization engine that provides the optimal scheduled program for the next day by satisfying the load demand in each time slot, forecasting mechanisms for consumption and generation units within the grid and a real-time tool that validates the possible deviations between the actual measurements and the planned for the day-ahead. The day-ahead forecasts are complemented by short-term forecasts that are triggered in specific time intervals. Therefore, when a deviation occurs, the validation sub-component is responsible for the rescheduling the plan based on projected short-term forecasted results. The internal system architecture of the optimization component is depicted below:

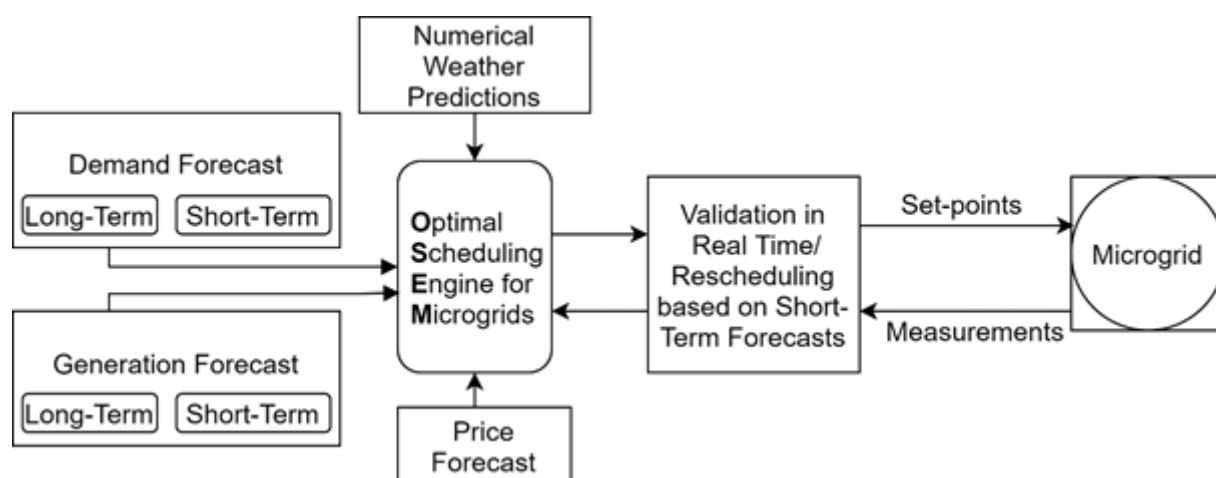


Figure 3-10: OptiMEMS internal system architecture.

3.2.3.3 Kiplo Core Platform

Kiplo Core Platform is integrated into the Cleanwatts (CWD) platforms that are being developed with the main purposed of helping utilities, aggregators and users in the management of their energy assets and the associated flexibility. By monitoring and managing accurate real-time data from demand and supply side, it allows the optimization of available energy resources using Demand Response activities, flexibility management, connection with upstream markets and the possibility to enable P2P energy markets. Kiplo Core Platform is constantly updating the inputs to dynamically optimise the global operation of the Virtual Power Plant (VPP).

At the current state of development, it can connect to a wide range of sensors and actuators (IoT devices), using several communication protocols, and allowing seamless third-party integration via API and web services. Kiplo Core Platform is equipped with an API that allows easy integration of external advanced services (load and RES forecasting, clustering, and storage optimization) and management of novel types of equipment (e.g., flywheels, hybrid transformer, heat storage). It is integrated within a global design with a tiered and service-oriented architecture, allowing an easy expansion and integration (adding new modules or services) and interoperability (upgrading or replacing some modules).

Kiplo Core Platform will be a core platform, performing the data collection from the ESB, analysis and processing of certain data, storage and serving as communication exchange platform between several of the different iVPP components. At the level of the Centralized Dispatcher (CD) component, besides the inputs from external components, IoT devices and energy assets, it will also have the inputs from the “Aggregation and Classification”, “Forecasting engine” and “optiMEMS” to consolidate the dispatching setpoints.

The general platform architecture for the CWD platforms is shown on the diagram below. Kiplo Core Platform is integrated within this architecture, which will be used as interface for other iVPP components, being a core platform for the Terceira Island pilot.

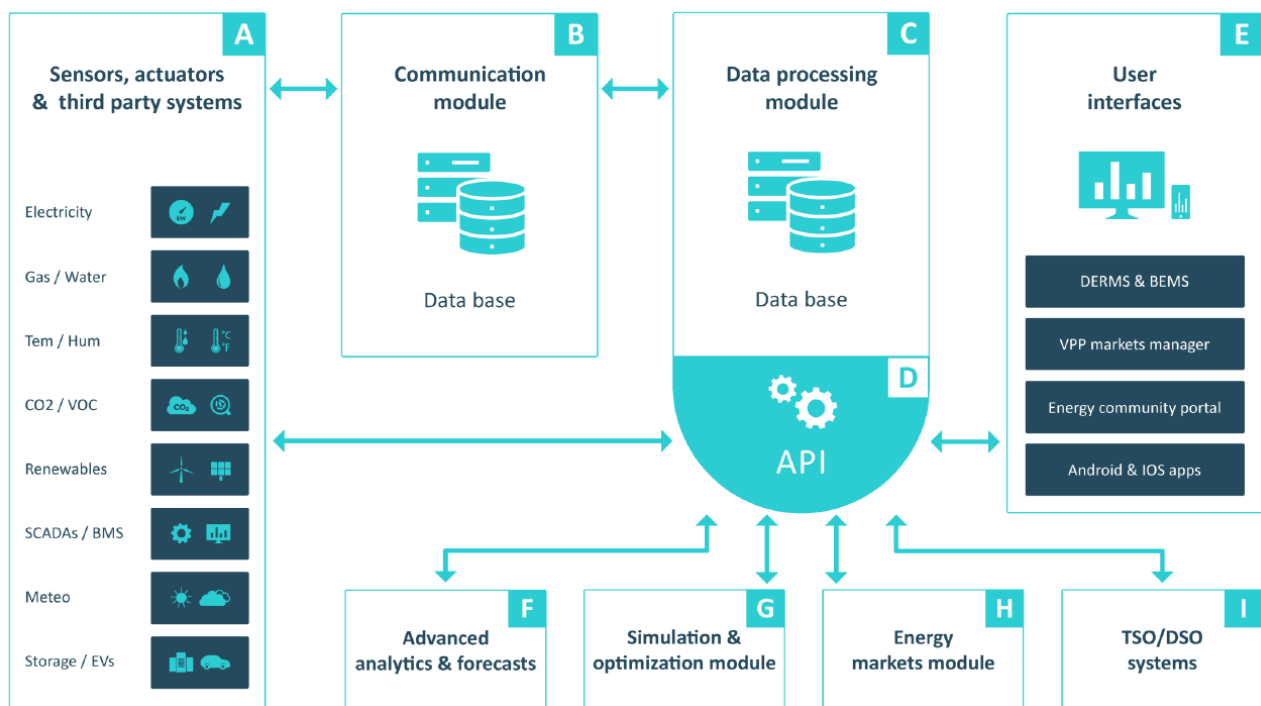


Figure 3-11: General platform architecture for CWD platforms.

3.2.4 DLT-based Transactive Platform

The DLT-based Transactive Platform aims to develop a P2P market that enables prosumers in a local network to directly trade energy with each other, by avoiding RES curtailment and future grid transport costs. The P2P trading system is based on

blockchain technology that guarantees the transparency and security of the transaction, which remains permanently recorded in the platform, allowing all parties to audit the results. Fungible tokens based on the ERC-20 standard will be exploited as a payment for the purchase of energy between prosumers.

The main advantages of the DLT component are:

- **Neutrality of the market:** the absence of a central owner of the market guarantees the satisfaction of all stakeholders' interests without any catalysation around big players. Moreover, the adoption of an open mechanism for price calculation (everybody knows the algorithm) is another way to engage prosumers by reducing their concerns about system fairness.
- **Trustworthiness - transparency and immutability:** by design, the transactions in the blockchain are transparent and immutable. This can help the resolution of any disputes among the market participants and ensure non-repudiation for transactions performed.
- **Self-enforceability of smart contracts:** self-enforceable means that once the smart contract is configured on and running, the execution of its code is automatic and will not require a specific approval. In the proposed system, the transfer of tokens between prosumers is performed automatically by a smart contract after the validation of the energy transaction. No central authority can interfere with the transaction.

Figure 3.11 shows a representation of how the P2P market works.

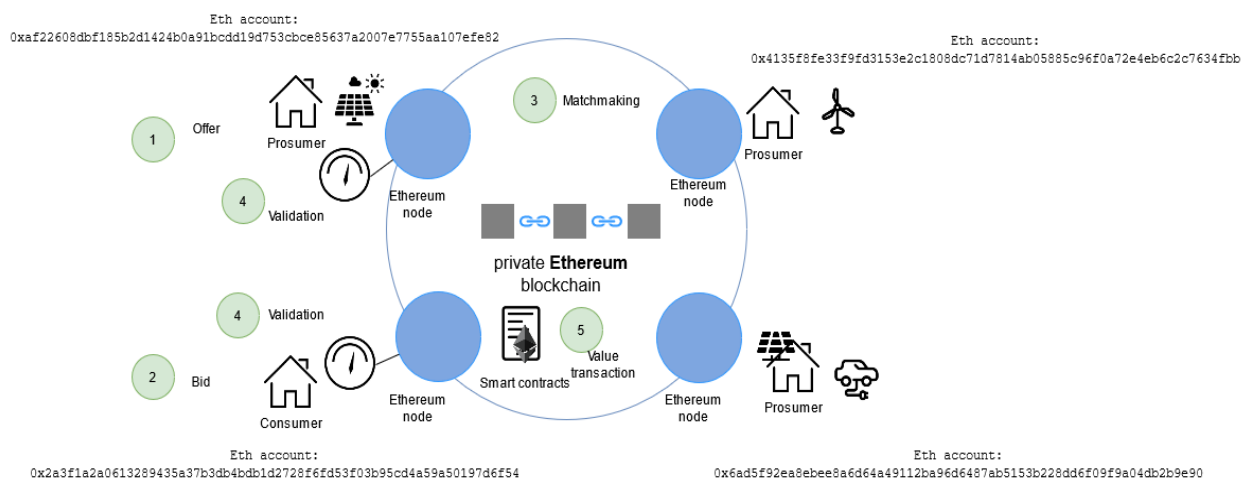


Figure 3-12: Local P2P energy trading based on blockchain technology.

The blockchain infrastructure provides the use of a private network, capable of supporting programmable smart contracts. The reference platform will be based on Ethereum. Prosumers and consumers are active participants of the network by owning their own Ethereum node. Each node has an associated account in the blockchain that can be used to sign transactions and must ensure a connection to the smart-meter. The P2P market collects bids and offers from participants. Each market action refers to an amount of energy and a price for a specific time slot. At the end of each market session, bids and offers are collected and matched together. The final price and quantity are determined adopting the clearing price algorithm. The consolidated market actions are

permanently stored on the private blockchain, additionally, it is also used to store the - timestamped- measures of the grid power consumptions, provided by smart meters. These measures are used as reference levels to validate market agreements and verify that each participant acts as expected.

3.2.5 GOPACS

GOPACS (Grid Operation Platforms for Congestion Solutions) is a unique initiative in Europe that provides a platform to solve congestion problems in the electricity grid. It is a result of active collaboration between the Dutch national grid operator (Transmission System Operator, TSO) TenneT and the regional grid operators (Distribution System Operators, DSOs).

GOPACS is developed to work in a consistent way, regarding the key European directives that are related to the market-based mitigation of grid congestion and offers both small and large parties involved in the market a simple means to generate revenues and contribute to resolve congestion situations, using their available energy flexibility.

Additionally, it assists the grid operators, due to their collaboration, to avoid causing problems in the electricity grid, when a congestion occurs elsewhere at one of the other grid operators. With GOPACS, grid operators currently collaborate with intraday market trading platform ETPA.

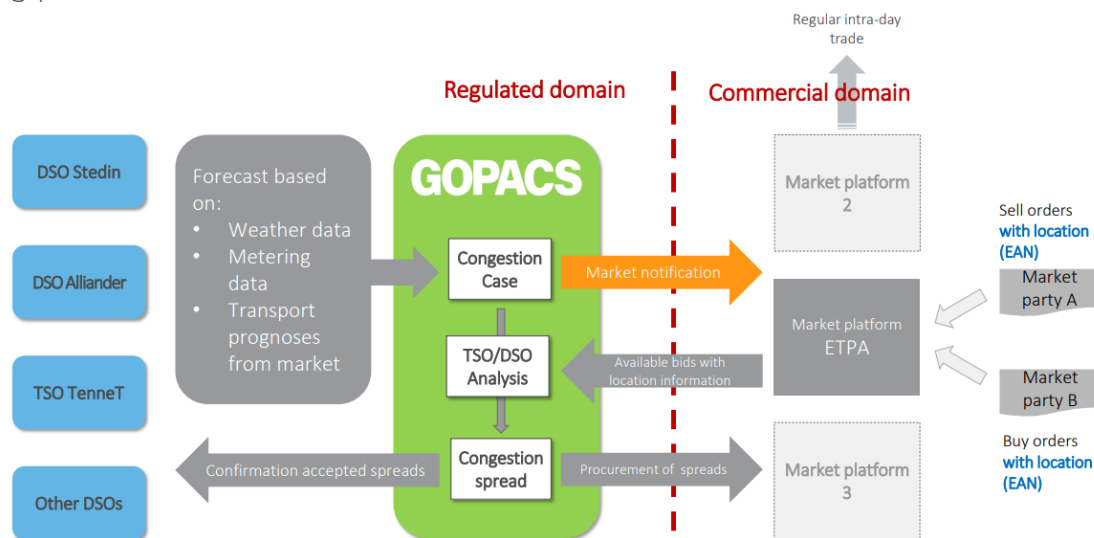


Figure 3-13: GOPACS: Grid Operator Platform in the Netherlands for market-based congestion solutions.

3.3 User Interface Layer

User interface layer, as the name suggests, will provide the system's operator with the appropriate interface for the definition of the decisions made in the decision layer by the other modules, as well as manual controls. The sole component of the user interface layer will be the Virtual Energy Console, described in the following chapter.

3.3.1 Virtual Energy Console

The Virtual Energy Console is an interoperable user-friendly monitoring console (UI - User Interface) to effectively assess energy flows for the VPP operator to properly manage, visualize and dispatch their energy assets. It will provide the market with a complete and innovative dashboard that will be tested and validated in real world conditions.

The Virtual Energy Console will be specially designed according to the requirements of IANOS project. The dashboard will allow the VPP operator to easily access different dataset and important information in line with IANOS KPIs, such as generation mix of the VPP portfolio, penetration of RES in the system and historical data. It is based on the existing CWD Kiplo platform User Interfaces (UI) that will be expanded to include the necessary functionalities associated with the intelligent VPP operations including: i) Multiple Data Fusion, ii) Monitoring and Profiling, iii) Aggregation and Classification). In relation with them the Virtual Energy Console will be designed to embrace relevant advances in relation to the various services to be offered. It will use innovative visualization and visual analytics methods and tools, to provide the operator with an intuitive environment, where the operator can quantitatively assess both the current operational capacity and running VPP operations.

The Virtual Energy Console will include the necessary innovations on visual and data analytics, enabling the dynamic connection of different datasets with several types of visualization, so that user selection in one visualization feature can have a direct impact on the others. The indicative information of-value, to be visualized through this component will entail i) composition summary (mix) of VPP portfolio of the units, ii) monitoring of dispatchable vs vRES installed capacity, iii) actuation of energy assets and iv) a diagnostics history.

In terms of communication with the various grid assets, the platform will exchange information and signals with the IoT devices and energy assets mainly using secure and standard TCP/IP and MQTT protocols. Other protocols and API support will be added to the platform to assure interoperability among all market vendors. These protocols are to be defined during the integration project phase.

3.4 Communication Layer

In the communication layer, all the necessary components for the communication between IANOS iVPP platform and the numerous geographically scattered units are found. The components will enable the secure monitoring of the exchange of data from the renewable energy assets and other field components, as well as deliver the appropriate setpoints set by the Centralized Dispatcher.

3.4.1 IANOS Secured Enterprise Service Bus

The IANOS iVPP framework functionalities and energy services require the exchange and appropriate handling of data and control commands among the different system



components and field devices. This number of field-level heterogeneous data flows will be ingested in the iVPP by means of an enterprise service bus. The iVPP secured enterprise service bus (ESB) will be the component that will play this data transfer role, with a special focus on cybersecurity aspects.

Interoperability is seen as the key enabler of smart grid. A prominent definition describes interoperability as “the ability of two or more devices from the same vendor, or different vendors, to exchange information and use that information for correct cooperation”. In other words, two or more systems (devices or components) are interoperable, if these “two or more” systems are able to perform cooperatively a specific function by using information which is exchanged. This concept is illustrated below:

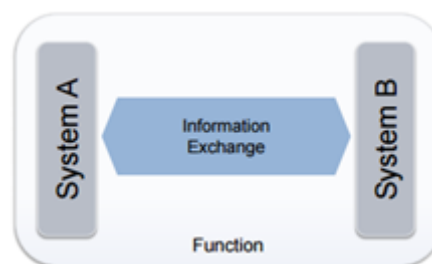


Figure 3-14: Interoperability definition.

Interoperability can be categorized into eight levels, from basic physical connectivity to a full alignment of business, policies and goals:



Figure 3-15: Interoperability categories.

The different layers are focusing on specific interoperability aspects, from the most basic connectivity to business goal alignment:

- The lower layer refers to the physical connectivity of the assets.

- The 'network' and 'syntactic' interoperability is basically reached by agreeing on communication protocol (E.g. REST Web Services, MQTT, PLC, etc.)
- Semantic and business context interoperability is focusing on the pieces of information exchanged, and tackles the selections of the appropriate data models to wrap the information messages.
- The upper layers are focusing on the functional alignment of components and comprises things like interoperability in the orchestration of messages, sequence of messaging, event handling, etc.

For some of the interoperability layers there are sometimes standards widely used, which clearly define the interaction (especially for the communication protocols and data models). In this case, normally it is worth adopting the standard at both sides of communication. On the other hand, if such a standard for the communication does not exist, one has to be selected and adapted or even a specific ad-hoc mechanism should be defined. In any case, both parties to the communication must 'speak' the same language and give the same meaning to the information exchanged.

In modern, decoupled, smart grid control systems, a set of different applications (potentially from different providers) are deployed for applying specific tasks. IANOS will not be an exception to that, and the different modules will be provided so that they can be 'plugged' or not to an installation or even replaced by other existing commercial software with few changes. Furthermore, the field devices in the different pilots will be different (and compatible with different protocols, mechanisms, etc.) and thus a huge number of potential interconnections can be found while deploying IANOS components. Furthermore, any new component may imply on developing or adapting several connectivity solutions to get full connectivity. This is even worst considering that legacy-monolithic distribution systems are not well suited to be extended with new functionalities offered by other modules, and therefore specific adapters and gateways are needed everywhere to maintain a coherence in data and business operations. This approach can become chaotic in small-to-mid size systems.

The solution proposed is to start using a common model for information exchanged. With a common information model, different applications communicate each other using the same language. This approach reduces the number of data transformations required to N from $N * (N-1)$. Integrating legacy applications typically involves the creation of application wrappers that map legacy data formats to a common one:

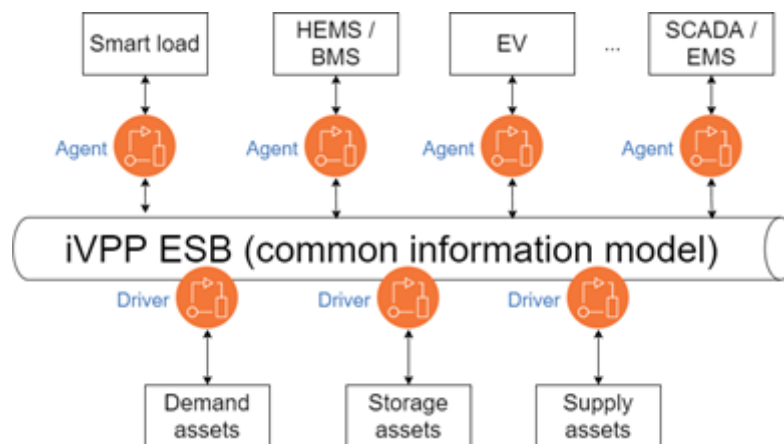


Figure 3-16: Common data exchange format interoperability.

The field devices will be interconnected by drivers in the ESB, featuring the required communication protocols and transforming the data to the common information model in the ESB.

On the other hand, the different components in the VPP orchestration toolkit connect to the iVPP ESB by using gateways that adapt from the specific data model and the common information model.

This model will allow IANOS to be extended and support multiple field device types, and at the same time, the intelligent agents and applications in the VPP will be able to work without worrying about the details (connectivity, data models, etc.) of the specific field devices in each pilot or scenario.

3.4.2 dEF-Pi

dEF-Pi is an open platform on which energy demanding, producing or storing assets can be enclosed and transfers asset-specific data to a standardized S2-interface (EN 50491-12-2). The goal of dEF-Pi is to provide a runtime environment that makes it possible to quickly design and implement services dealing with energy management. DeF-Pi assists in resolving the problems that arise when deploying services on separated nodes, by supporting more than one node that is hosting instances of services. One of the main principles of the platform is scalability, which is reflected in the lightweight of the platform in order to handle large amounts of nodes and, subsequently, a large amount of service instances.

Another key principle is the robustness of the platform, since when the amount of nodes or services increases, the changes of failures increase exponentially. Making sure that nodes that become unresponsive are handled correctly, thus errors are contained and do not cause errors elsewhere in the platform. Because of the distributed nature of dEF-Pi, security is also an important fact in the platform. The central component is the orchestrator, which is responsible for the deployment of services and the initialization of connections between instances of services. The orchestrator exposes an API that is used by the web-based user interface. This approach allows third-party developers to incorporate the orchestrator by using the API. Services are stored in a central registry. The services are stored as Docker images, with additional metadata attached to the images.

that describe the interfaces that service is able to receive connections. The description of the interfaces occurs via either Protocol Buffers (protobuf) or XML Schema Definition (XSD), in this way, the compatibility of interfaces is ensured when two interfaces have compatible definitions. Nodes in the environment are hosts that are capable of running the Docker images as containers. The nodes are connected to a Virtual Private Network (VPN) in order to have a secured private network in which the processes are able to communicate with each other, as well as the processes being able to receive instructions from the orchestrator.

3.4.3 Back-end Agent

Back-end Agent is the core component of the iVPP platform and serves as a link between the platform's other components. It is especially responsible for the daily execution of the various iVPP components, as well as monitoring and ensuring their proper operation via logging systems. The back-end agent concept, as well as the various components that comprise the iVPP platform, is based on microservices logic, with each component running independently. Thus, REST techniques are used to facilitate communication and interaction between the iVPP components and the back-end agent. An event scheduler deployed within the back-end agent orchestrates the daily operation and inspection of procedures that need to be executed at specific times during the day.

3.5 Physical Layer

The physical layer includes the various physical assets of IANOS project. The different assets were analyzed in D2.2, "Report on Islands requirements engineering and Use Cases definitions" for each pilot island. In this document, a more detailed presentation of the non-intrusive characterization and use of energy flexibility in water heating systems is provided.

3.5.1 *Non-intrusive characterization and use of energy flexibility in water heating systems*

UNINOVA's non-intrusive characterization and use of energy flexibility in water heating systems is made from a set of sensors coupled and installed to a classical water heater without the need to change or modify it. The information from the operation of the water heaters is collected and then passed through a microcontroller. Then, it is communicated wirelessly to the servers of UNINOVA and through them to the iVPP. High-level instructions will be provided from the iVPP, which will be passed to UNINOVA's servers. From there, the instructions will be delivered to the individual microcontrollers that will control the water heaters. The components of the system in detail are:

- a) A set of sensors which gets power and temperature data, with minimum intrusion to the existing user's equipment

- b) A microcontroller with Wi-Fi communication which receives control signals and sends and gets data
- c) An actuator that makes the supply of power to the heating element possible
- d) A remote system of computation and control where the energy flexibility characterization and control strategy are computed and the communication with the iVPP is made possible



4 System Requirements

In this section, the functional and non-functional requirements of the system are provided. For each functionality of the corresponding component, the appropriate requirements are provided in the following tables, as well as the priority of every requirement, a general description of the functionality, and the associated use cases.

4.1 Functional Requirements

The functional requirements provide descriptions of what the system must and must not do, thus providing the necessary features that describe the intended behaviour of the system. The following table presents the functional requirements of each component of the IANOS system.

Table 4-1: LCA/LCC Toolkit functional requirements

Name of component		Responsible Partner
VERIFY-D platform - LCA & LCC toolkit		CERTH
Functionality	Rationale	Associated Use Cases
Assesses the environmental impact and financial costs for new investments.	The energy communities, the prosumers and the stakeholders of the islands in general will need aid in the decision-making process for their investments. The LCA/LCC toolkit will offer insight for Energy Savings, Reduced Fossil Fuel Consumption, Reduced GHG emissions and primary energy demand and consumption related with new investments in RES and storage as well as all the projected costs.	UC 9
id	Functional Requirements	Priority
FR-VRF-01	Collect data on a real time scale (Every 30 or 60 min).	High
FR-VRF-02	Support analysis of multiple energy streams (Electricity, Gas, Waste, Heat).	High
FR-VRF-03	Detect data anomalies.	Medium
FR-VRF-04	Support back-end PostgreSQL Database.	High
FR-VRF-05	Provide long term environmental impact analysis.	High
FR-VRF-06	Support multiple data formats (.json, .xml, .csv)	High



FR-VRF-07	Support clustering techniques for consumer segmentation.	Medium
FR-VRF-08	Support seamless communication with IANOS Secured Services Enterprise Bus.	High
FR-VRF-09	Support HTTP REST services for data exchange and MQTT protocol for communication purposes with third parties.	High
FR-VRF-10	Get Static data regarding the technical characteristics of the technologies.	High
FR-VRF-11	Get the annual production and consumption needs.	High
FR-VRF-12	Get Dynamic Data regarding Electrical Production and consumption, Fuel consumption and number of prosumers/consumers.	High
FR-VRF-13	Store and encrypt the data.	High
FR-VRF-14	Support open access to the data via request.	Medium

Table 4-2: CrowdEquity Platform functional requirements

Name of component		Responsible Partner
CrowdEquity Platform		CERTH
Functionality	Rationale	Associated Use Cases
Provide an online platform for crowdfunding of new projects in exchange for equity.	Part of IANOS project is the involvement of the local communities in the planning and ownership of the new RES projects that are going to be created. For this purpose the CrowdEquity toolkit will be provided. This toolkit will provide an online platform for sharing ideas and business plans from investors while providing them to a source of funding via Crowd Funding in exchange for equity.	UC 9
id	Functional Requirements	Priority
FR-CEP-01	Secures the users' personal information.	High
FR-CEP-02	Support an online environment for project sharing.	High
FR-CEP-03	Provide visual representation and statistical analysis on the projects.	Medium

FR-CEP-04	Support transactions with both FIAT currency and DLT based currency.	High
FR-CEP-05	Communicate with a mobile application and token wallets.	Medium
FR-CEP-06	Communicate with the LCA/LCC and CBA toolkit.	High
FR-CEP-07	Support oversight of the user's portfolio and transactions.	High

Table 4-3: Cost Benefit Analysis Component functional requirements

Name of component		Responsible Partner
Cost Benefit Analysis Component		UBE
Functionality	Rationale	Associated Use Cases
Provide an online tool for conducting cost-benefit analysis.	Users of the IEPT toolkit will need to have a clear overview of the possible costs and benefits of their projects in order to decide on their realization. The CBA component will conduct this analysis and give helpful insights to the investors.	UC 9
id	Functional Requirements	Priority
FR-CBA-01	Communicate seamlessly with the other modules of the IEPT Toolkit.	Medium
FR-CBA-02	Store previous results and data for future use.	High
FR-CBA-03	Be openly accessible to every user.	Medium
FR-CBA-04	Support multiple data formats (.xml, .json, .csv)	High
FR-CBA-05	Perform analysis for multiple scenarios.	High

Table 4-4: Grid Oriented Optimizer (INTEMA) functional requirements

Name of component		Responsible Partner
Grid oriented optimizer (INTEMA)		CERTH
Functionality	Rationale	Associated Use Cases
Provide an online tool for preliminary power system dimensioning.	When planning for new investments in RES and Storage devices, their impact on the grid and the total power flow must be taken into account. For this reason, the Grid Oriented Optimizer will calculate the energy system's power flows at each node, the optimal power	UC 9

	flow, frequency stability, contingency plans renewables power generation and load.	
id	Functional Requirements	Priority
FR-GOO-01	Communicate seamlessly with the other modules of the IEPT Toolkit.	Medium
FR-GOO-02	Calculate multiple scenarios based on user input.	Medium
FR-GOO-03	Be openly accessible to every user.	Medium
FR-GOO-04	Support multiple data formats (.xml, .json, .csv)	High
FR-GOO-05	Perform calculations for short time scales.	High
FR-GOO-06	Has different modules for Power Flow calculation, optimal power flow calculation, Flexibility stability calculation, contingency plans, renewables power generation and load.	High
FR-GOO-07	Functions for all the different power systems (Heat, Electricity, Gas)	High

Table 4-5: Forecasting Engine functional requirements

Name of component		Responsible partner
Forecasting Engine		CERTH
Functionality	Rationale	Associated Use Cases
Forecast prosumer demand/generation	Prosumer demand/generation forecasting based on prediction models using historical data of consumption as well as including real time forecasting when possible technically, to support day-ahead and intra-day optimal dispatch and ancillary service provision.	UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8
id	Functional Requirements	Priority
FR-FE-01	Support of different time horizons, from 15 minutes to 24 hours ahead.	High
FR-FE-02	Utilize both physical and data-driven prediction models.	High
FR-FE-03	Detect and fill possible missing values, remove outliers and implement normalization techniques.	Medium
FR-FE-04	Provide at least 2 different ML methodologies for prediction.	High
FR-FE-05	Support data inputs/ outputs in different formats (json, xml, and csv).	Medium

FR-FE-06	Communicate seamlessly with the Enterprise Service Bus to retrieve historical measurements	High
FR-FE-07	Provide day-ahead and short term forecasts on a daily basis	High
FR-FE-08	Retrain the models after a predefined time period	High
FR-FE-09	To be executed as a standalone service or as a standalone or as package library	High

Table 4-6: Centralized Dispatcher functional requirements

Name of component		Responsible Partner
Centralized Dispatcher		CERTH, TNO, CWD
Functionality	Rationale	Associated Use Cases
Optimal set-point calculation and dispatch.	Computation of optimal set-points for the utility scale and behind-the-meter assets and dispatch of the assets, according to the end-user's specifications and comfort levels, the forecasted production/consumption, the day-ahead and intra-day market scheduling and the need for ancillary services.	UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8
id	Functional Requirements	Priority
FR-CD-01	Support multiple time intervals, from real time to day-ahead and days-ahead.	High
FR-CD-02	Receive load/ device monitoring data at any time.	High
FR-CD-03	Communicate continuously with the GOPACS platform, the market operator and the VPP operator.	High
FR-CD-04	Communicate seamlessly with the HEMS.	High
FR-CD-05	Decision-making logic based on profit maximization and cost minimization and maximization of the KPIs as defined in D4.7	High
FR-CD-06	Calculation of DERs set-points according to the use case goals and apply these set-points to the dispatchable units.	High
Functionality	Rationale	Associated Use Cases
FlexEngine	The CD should have a component that can calculate the flexibility that can be provided by the assets from behind-the meter to utility scale and provide information to the market operator, the aggregators, the market players and the	UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8

	assets owners. This flexibility may then be used in the appropriate market service.	
id	Functional Requirements	Priority
FR-FLX-01	Support calculation of Real time to day-ahead flexibility for all devices that are part of the use cases.	High
FR-FLX-02	Calculate flexibility estimation of individual prosumers.	High
FR-FLX-03	Communicate with the CD and with the Forecasting module.	High
FR-FLX-04	Send information to the Virtual Energy Console and the market operator.	High
FR-FLX-05	Provide calculation of flexibility in an automated way.	High

Table 4-7: Aggregation and Intelligent Segmentation functional requirements

Name of component		Responsible Partner
Aggregation and Segmentation	Intelligent	CERTH
Functionality	Rationale	Associated Use Cases
Aggregation, Classification, Clustering	The aggregation and classification module of the iVPP will enable the user to aggregate the different assets of the overall system and create different pools, providing useful information for DR programs. Also, it will provide insights to the aggregator regarding the state of their portfolio and assist with forecasting.	UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8
id	Functional Requirements	Priority
FR-AIS-01	Support data inputs/ outputs in different formats (json, xml, and csv).	High
FR-AIS-02	Receive load/ device monitoring data at any time.	High
FR-AIS-03	Communicate continuously with the SO, the market operator the Forecasting Engine and the VPP operator.	High
FR-AIS-04	Communicate Seamlessly with the Enterprise Service Bus	High
FR-AIS-05	Provide meaningful visualizations regarding the energy behavior of the residential users to the respective Operators/Aggregators/Retailers	High
FR-AIS-06	Create clusters of residential customers and provide them as input to the Forecasting Engine to create forecasts based on the clusters.	High

Table 4-8: Virtual Energy Console functional requirements

Name of component		Responsible Partner	
Virtual Energy Console		CWD	
Functionality	Rationale	Associated Use Cases	
Graphical user interface and data representation	Provides the VPP operator with a visualization of all the necessary data for a clear view of the operation of the VPP.	UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8	
id	Functional Requirements	Priority	
FR-VEC-01	Utilize many tools for data visualization and representation.	High	
FR-VEC-02	Communicate seamlessly with all the other iVPP modules.	High	
FR-VEC-03	Allow remote operation.	High	

Table 4-9: DLT-Based Transactive Logic functional requirements

Name of component		Responsible Partner	
DLT-Based Transactive Logic		ENG	
Functionality	Rationale	Associated Use Cases	
Enables P2P energy trading	Provides a tool to the prosumers to offer their excess of production or consumption to a Local Energy Market.	UC1	
id	Functional Requirements	Priority	
FR-DLT-01	Utilizes blockchain technology.	High	
FR-DLT-02	Secures the prosumers personal information.	High	
FR-DLT-03	Communicates with the Centralized Dispatcher.	High	
FR-DLT-04	Provides tokenization of the transactions.	Medium	

Table 4-10: Secure Enterprise Service Bus (ESB) functional requirements

Name of component		Responsible Partner	
IANOS Secure Enterprise Service Bus (ESB)		ETRA	
Functionality	Rationale		Associated Use Cases
Handles data transactions	Provides the IANOS iVPP framework the required exchange and handling of data and control commands among the different system components and field devices.		UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8, UC9
id	Functional Requirements		Priority
FR-ESB-01	Consider data transactions for various energy services.		High
FR-ESB-02	Develop security and privacy.		High
FR-ESB-03	Address national and DSO rules regarding the lawful constraints about privacy.		High
FR-ESB-04	Allow different privacy profiles for each functionality and for each level of data dynamics.		High
FR-ESB-05	Communicate seamlessly with asset agents and other components.		High

Table 4-11: Back-end Agent Functional Requirements

Name of component		Responsible Partner	
Back-end Agent		CERTH	
Functionality	Rationale		Associated Use Cases
Consolidates interactions and communications amongst iVPP modules.	Responsible for handling the daily operation of iVpp modules on a daily basis.		UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8
id	Functional Requirements		Priority
FR-BEA-01	Seamslessly communication with the iVPP components		High

FR-BEA-02	Executing the iVPP on a daily basis using a scheduler	High
FR-BEA-03	Ensures the smooth operation of the iVPP components utilizing logging mechanisms	High

4.2 Non-Functional Requirements

The non-functional requirements describe how that the system should do a functionality, while not necessarily affecting the main functionality of the system. By defining the non-functional requirements, the usability and the effectiveness of the system is ensured. IANOS system non-functional requirements are presented in the following table.

Table 4-12: LCA/LCC Toolkit non-functional requirements

Name of component		Responsible Partner
VERIFY-D platform - LCA and LCC Toolkit		CERTH
Functionality	Rationale	Associated Use Cases
Assesses the environmental impact and financial costs for new investments.	The energy communities, the prosumers and the stakeholders of the islands in general will need aid in the decision making process for their investments. The LCA/LCC toolkit will offer insight for Energy Savings, Reduced Fossil Fuel Consumption, Reduced GHG emissions and primary energy demand and consumption related with new investments in RES and storage as well as all the projected costs.	UC 9
id	Non-Functional Requirements	Priority
NR-VRF-01	Is secure from cyber attacks.	High
NR-VRF-02	Is kept up to date with the most recent regulations regarding finance and environment.	High
NR-VRF-03	Can store data in a secure and tamperproof manner.	Medium
NR-VRF-04	Can be scaled upwards and downwards to handle more assets and different types of assets.	High



Table 4-13: CrowdEquity platform non-functional requirements

Name of component		Responsible Partner	
CrowdEquity Platform		CERTH	
Functionality	Rationale		Associated Use Cases
Provide an online platform for crowdfunding of new projects in exchange for equity.	Part of IANOS project is the involvement of the local communities in the planning and ownership of the new RES projects that are going to be created. For this purpose, the CrowdEquity toolkit will be provided. This toolkit will provide an online platform for sharing ideas and business plans from investors while providing them to a source of funding via Crowd Funding in exchange for equity.		UC 9
id	Non-Functional Requirements		Priority
NR-VRF-01	Is user friendly.		Medium
NR-VRF-02	Respects GDPR codes.		High
NR-VRF-03	Provides multiple payment methods for real currency.		Medium
NR-VRF-04	Can support multiple users at a given time.		High
NR-VRF-05	Is secure from cyber-attacks.		Medium
NR-VRF-06	Stores data in a secure and tamperproof manner.		High

Table 4-14: Cost Benefit Analysis Component non-functional requirements

Name of component		Responsible Partner
Cost Benefit Analysis Component		CERTH
Functionality	Rationale	Associated Use Cases
Provide an online tool for conducting cost-benefit analysis.	Users of the IEPT toolkit will need to have a clear overview of the possible costs and benefits of their projects in order to decide on their realization. The CBA component will conduct this analysis and give helpful insights to the investors.	UC 9
id	Non-Functional Requirements	Priority
NR-CBA-01	Is user friendly.	Medium
NR-CBA-02	Stores data in a secure and tamperproof manner.	High
NR-CBA-03	Is secure from cyber-attacks.	High
NR-CBA-04	Is updated with the newest environmental and financial data.	Medium
NR-CBA-05	Can be scaled downwards and upwards for more and different types of assets.	High

Table 4-4-15: Grid oriented optimizer (INTEMA) non-functional requirements

Name of component		Responsible Partner
Grid oriented optimizer (INTEMA)		CERTH
Functionality	Rationale	Associated Use Cases
Provide an online tool for preliminary power system dimensioning.	When planning for new investments in RES and Storage devices, their impact on the grid and the total power flow must be taken into account. For this reason, the Grid Oriented Optimizer will calculate the energy system's power flows at each node, the optimal power flow, frequency stability, contingency plans and renewables power generation and load.	UC 9
id	Non-Functional Requirements	Priority
NR-GOO-01	Provides a GUI.	High
NR-GOO-02	Is scalable both upwards and downwards, both for more assets and different types of assets.	High
NR-GOO-03	Is secured from cyber-attacks.	High

Table 4-16: Forecasting Engine non-functional requirements

Name of component		Responsible Partner
Forecasting Engine		CERTH
Functionality	Rationale	Associated Use Cases
Forecast prosumer demand/generation	Prosumer demand/generation forecasting based on prediction models using historical data of consumption as well as including real time forecasting when possible technically, to support day-ahead and intra-day optimal dispatch and ancillary service provision.	UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8
id	Non-Functional Requirements	Priority
NR-FE-01	Is secure from cyber attacks.	High
NR-FE-02	Execution time of providing the daily forecasts should be less than 30 seconds.	High
NR-FE-03	Training time should be less than 15 minutes, depending on the amount of training data.	Medium
NR-FE-04	Can be scaled upwards and downwards regarding the number and type of assets.	High
NR-FE-05	Support both Power and Energy consumption/generation prediction	Medium
NR-FE-06	Provide evaluation metrics for the forecasting performance (MAE, RMSE, SMAPE, MMR)	High
NR-FE-07	Provide explanatory figures for the evaluation of the forecasting accuracy	High

Table 4-17: Centralized Dispatcher non-functional requirements

Name of component		Responsible Partners	
Centralized Dispatcher		CERTH, TNO, CWD	
Functionality	Rationale		Associated Use Cases
Optimal set-point calculation and dispatch.	Computation of optimal set-points for the utility scale and behind-the-meter assets and dispatch of the assets, according to the end-user's specifications and comfort levels, the forecasted production/consumption, the day-ahead and intra-day market scheduling and the need for ancillary services		UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8
id	Non-Functional Requirements		Priority
NR-CD-01	Is secure from cyber-attacks.		High
NR-CD-02	Can be scaled upwards for the calculation of more users set-points/assets.		High
NR-CD-03	Execution time of extracting the optimal setpoints should be less than 5' minutes		High
NR-CD-04	When no data are available for optimal decision the no new setpoints are defined for the assets and the assets continue their operation as before		High
NR-CD-05	When no data are available due to a component malfunction or a communication error, the user should be notified via the Virtual Energy Console in order for the user to find the solution.		High
NR-CD-06	When no optimal schedule can be evaluated from the CD (for example no flexible assets to be controlled) the CD should dictate the continues system operation.		High
Functionality	Rationale		Associated Use Cases
Flex Forecaster	The CD should have a component that can calculate the flexibility that can be provided from the assets from behind-the meter to utility scale and provide information to the SO, the market operator, the aggregators, the market players and the assets owners. This flexibility may then be used in the appropriate market service.		UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8
id	Non-Functional Requirements		Priority
NR-FLX-01	Is secure from cyber-attacks.		High
NR-FLX-03	Can compute flexibility in a near to real time scale		High
NR-FLX-04	Is scalable both regarding the quantity of assets and the type of assets to be forecasted.		High

Table 4-18: Aggregation and Classification non-functional requirements

Name of component		Responsible Partner
Aggregation and Intelligent Segmentation		CERTH
Functionality	Rationale	Associated Use Cases
Aggregation, Classification and clustering	The aggregation and classification module of the iVPP will enable the user to aggregate the different assets of the overall system and create different pools, providing useful information for DR programs. Also, it will provide insights to the aggregator regarding the state of their portfolio and assist with forecasting.	UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8
id	Non-Functional Requirements	Priority
NR-AIS-01	Has runtimes of up to 5 minutes, depending on the amount of data.	High
NR-AIS-02	Is secure from cyber-attacks.	High
NR-AIS-03	Can be scaled both for more users and for different features.	Medium
NR-AIS-04	Provide evaluation metrics for the results (silhouette score)	High
NR-AIS-05	Provide explanatory plots for the evaluation of the results (scatter plots)	High

Table 4-19: Virtual Energy Console non-functional requirements

Name of component		Responsbile Partner	
Virtual Energy Console		CWD	
Functionality		Rationale	Associated Use Cases
Graphical user interface and data representation		Provides the VPP operator with a visualization of all the necessary data for a clear view of the operation of the VPP.	UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8
id	Non-Functional Requirements		Priority
NR-VEC-01	Provides a GUI with user friendly visualization of data.		High
NR-VEC-02	Is secure from cyber attacks		High
NR-VEC-03	Performs analysis and data representation in a very short time frame (seconds)		High
NR-VEC-04	Allows easy and transparent access to data from all stakeholders.		High

Table 4-20: DLT-Based Transactive Logic non-functional requirements

Name of component		Responsible Partner	
DLT-Based Transactive Logic		ENG	
Functionality		Rationale	Associated Use Cases
Enables P2P energy trading		Provides a tool to the prosumers to offer their excess of production or consumption to a Local Energy Market.	UC1
id	Non-Functional Requirements		Priority
NR-DLT-01	Respects GDPR codes.		High
NR-DLT-02	Provides secure communication for the user.		High
NR-DLT-03	Provides a user-friendly application.		Medium
NR-DLT-04	Can be scaled to accommodate more users.		High
NR-DLT-05	Stores transactions in a secure manner.		High

Table 4-21: Secure Enterprise Service Bus (ESB) non-functional requirements

Name of component		Responsible Partner	
IANOS Secure Enterprise Service Bus (ESB)		ETRA	
Functionality	Rationale	Associated Use Cases	
Handles data transactions	Provides the IANOS iVPP framework the required exchange and handling of data and control commands among the different system components and field devices.	UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8, UC9	
id	Non-Functional Requirements	Priority	
NR-ESB-01	Consider active networks in condition of high renewables penetration, increased electrical vehicles presence, storage and DMS.	High	
NR-ESB-02	Consider the will of end-customer to participate on a benevolent way to services.	High	
NR-ESB-03	Can be scaled to accommodate more assets.	High	
NR-ESB-04	Is secure from cyber attacks.	High	
NR-ESB-05	Sends error message to the Virtual Energy Console when communication with an asset breaks down	High	

Table 4-22: Back-End Agent Non-Functional Requirements

Name of component		Responsible Partner
BackEnd Agent		CERTH
Functionality	Rationale	Associated Use Cases
Consolidates interactions and communications amongst IVPP modules	Responsible for handling the daily operation of IVPP modules	UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8
id	Non-Functional Requirements	Priority
NR-BEA-01	Consists of several independent components	High
NR-BEA-02	Provides secure communication among the IVPP components	High
NR-BEA-03	Scalable to accommodate more components	High
NR-BEA-04	Is secure from cyber attacks.	High

5 Development View

The development view of the IANOS system is going to be presented in this section. As mentioned in the previous chapters, the development view will provide information about the use of the existing software, considering the several legal and accessibility issues, as well as the technical requirements and the dependencies of each component. Finally, the inputs and outputs for each component will be highlighted in this section.

5.1 Business Layer

5.1.1 *VERIFY-D platform tool (LCA and LCC module)*

The VERIFY tool receives input from the IANOS Secured Enterprise Service Bus (ESB), which securely monitors energy-related and contextual data from the dispersed field components (storage vectors, dispatchable generators, flexible prosumers).

The output will be fed into the IEPT (T3.3) to assist the system planner on examining the potential of infusing new technologies in its RE portfolio and recommending on additional energy storage solutions. In addition, the extracted output results of the LCC module will support the community-driven investment plans of T3.2 (Crowdequity component).

The aforementioned connections of the VERIFY tool are presented in the figure below:

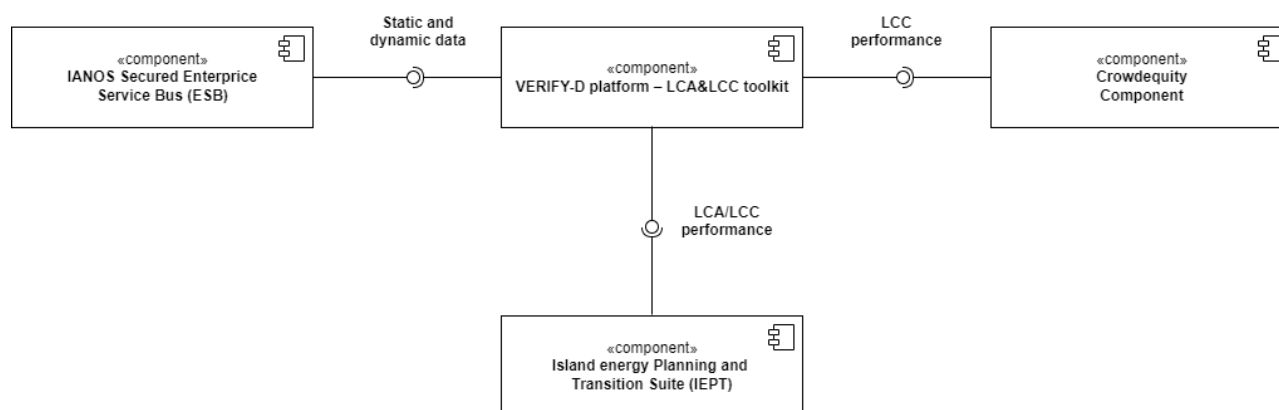


Figure 5-1: VERIFY-D platform tool (LCA and LCC module) functionality diagram.

5.1.2 *CrowdEquity Component*

CrowdEquity component will provide an equity crowdfunding platform for the project's stakeholders to invest in the development of new renewable energy assets. The component will receive input from the LCC assessment component that will support the community-driven investment plans, as well as the DLT-based Transactive Logic component in the form of tokenized energy data, which will be used in the transactions. Additionally, user input, such as user's personal information, will be fed to the component.

The platform will send output data varying from transaction details and results, available investment opportunities, and more.

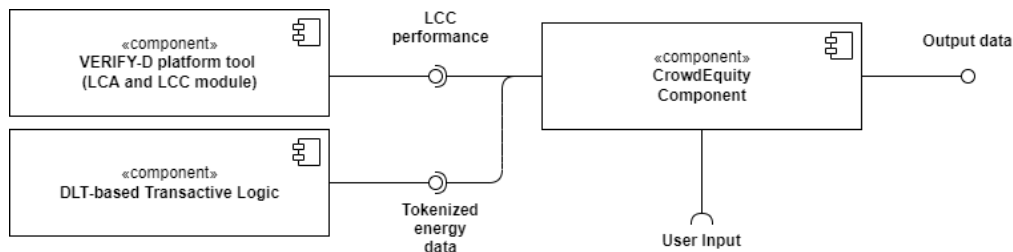


Figure 5-2: CrowdEquity component's functionality diagram.

5.1.3 System Modeller

The core component is ESSIM which provides a REST interface to start a simulation, it requires an ESDL file that defines the appropriate scenario. Typically, the ESDL MapEditor is used to create and generate scenarios and uses the REST API of ESSIM to start a simulation. These scenarios are stored as ESDL files in the ESDL Drive which provides Git like features for storing ESDL files.

Subsequently, ESSIM starts a simulation based on an ESDL file and stores its results in both a document storage and a timeseries storage. If configured, the KPI modules will calculate KPIs based on simulation results of ESSIM.

Simulation results can be viewed in both the ESDL MapEditor and Grafana dashboards that are generated after each simulation run. Aggregated simulation results are input for the IEPT Suite.

ESSIM development view is depicted below:

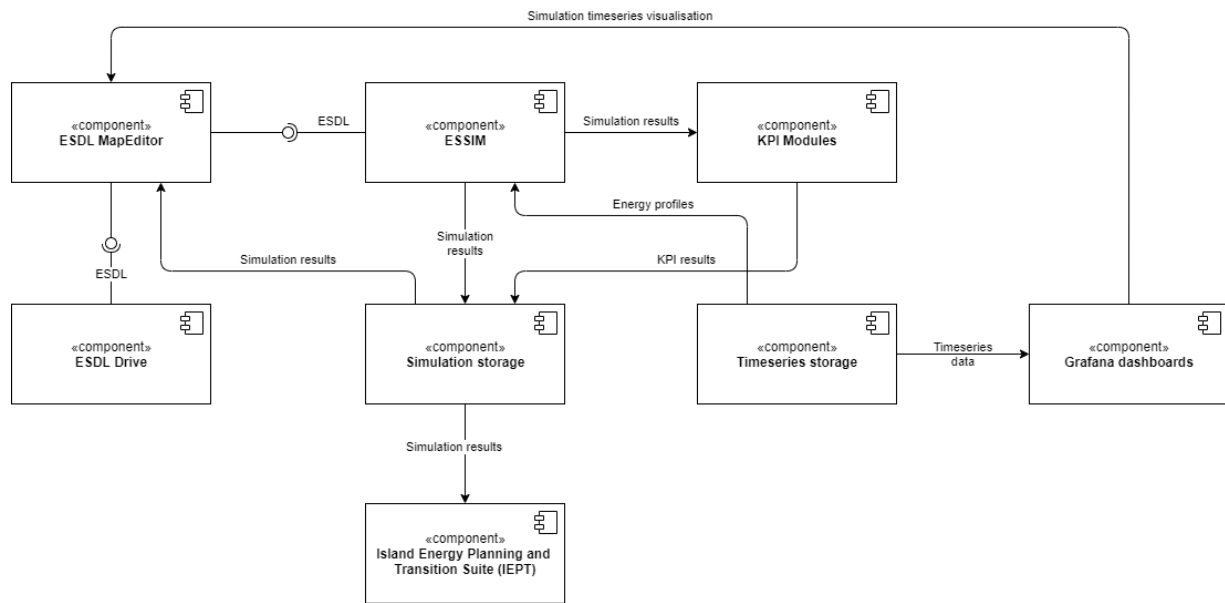


Figure 5-3: ESSIM development view.

5.1.4 Grid-Oriented Optimizer - INTEMA.grid

As an input, INTEMA.grid receives off-line data from the IANOS Secured Enterprise Service Bus (ESB), which is a secure communication pipeline, monitoring energy-related variables, while for the case of forecasting algorithms real-time data are required from the side of generation and/or consumption.

The output will be fed into the IEPT (T3.3) to estimate the potential of additional installed RES capacities, along with their impact on the grid's stability in terms of frequency and voltage response. Moreover, through this grid planning tool, multiple energy storage systems and strategies of operation can be investigated to assess their impact on grid flexibility and offering of ancillary services, while facilitating the selection and dimensioning of most promising solutions towards increasing penetration of currently installed RES, while minimizing the risk of curtailment events.

The aforementioned connections of the INTEMA.grid tool are presented in the figure below:

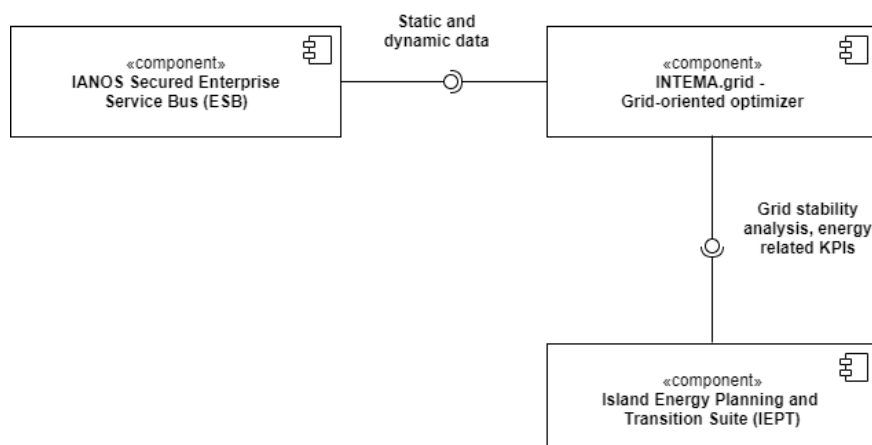


Figure 5-4: INTEMA.grid – grid-oriented optimizer's functionality diagram.

5.1.5 CBA component

The CBA component receives the outcome of the LCA/LCC performance outcome from the corresponding component. In addition, from the INTEMA grid-oriented optimizer, the CBA component gets as an input grid stability analysis in order to be introduced as an aspect in the benefits that an energy grid intervention provides. The last input is coming from the System Modeller, which is able to assess the power system adequacy considering the interventions conducted in the IANOS demonstrators. The outcome of the CBA components is an outcome regarding the CBA from a holistic perspective from different aspects of the involved stakeholders. This outcome can be also considered as the main result of the IEPT component.

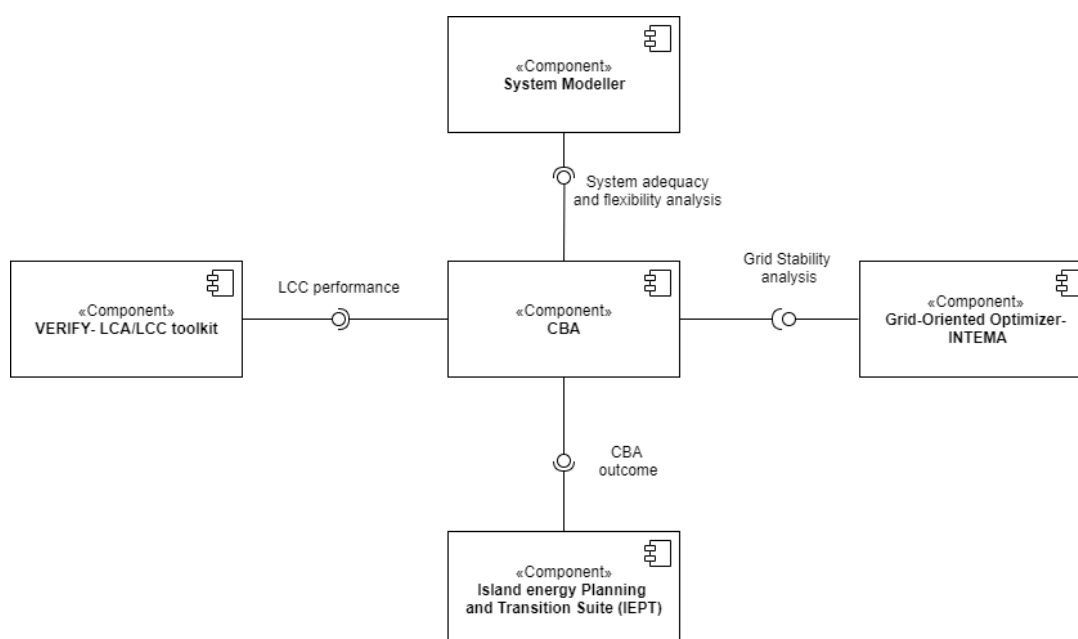


Figure 5-5: CBA component's functionality diagram.

5.2 Decision Making Layer

5.2.1 Aggregation and Intelligent Segmentation

In the context of energy clustering, the data aggregation module conducts the various procedures applied on raw load measurements to provide the proper inputs for the clustering model depending on the relative objective stated by the relevant portfolio Manager/Aggregator/Retailer.

The component will consist of two main sub-components, the Aggregation and the Segmentation. Firstly, the Aggregation sub-component will be responsible for collecting smaller, similar, and adjacent data from heterogenous sources and combining them to larger sums. The Segmentation sub-component uses data mining techniques to extract useful information from the data. The sub-component will utilize unsupervised learning algorithms to examine patterns in the data and add them into clusters, based on the operator's relative objective. The labels of the clustered data will be stored in the database and be available to the other components of the iVPP. The clustering sub-component will have a dedicated functionality regarding the assistance with forecasting. Another sub-component of Segmentation will be the classification sub-component. Classification is a supervised machine learning technique. The classification sub-component will classify new prosumers into the already existing groups created by the clustering sub-component based on a number of load measurements.

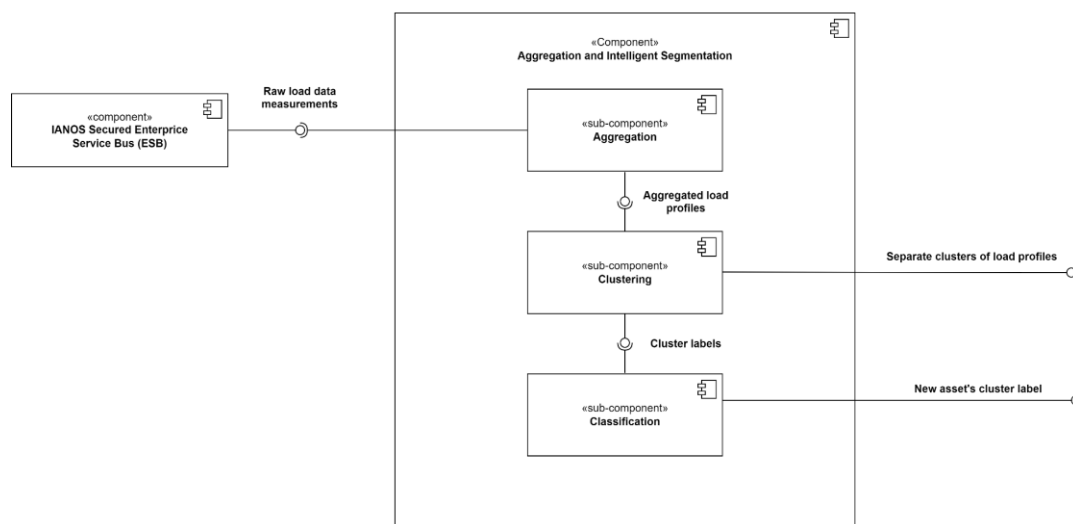


Figure 5-6: Aggregation and Classification functionality diagram.

5.2.2 Forecasting Engine

As stated in the previous section, IANOS Forecasting Engine component is responsible for the forecasting of the renewable energy asset's energy production, as well as the forecasting of the energy consumption of the energy demanding assets. It receives dynamic consumption and production data from the IANOS Secured Enterprise Service Bus (ESB), clusters produced by the Aggregation and Classification component, weather data from the external Weather service API and finally price data from the Energy Market.

The Forecasting Engine component is divided into three sub-components, depending on the forecasting data, namely Consumption Forecasting, Production Forecasting and Price Forecasting subcomponents. The three subcomponents will utilize physical models and data-driven models, using various machine learning techniques in order to provide predictions about the future energy consumption and production of the assets, as well as the electricity price in different time intervals.

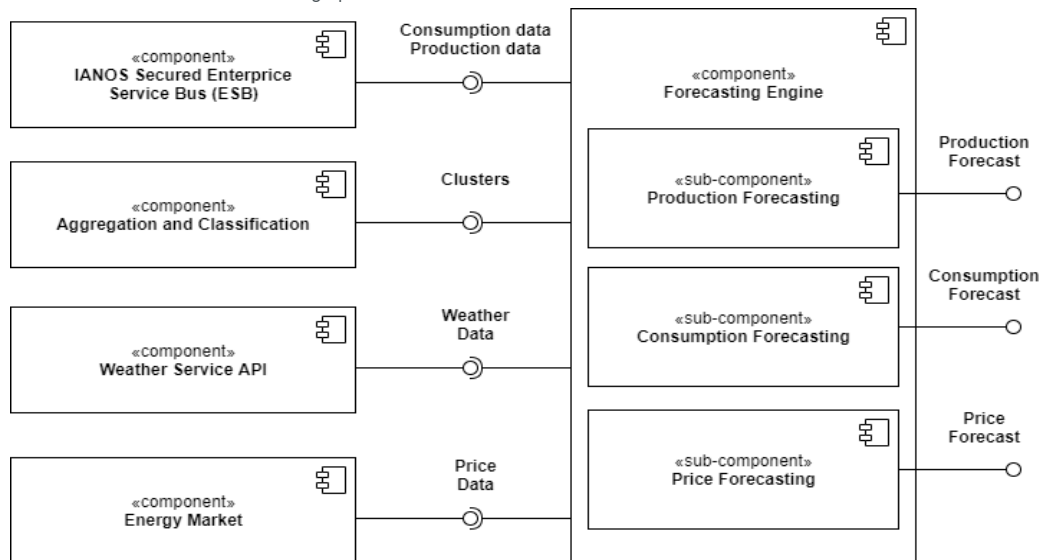


Figure 5-7: Forecasting Engine functionality diagram.

5.2.3 Centralized Dispatcher

The development view of the Centralized Dispatcher is separated into two different views, the Terceira view where the Centralized Dispatcher utilizes the KIPLO and optiMEMS tools and the Ameland view where it utilizes the REFLEX tool.

5.2.3.1 Terceira View

KIPLO Core Platform

As mentioned in the logical view section, KIPLO Core Platform will be the main platform that will perform the collection analysis and processing of certain input data,

communicate with optiMEMS, and solidify the dispatching setpoints. It will be used as interface for other iVPP components, being the core for the Centralized Dispatcher.

KIPLO will utilize a communication module in order to receive real-time measurement data from the device agents via IANOS secured Enterprise Service Bus (ESB). Additionally, the Forecasting Engine will provide KIPLO with production and consumption forecasts through an API, while ESB will also provide the platform with the generated dispatch schedule, that were generated in optiMEMS component. By utilising different communication protocols, KIPLO will also be able to communicate with other third-party components to receive input data and send output data.

Using the data processing module and data analytics modules KIPLO will provide the ESB with the scheduling of the assets' operation and the appropriate dispatch setpoints. Lastly, it will communicate with Virtual Energy Console to receive user interface input data and send user interface output data. The following figure provides the functionality diagram of KIPLO Core Platform.

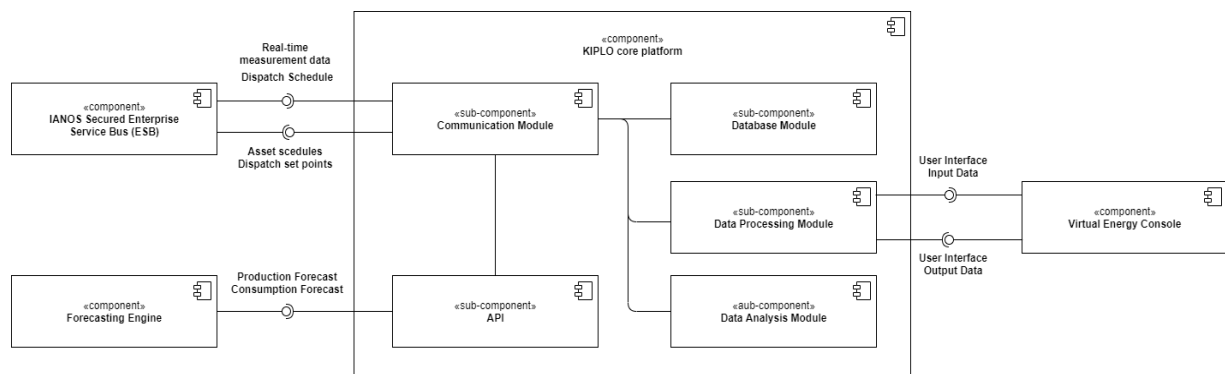


Figure 5-8: KIPLO Core Platform functionality diagram.

OptiMEMS

In the Terceira view, optiMEMS will provide the system with the optimal scheduling of the small and large scale assets of the grid. In IANOS system, the optiMEMS component will consist of the optimization (OSEM) module that will provide the optimal dispatch schedule and a communication module that will receive the necessary data from other external components.

The existing forecasting module of OptiMEMS will not be utilized but the component will receive consumption and production forecast data from the Forecasting Engine, as well real-time measurements from the device agents via IANOS Secure Enterprise Service Bus (ESB). Additionally, third party component will provide the module with the necessary electricity price data for the optimization process. Through the prosumer app, the end-user of the system will be able to set comfort restrictions as well as operational settings. Finally, the VPP operator will send the info regarding the technical constraints of the BESS, thermal storage, EV/V2G charging respectively.

The output of the component will be the optimal dispatch schedule of IANOS energy assets mainly for BESS, thermal storage, and EV/V2G charging, and it will be sent to the KIPLO Core platform via the Secure Enterprise Service Bus (ESB).

The following figure represents the functionality diagram of the OptiMEMS component.

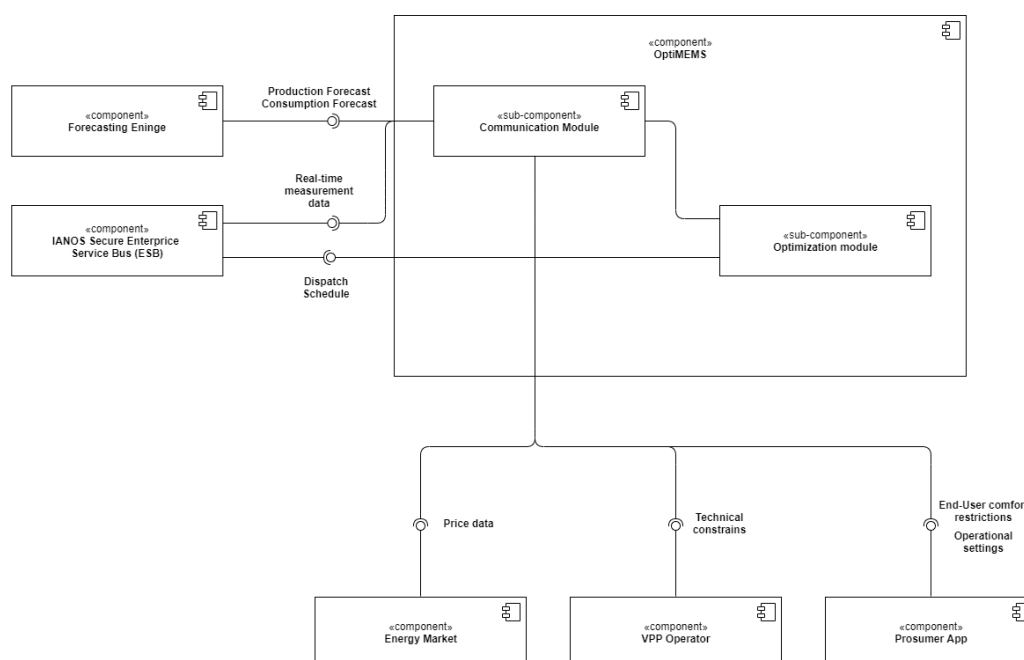


Figure 5-9: OptiMEMS functionality diagram.

5.2.3.2 Ameland View

Reflex

In the Ameland view the Centralized Dispatcher's functionality will be provided by the ReFlex software solution. Reflex component will receive input through the S2 interface. The S2 interface accommodates information such as device constraints, comfort levels, measurements and forecasts. In case forecasts cannot be provided by the device itself, the forecasting engine is used to provide a forecast for that device. Data received from the devices will also be put on the IANOS enterprise service bus (ESB) for further processing by other components. The dEF-Pi component will provide Reflex with flexibility options from the field devices, while receiving instructions.

ReFlex provides an API to operate the VPP and receive the above-mentioned data. All incoming S2 messages are provided to the Planner to create plans that optimize the dispatch of the devices. The Trader modules will ask, based on market information, the Planner for possible plans based on their constraints, and the Planner will subsequently provide flexibility options that the Trader can use. The Planner makes ensures that demand and supply match according to the specified target (e.g. self-consumption, or congestion limits) of the Trader. When a Trader selects one of the options, the Planner makes ensures that this option is dispatched to the devices, while at the same time a bid is placed in the associated market by the Trader.

When new events (e.g. updated measurements or forecasts) arise that require the current plan to be adapted, this whole process is executed again.

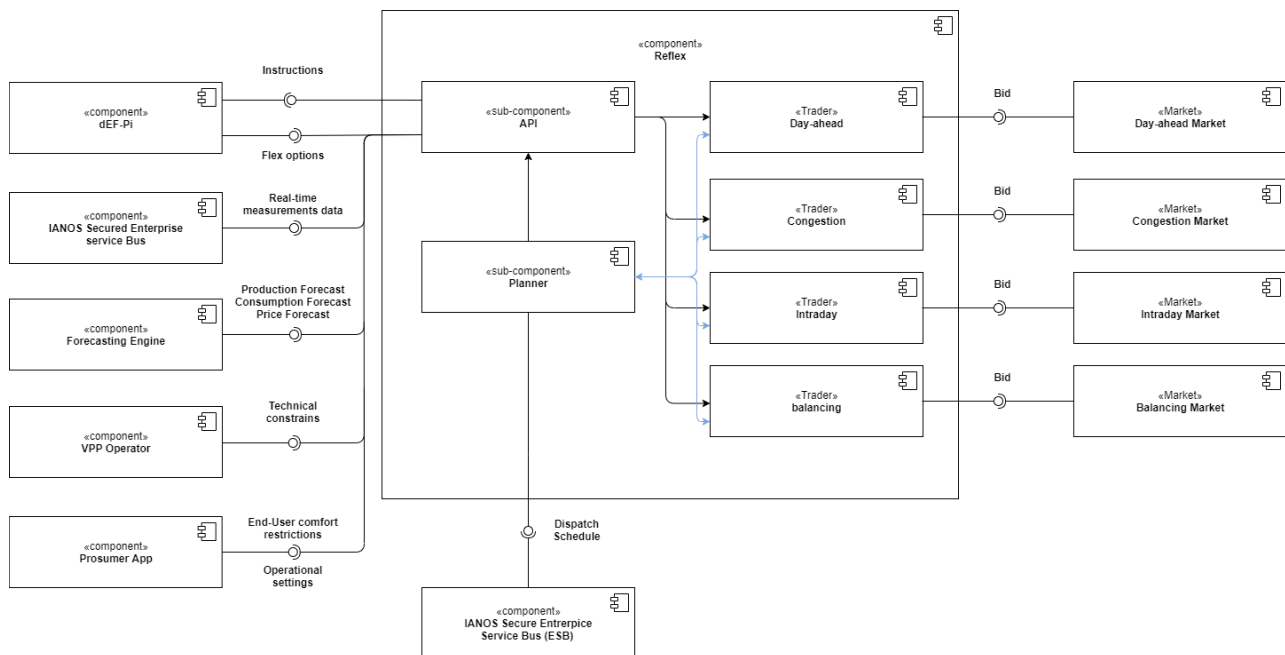


Figure 5-10: ReFlex functionality diagram.

5.2.4 DLT-based Transactive Logic

The DLT-based transactive platform is composed of three components:

- P2P energy trading platform implements the business logic of the P2P market, the validation of energy transactions between prosumers, prosumer management, aggregation of production and consumption measures coming from the smart meters.
- DLT services component represents the intermediate level of interaction between the logic of the application and the blockchain infrastructure. The DLT services component mainly retrieves information about users' wallets from the blockchain and invokes smart contract methods to manage the P2P market transactions.
- Blockchain infrastructure is the Ethereum private network able to provide networking, transactions validation, and on-chain storage capabilities.

Figure 5.11 shows the DLT-based transactive platform functionality diagram.

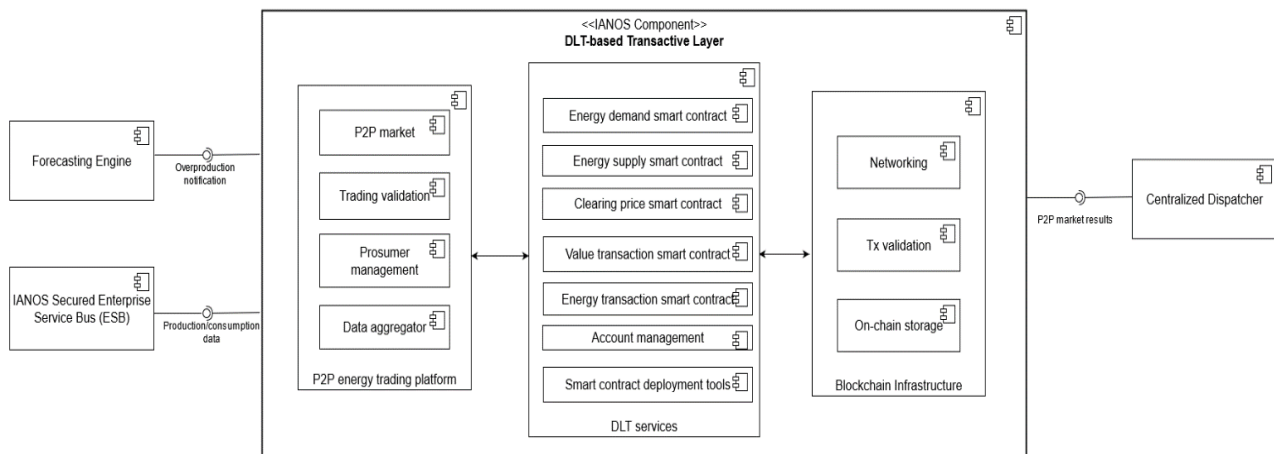


Figure 5-11: DLT-based Transactive component functionality diagram.

The DLT-based transactive platform needs input, mainly from two IANOS components:

- the Forecasting Engine notifies to the P2P market of a grid unbalance identification. This is the trigger event that opens the market session.
- the IANOS Secured Enterprise Service Bus (ESB) will provide consumption and production data from the different assets. This input is needed for the validation process after the close of the market session.

The DLT-based transactive platform should inform about the P2P market results to the Centralized Dispatcher, so the optimization engine can provide the optimal scheduled program for the next day.

5.3 User Interface Layer

5.3.1 Virtual Energy Console

The Virtual Energy Console is fundamentally a user interface application that will implement a set of modules (interaction and functionality sections) to meet the established requirements. The following modules will be developed:

- The “Energy Community Module” will comprise the creation, operation, and management procedures of the Energy Community. It will include, among others, the visualization of the historic, real time and forecast data for the generation and consumption, the setup of a new energy community and integration of new participants.
- The “Flexibility Module” comprises the aggregation, trading, and settlement of flexibility including various types of flexible assets. This module will show, for instance, the actual and forecasted aggregated flexibility, and the flexibility transactions.
- The “User Management Module” will deal with the user authentication mechanisms, user definitions and data access restrictions. This will be mainly a module that will deal with the management features of the user interface.

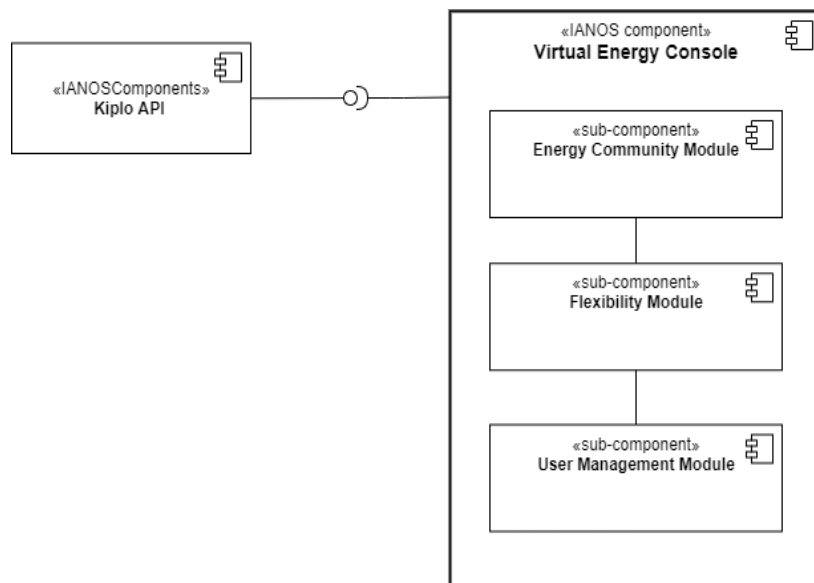


Figure 5-12: Virtual Energy Console Functionality Diagram.

5.4 Communication Layer

5.4.1 IANOS Secured Enterprise Service Bus

iVPP ESB will be based on the CITRIC smart city platform that follows the RIVER © architecture created by ETRA. RIVER is a Reactive, Interoperable, Visible, Elastic and Resilient architecture oriented to microservices and events.

It is an open architecture with capacity to grow its service network, reactive because it is event-driven, interoperable because it is supported by standard protocols and agnostic models of data, visible because it is monitored in its operation, elastic because it can be scaled out and independently in each of its services, and resilient because it is orchestrated and monitored to be fault tolerant.

CITRIC's microservices distribution fully complies with UNE178104:2017 in its orientation towards functional layer.

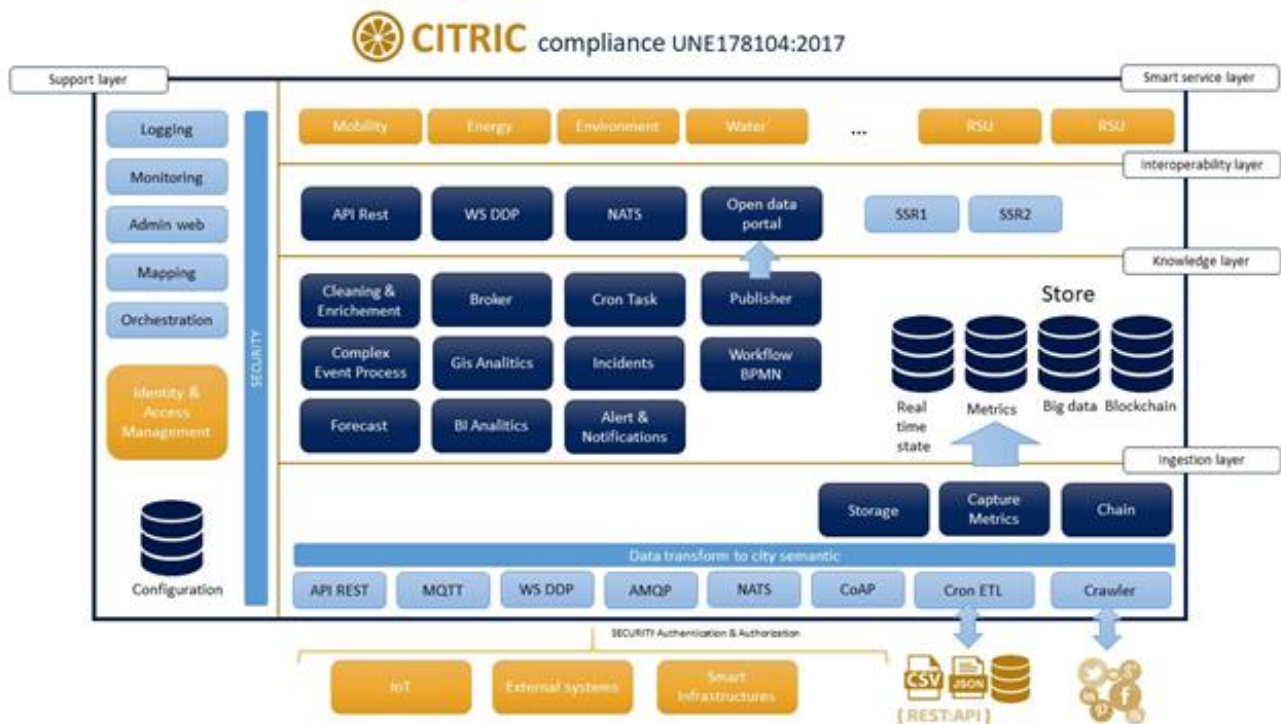


Figure 5-13: CITRIC's architecture.




All microservices are scaled out using replicas and load balancers based on the particular needs of each installation.





CITRIC can be deployed over Docker Swarm nodes or over Kubernetes cluster based depending on the dimension of the system. Besides the in-premise installation, it can easily be deployed in the Google, AWS, or Azure clouds.

CITRIC is composed by a plethora of services and modules that interact among each other and provide services. All of them are containerized, making it possible to deploy and update any architecture component in just a few minutes.

The CITRIC platform supports the following technologies for the bulk Data ingestion.

Table 5-1: CITRIC platform suggests technologies for bulk data ingestion

API REST	CITRIC has a robust secure and protected HTTPs Rest API with rate limit control to prevent attacks and allows a lot of flexibility for data ingestion through it. Besides authentication, API security allows to authorize only a certain set of data to each authenticated user. The implementation of this API has been done with Express.	
MQTT	The MQTT protocol is widely used by lightweight devices (IoT) and supported by CITRIC. To do this, CITRIC uses RabbitMQ as the MQTT ingestion service.	
AMQP	It is possible to ingest data directly with AMQP protocol to ingest directly over queues that are consumed by	

	microservices, which transform and store information in the platform storage system.	
WS DDP	It is possible to connect to the platform using websockets (e.g. from web browsers). The protocol over WS implemented is Distributed Data Protocol (DDP). A number of methods are implemented over this protocol that provide a very efficient way to do mass ingestion.	
NATS	NATS is the underlying broker on the platform and is responsible for managing the entire microservices communication network. It acts as an enterprise service bus (ESB); communication through it makes use of topics that are mapped to services. CITRIC offers a set of topics that allow you to interact directly with storage microservices with their corresponding level of security.	
CoAP	CoAP is a lightweight protocol similar to MQTT but communicating over UDP. It is normally used by moving devices that jeopardize the quality of communication. CITRIC offers this ingestion service to integrate data coming from moving vehicles that interact with the platform. This microservice is natively developed by ETRA.	
Cron ETL	This service is responsible for ingesting data offered in external files, databases or web services. The process can be triggered in a time-basis or by modifying or creating new ready-to-load files on the platform. As an ETL tool, CITRIC uses Node-Red, an interactive tool to design workflows to ingest data with possible transformation.	

Different processes can be applied to the ingested data (this is configurable):

- **Transformation:** Every ingestion process previously goes through a transformation process to normalize the data before entering it into the platform. Transformation schemas are pre-configured for each source in the configuration database and are particular to each platform deployment as they must be tailored to particular data sources and how they are stored in the storage service and then served to the higher layers.
- **Security:** Every ingestion process is authenticated and authorized by a layer of security in each microservice. Each credential per token or user/password has an authorization scheme to access a subset of platform data at three levels of security: read, write, and only public attributes.
- **Storage:** If data persistence is needed, this microservice handles the storage of data when it is injected or modified. It manages the real-time database supported by MongoDB and the big data database, supported in our case by an ElasticSearch cluster.
- **Message brokering:** the information received is routed to pre-configured endpoints, allowing for a scalable architecture.

The CITRIC architecture will be adapted and extended to support the functionalities required in IANOS. CITRIC platform plus the required additions will compose the iVPP ESB.

In the initial definition of use case, a number of data exchanges have been identified. They are summarized in the following figure:

Origin\End	IANOS iVPP	Forecast Provider	Localized EMS (FEID Plus & HBMS)	Demand Side Assets/Prosumer	Large-scale BESS	Dispatchable Assets	Non-Dispatchable Assets	Grid	Storage Assets	Locally implemented actuators	Hybrid Transformer	Smart Energy Router	V2G Charging Station	Electric Charging station	Natural Gas Platform	AHPD	Electrolyser	Load Points	Gas Grid	Local non-Dispatchable Assets	District Heating Network
IANOS iVPP				6	13,14	13			14		18	20	23	26		34					
Forecast Provider	5,42,43																				
Localized EMS (FEID Plus & HBMS)	1,2,3,4																				
Demand Side Assets/Prosumer			1,2,3,4																		
Large-scale BESS	3,7																				
Dispatchable Assets	8,9,10,27,28,37,38,39																				
Non-Dispatchable Assets	10,11																				
Grid	12									12											
Storage Assets	15,16																				
Locally implemented actuators																					
Hybrid Transformer	17																				
Smart Energy Router	29																				
V2G Charging Station	21,22																				
Electric Charging Station	24,25																				
Natural Gas Platform	29																				
AHPD	30,31																				
Electrolyser	35,36																				
Load Points	32,33																				
Gas Grid																					
Local non-Dispatchable Assets																					
District Heating Network	41																				

Figure 5-14: Information exchanges matrix

This matrix specifies the pieces of information exchanged among system components. These pieces of information are the following:

Table 5-2: Exchanged Information types

ID	Information	ID	Information
1	Energy Consumption Data	23	Optimal Setpoints For V2X Charging Stations
2	Energy Generation Data	24	Electric Charging Station Real-Time Data
3	Battery Real-Time Data	25	Electric Charging Station Hard Technical Constraints
4	End-User Comfort Restrictions And Operation Settings	26	Optimal Setpoints For Electric Charging Stations
5	Local Meteorological Forecasts	27	Fuel Cells And CHP Hard Technical Constraints

ID	Information	ID	Information
6	Optimal Setpoints For Demand Side Assets	28	Fuel Cells And CHP Real-Time Data
7	BESS And Electrolyzer Hard Technical Constraints	29	Natural Gas Platform Real-Time Data
8	Dispatchable Assets Real-Time Data	30	Digester Hard Technical Constrains
9	Dispatchable Assets Hard Technical Constraints	31	Digester Real-Time Data
10	Local Generation Energy Prices	32	Loads Hard Technical Constrains
11	Non-Dispatchable Assets Data	33	Loads Real-Time Data
12	Grid Data	34	Optimal Set-Point For AHPD
13	Optimal Set-Points For Dispatchable Assets	35	Electrolyser Hard Technical Data
14	Optimal Set-Points For BESS	36	Electrolyser Real-Time Data
15	Storage Assets Hard Technical Constraints	37	Fuel Cells And CHP Hard Technical Constraints
16	Storage Assets Real-Time Data	38	Fuel Cells And CHP Real-Time Data
17	Hybrid Transformer Data	39	Heat And Hybrid Pumps Real-Time Data
18	Optimal Set-Point For Hybrid Transformer	40	Heat And Hybrid Pumps Hard Technical Constraints
19	Smart Energy Router Data	41	District Heating Network Data
20	Optimal Set-Point For Smart Energy Router	42	Forecasted Energy Consumption Data
21	V2X Charging Station Real-Time Data	43	Forecasted Energy Generation Data
22	V2X Charging Station Hard Technical Constraints		

The following table presents the different technical constrains for each asset that is used in IANOS project.

Table 5-3: Technical constrains per asset

Asset	Technical Constraints
EVs	<ul style="list-style-type: none"> • Total power (KW) • Desired SOC for the next day/hours • Maximum charging/discharging rate of the fleet • Maximum charging/discharging power (Power Reserve capabilities- Upward/downward band constraints) • Minimization of the charging cost
BESS	<ul style="list-style-type: none"> • Rated power of the battery • Charging/discharging rate • Maximum charging/discharging power (congestion management and reserve ancillary services) • Minimum SOC for the battery to have flexibility (Totally discharging a battery is not ideal)
TCL	<ul style="list-style-type: none"> • Upward/downward band constraints • Desired range of the room's/house's temperature • House occupancy • Electrical efficiency



Asset	Technical Constraints
PV	<ul style="list-style-type: none"> • Power generation (KW) limit • Limit for the curtailment of PV generation • Range of downward/upward bands • Efficiency
Thermal Units	<ul style="list-style-type: none"> • Startup/shut down costs • Minimum generation/down time • Minimum and maximum generation (or thermal capacity if it is a water heater) capacity • Initial state of the thermal unit at the beginning of the day D
CHP	<ul style="list-style-type: none"> • Minimum and maximum power output • Ramp up/ down power generation • Amount of natural gas consumed (electrical efficiency)
Auxiliary boiler	<ul style="list-style-type: none"> • Amount of natural gas consumed – efficiency • Thermal capacity • Initial state
Reversible heat pumps	<ul style="list-style-type: none"> • Efficiency (COP)
Thermal energy storage	<ul style="list-style-type: none"> • Energy stored at the beginning (Initial state) • Loss fraction (min requirements for the thermal energy dissipated during a time interval) • Charging and discharging heat rates

As it can be seen, most of the communications in the project will happen between iVPP platform and other components, in both directions. Information linked to measurements, assets technical characteristic and forecast are flowing to the iVPP from field devices and external data providers whilst optimal set-points are flowing in the opposite direction (from iVPP to other components).

FEID-PLUS device, which acts as a local EMS at customer premises, also connects to DSM assets. iVPP will also integrate the data handled by this device.

The details of the iVPP data model for field energy devices messages are presented in the Annex I: Secured ESB communication data model based on the analysis of the data handled by the energy field assets in the region.

The functionality diagram below depicts a view of how the ESB interacts with the rest of the components:

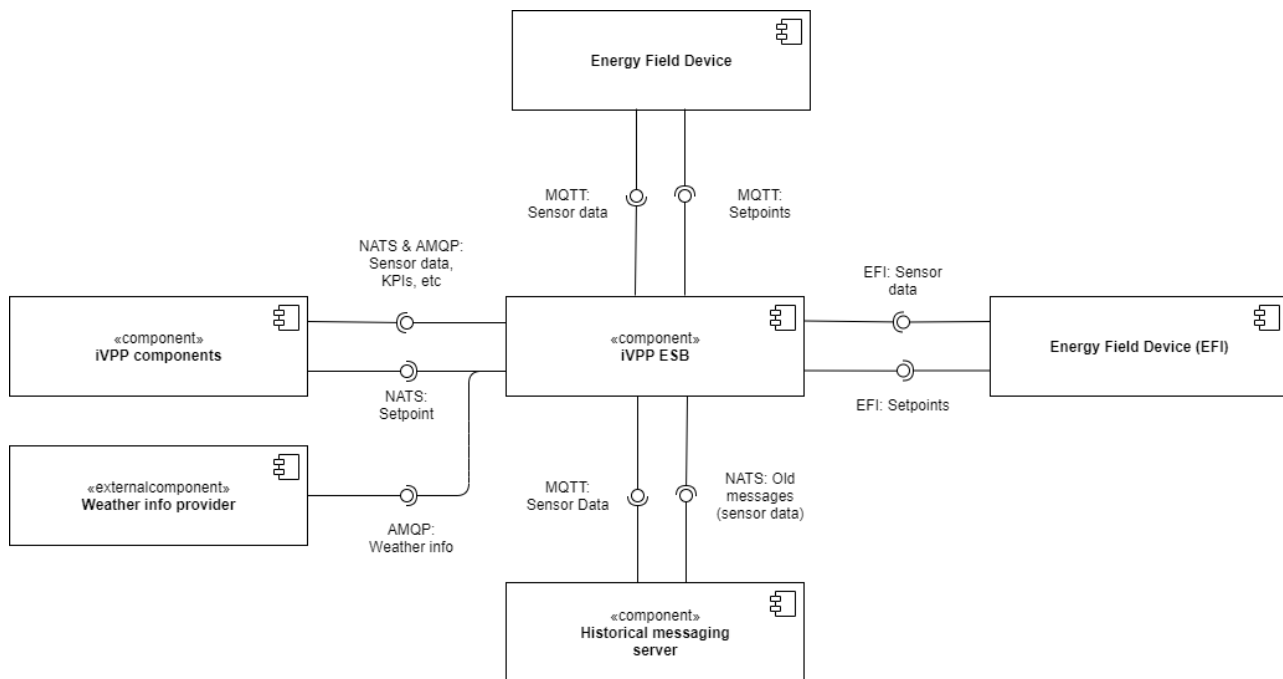


Figure 5-15: IANOS Secure Enterprise Service Bus (ESB) Functionality Diagram.

Here, all the interfaces between the ESB and the external components are shown. The raw sensor data will flow into the system in MQTT or EFI protocols in an asynchronous way. ESB will be configured to route and transform the messages to the appropriate destination (iVPP components that make use of data).

The historical messaging server component will be listening to sensor data, storing the messages in a buffer and will provide a NATS interface to allow other components to request for the last message received or for all messages in a past time window.

The weather info provider component will periodically upload real time and forecasted weather information for a set of pre-defined locations. The data model of such messages is defined in the Annex I: Secured ESB communication data model.

Finally, the different iVPP may generate set points that will be sent to the ESB in NATS and forwarded to the energy field device in MQTT or EFI protocol.

5.4.2 dEF-Pi

As mentioned in previous section, the dEF-Pi component will be used as an open platform on which IANOS different assets can be enclosed and transfer asset-specific data. More specifically, the component will interact with the Reflex platform using the standardized S2 interface (EN 50491-12-2), receiving instructions. The instructions are going to be provided to the Energy Field Devices, using S2 interface. Using the same interface, Energy Field Devices are going to provide flexibility options to the dEF-Pi component, which will send the flexibility options to Reflex platform using S2 interface. Additionally, dEF-Pi component will interact with the Monitoring platform, sending monitoring data, using MQTT standard.

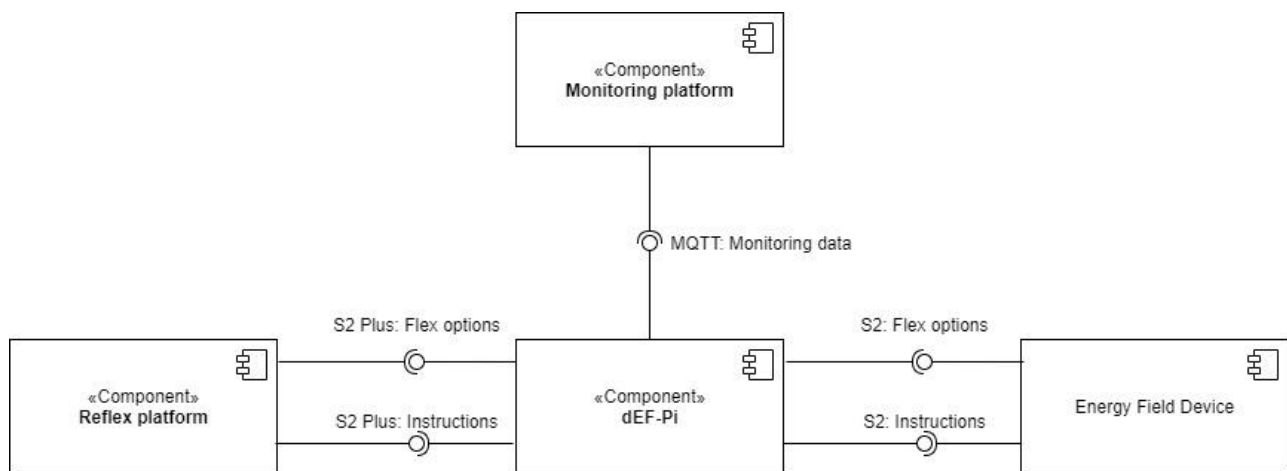


Figure 5-16: dEF-Pi component functionality diagram.

5.4.3 Back-end Agent

As mentioned in the previous chapter, Back-end Agent serves as a link between the iVPP components and the platform's other components. More specifically, the component interacts with the Forecasting Engine, sending historical values and weather conditions and receiving forecasts, the Aggregation and Intelligent Segmentation, sending historical values and receiving clusters, and finally, the OptiMEMS and Kiplo components to send consumption, production and price time series data and receive set-points. In addition, the Back-end Agent component communicates with the ESB to send data using a RESTful API and to receive data using MQTT protocol.

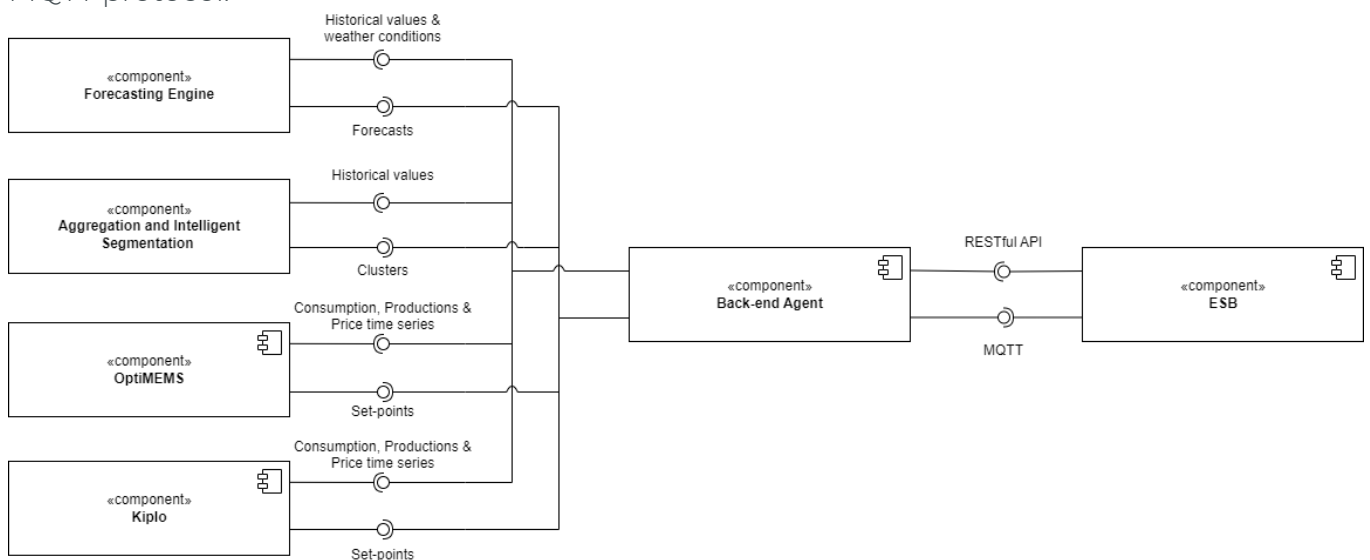


Figure 5-17: Ivpp Back-end Functionality Diagram

6 Dynamic View

The dynamic view of the system focuses on describing the workflow and data exchange between the different components and actors of the system. Interrelation and interactions between components are included, showing the information flow between the components and different actors in the previously defined Use Cases. Additionally, the UML sequence diagrams are provided, in order to provide an overview of the sequential workflow of the system in each Use Case.

The nine Use Cases of IANOS project were defined in D2.3. The current deliverable, a more detailed description of the different components is presented, with respect to the D2.3 definitions and descriptions.

6.1 Terceira pilot island dynamic view

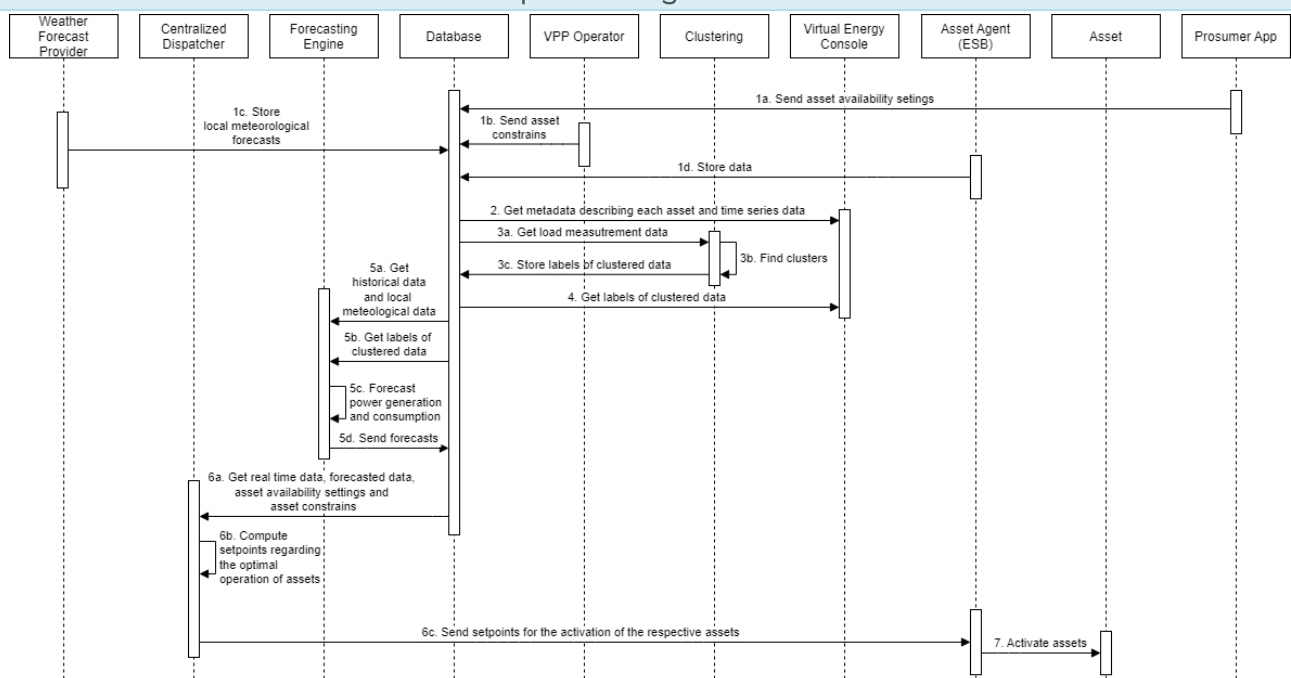
6.1.1 Use Case 1: Community demand-side driven self-consumption maximization

Use Case No.	1
Use Case Name	Community demand-side driven self-consumption maximization
<p>This Use Case occurs in a Local Energy Community (LEC) and focuses on controlling and optimizing energy assets with the main purpose of matching the energy generation from PV panels and small wind turbines and storage with the consumption of end-user or community level assets including i) electrochemical and heat batteries, ii) electric water heaters, iii) heat pumps and iv) micro-CHP through an intelligent virtual power plant (iVPP). The iVPP computes the optimization of behind-the-meter assets based on several information-sources provided by localized energy management systems (Home Energy Management Systems and Fog-Enabled Intelligent devices). Thereby, the iVPP is capable of controlling storage and demand-side assets by, for instance, shifting demand for periods of renewable generation surplus. Additionally, this use case comprises of the details regarding peer-to-peer energy trading schemes.</p>	
System Components Used – Scenario 1	
<ol style="list-style-type: none"> 1. Weather Forecast Provider (External Systems/Cloud Services) 2. Centralized Dispatcher (iVPP) (Kiplo, optiMEMS) 3. Forecasting Engine (iVPP) 4. Database (ESB) 5. VPP operator (iVPP) 6. Aggregation and Intelligent Segmentation (iVPP) 7. Asset Agent (ESB) 8. Asset (Household Smart Meters, Smart Plugs, Household Batteries, Heat Batteries, Water Heaters, Household PV Systems, Smart Energy Router) 	
Workflow of the overall System – Scenario 1	
<ol style="list-style-type: none"> 1a. The prosumer sends his/her comfort settings to the database. 1b. The Operator of the iVPP sends to the database relevant constraints regarding the operation of the various assets of the system. 	



- 1c. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database.
- 1d. The asset agents store their data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to the database.
2. The Virtual Energy Console gets metadata describing each asset and time series data in terms of consumption and generation of each asset from the database where the aggregator can get a clear overview of the assets and their condition.
- 3a. The Aggregation and Intelligent Segmentation component gets load measurement data from the database.
- 3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets into clusters based on a predefined objective.
- 3c. The labels of the clustered data are stored in the database.
4. The Virtual Energy Console gets the labels of the clustered data from the database.
- 5a. The Forecasting Engine gets historical data of previous measurements and local meteorological data from the database.
- 5b. The Forecasting Engine gets the labels of the clustered data from the database.
- 5c. The Forecasting Engine computes day-ahead and short-term forecasts for power generation and consumption.
- 5d. The Forecasting Engine sends the forecasted power generation, and power consumption to the database.
- 6a. The Centralized Dispatcher gets the real time data, forecasted data, assets constraints and comfort settings that have been stored in the database.
- 6b. The Centralized Dispatcher computes internally the setpoints regarding the optimal operation of the involved assets.
- 6c. The Centralized Dispatcher sends setpoints for the activation of the respective assets that have been selected by the optimization problem.
7. The asset agents send the aforementioned signals to the Physical Assets to start consuming energy.

UML Sequence Diagram – Scenario 1



System Components Used – Scenario 2

1. Weather Forecast Provider(External Systems/Cloud Services)



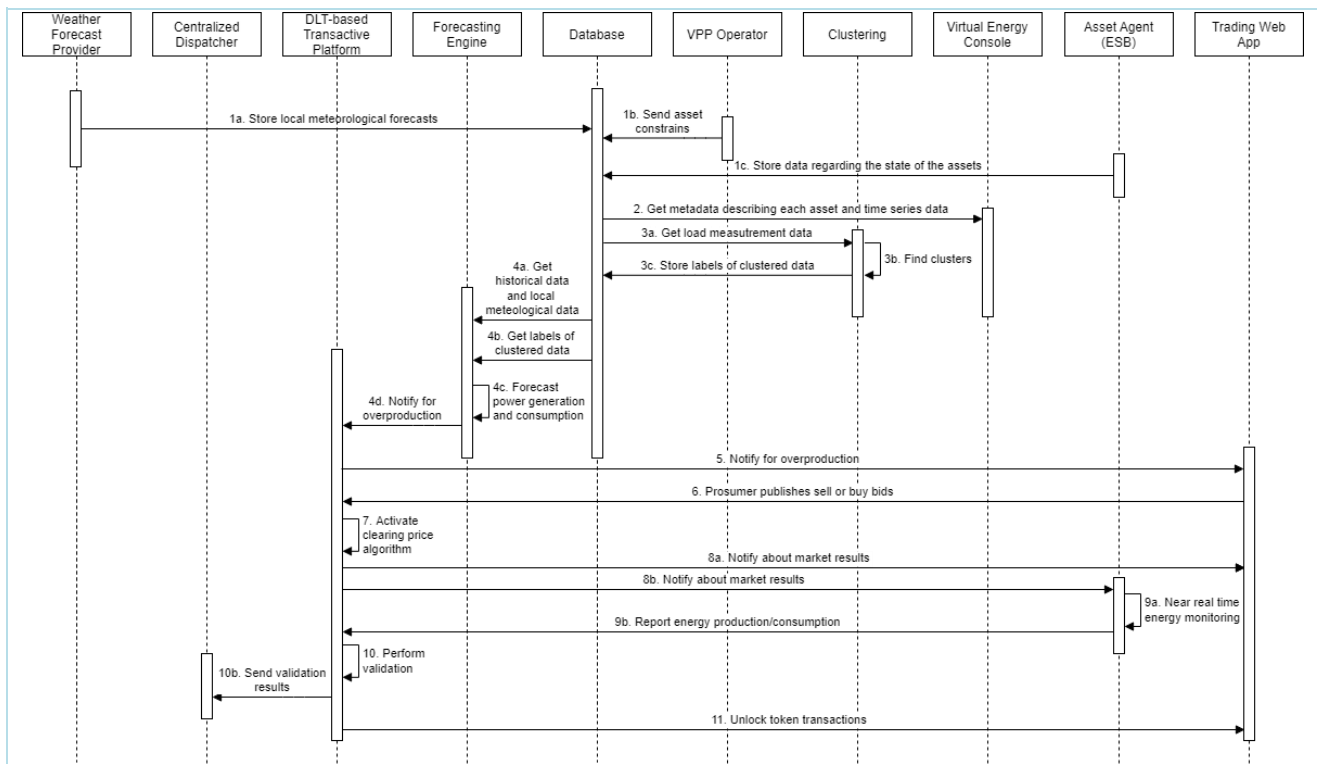
2. DLT-based Transactive Platform(iVPP)
3. Forecasting Engine (iVPP)
4. Database (ESB)
5. Aggregation and Intelligent Segmentation (iVPP)
6. Virtual Energy Console (iVPP)
7. Asset Agent (ESB)
8. Asset (Household Smart Meters, Smart Plugs, Household Batteries, Heat Batteries, Water Heaters, Household PV Systems, Smart Energy Router)
9. Trading Web App
10. Prosumer App

Workflow of the overall System – Scenario 2

- 1a. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database.
- 1b. The Operator of the iVPP sends to the database constraints regarding the operation of the various assets of the system.
- 1c. The asset agents store their data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to the database.
2. The Virtual Energy console gets metadata describing each asset and time series data in terms of consumption and generation of each asset from the database where the aggregator can get a clear overview of the assets and their condition.
- 3a. The Aggregation and Intelligent Segmentation Component gets sends historical data and asset constraints data from the database.
- 3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets of the aggregator into clusters to assist with forecasting.
- 3c. The labels of the clustered data are stored in the database.
- 4a. The Forecasting Engine gets historical data of previous measurements and local meteorological data from the database.
- 4b. The Forecasting Engine gets the labels of the clustered data from the database.
- 4c. The Forecasting Engine computes forecasts for power generation and consumption
- 4d. The Forecasting Engine notifies the DLT-based Transactive Platform for energy overproduction.
5. DLT-based Transactive Platform notifies the prosumer for overproduction via the Trading Web App.
6. The prosumer publishes sell or buy energy bids to the DLT-based Transactive Platform.
7. The DLT-based Transactive Platform activates the clearing energy price algorithm.
- 8a. The prosumer is notified about the market results from the DLT-based Transactive Platform via the Trading Web App.
- 8b. The DLT-based Transactive Platform also notifies the Asset Agents about the market results.
- 9a. The Asset Agents perform near real time energy monitoring to check whether the energy transactions are being held as agreed.
- 9b. The Asset Agents report the energy production and consumption monitoring back to the DLT-based Transactive Platform.
- 10a. The DLT-based Transactive Platform validates if the energy transactions were held according to the smart contract.
- 10b. The validation results are sent the Centralized Dispatcher, in order to facilitate any energy transaction deviations that emerged.
11. The DLT-based Transactive Platform unlocks token transactions via the Trading Web App.

UML Sequence Diagram – Scenario 2





6.1.2 Use Case 2: Community supply-side optimal dispatch and intra-day services provision

Use Case No.	2
Use Case Name	Community supply-side optimal dispatch and intra-day services provision
Use Case General Description	
<p>The present Use Case focuses on using the flexibility on the generation side, for utility-scale assets, to minimize energy curtailment in periods of renewable energy surplus. For this purpose, the intelligent Virtual Power Plant (iVPP) calculates the optimal day ahead dispatch and provides suggestions to the grid-assets with the intention of minimizing curtailment and delivering intra-day services to the grid. Accordingly, the iVPP considers three different types of utility-scale assets for this optimization: i) dispatchable assets, ii) non-dispatchable assets and iii) large-scale storage systems including both BESS and systems producing alternative fuels (electrolyzers), which support the decarbonization of islands with multi-purpose end uses. The iVPP computes the optimal dispatch set-point through provided information and delivers it to the dispatchable assets and large-scale storage systems in order to suggest behaviours to ensure the stability of the power system.</p>	
System Components Used	
<ol style="list-style-type: none"> Weather Forecast Provider (External Systems/Cloud Services) DSO (EDA) (External Systems/Cloud Services) Centralized Dispatcher (iVPP) (Kiplo, optiMEMS) Forecasting Engine (iVPP) Database (ESB) Aggregation and Intelligent Segmentation (iVPP) Virtual Energy Console (iVPP) Asset Agent (ESB) 	

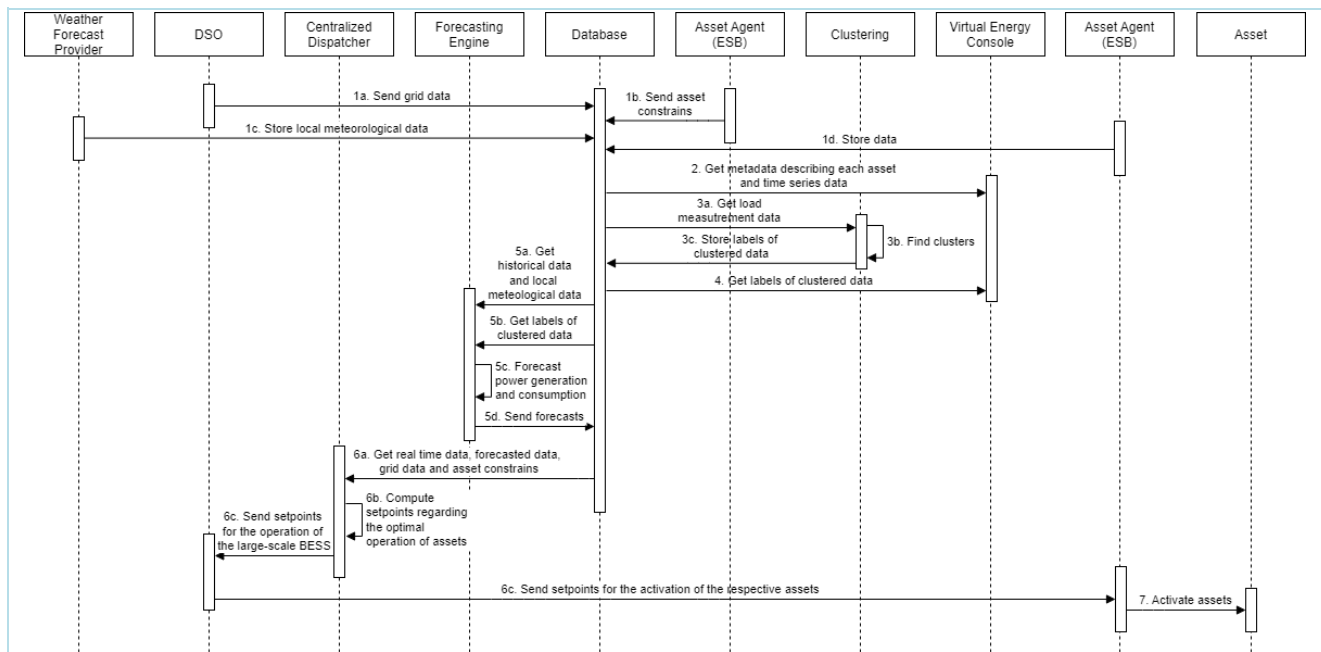
9. Asset (BESS, Geothermal plant, Wind Farms, Waste incineration power plant, Small Scale distributed PVs)

Workflow of the overall System

- 1a. The DSO sends grid data to the database.
- 1b. The Operator of the iVPP sends to the database constraints regarding the operation of the various assets of the system.
- 1c. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database.
- 1d. The asset agents store their data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to the database.
2. The Virtual Energy Console gets metadata describing each asset and time series data in terms of consumption and generation of each asset from the database where the aggregator can get a clear overview of the assets and their condition.
- 3a. The Aggregation and Intelligent Segmentation Component sends load measurement data to the database component.
- 3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets of the aggregator into clusters based on a predefined objective.
- 3c. The labels of the clustered data are sent to the database.
4. The Virtual Energy Console gets the labels of the clustered data from the database.
- 5a. The Forecasting Engine gets historical data of previous measurements and local meteorological data from the database.
- 5b. The Forecasting Engine gets the labels of the clustered data from the database.
- 5c. The Forecasting Engine computes forecasts for PV and Wind production as well as for power consumption.
- 5d. The Forecasting Engine sends the forecasted data to the database.
- 6a. The Centralized Dispatcher gets the real time energy production and consumption data, forecasted data, grid data and assets constraints that have been stored in the database.
- 6b. The Centralized Dispatcher computes internally the optimal dispatch of the assets.
- 6c. The Centralized Dispatcher sends setpoints for the operation of the large-scale BESS to the DSO.
7. The DSO sends the set points to the BESS asset agent for the activation of the large BESS.
8. The asset agent -upon validation- sends the aforementioned dispatch signals to the Physical Asset.

UML Sequence Diagram



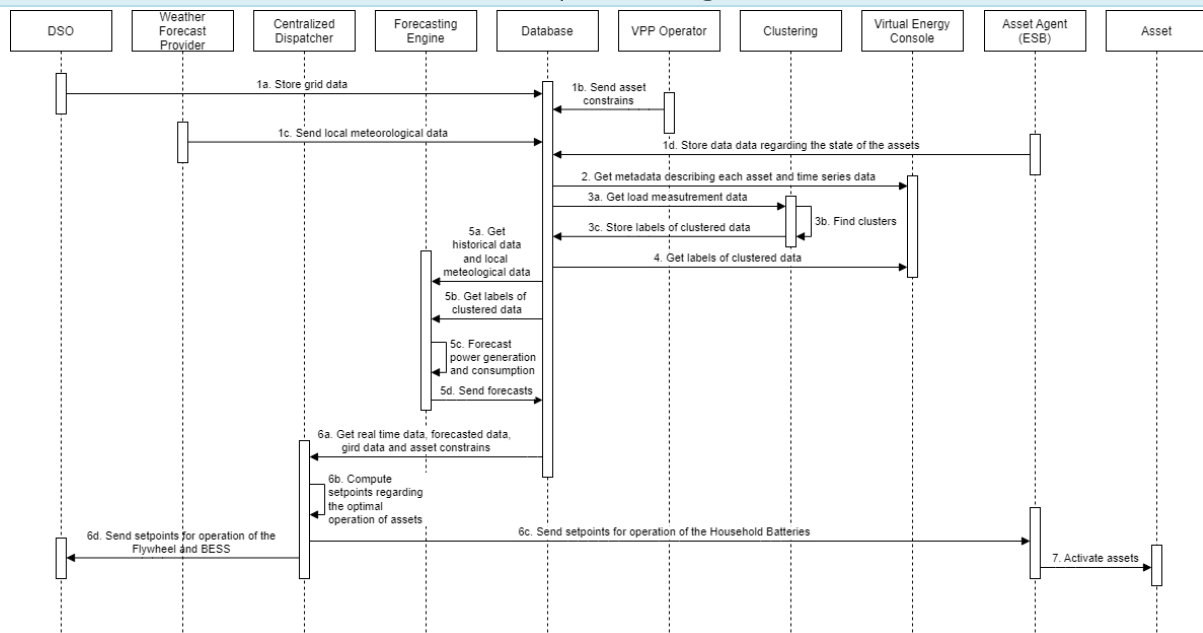


6.1.3 Use Case 3: Island-wide, any-scale storage utilization for fast response ancillary services

Use Case No.	3
Use Case Name	Island-wide, any-scale storage utilization for fast response ancillary services
Use Case General Description	
<p>This use case focus on providing fast balancing services to the grid by capacitating the power system with storage technologies, including small and large-scale BESS, but also very fast responsive assets such as flywheels and other means of very flexible production units as those of Fuel Cells, fully dispatchable. Storage technologies allow to store energy in periods of renewable energy surplus that will be used afterwards to assist the grid by contributing to frequency and voltage control. The intelligent Virtual Power Plant (iVPP) is responsible for coordinating the energy fluxes between the grid and the storage assets.</p>	
System Components Used	
<ol style="list-style-type: none"> 1. Weather Forecast Provider (External Systems/Cloud Services) 2. DSO (External Systems/Cloud Services) 3. Centralized Dispatcher (iVPP) (Kiplo) 4. Forecasting Engine (iVPP) 5. Database (ESB) 6. Aggregation and Intelligent Segmentation (iVPP) 7. Virtual Energy Console (iVPP) 8. Asset Agent (ESB) 9. Asset (Household Batteries, Flywheel, BESS) 	
Workflow of the overall System	
<p>1a. Distribution System Operator (DSO) sends grid data to the Database 1b. The Operator of the iVPP sends to the database constraints regarding the operation of the various assets of the system. 1c. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database.</p>	

- 1d. The asset agents store their data regarding their state (e.g. SoC for BESS) to the database.
2. The Virtual Energy Console gets data from the database where the aggregator can get a clear overview of the storage assets and their condition.
- 3a. The Aggregation and Intelligent Segmentation component gets load measurement data from the database.
- 3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets of the aggregator into clusters based on a predefined objective.
- 3c. The labels of the clustered data are sent to the database.
4. The Virtual Energy Console gets the labels of the clustered data from the database.
- 5a. The Forecasting Engine gets historical data of previous measurements and local meteorological data from the database.
- 5b. The Forecasting Engine gets the labels of the clustered data from the database.
- 5c. The Forecasting Engine computes forecasts for power generation and consumption.
- 5d. The Forecasting Engine sends the forecasted data to the database.
- 6a. The Centralized Dispatcher gets the real time data, grid data, forecasted data, and asset constraints that have been stored in the database.
- 6b. The Centralized Dispatcher computes internally the optimal dispatch of the energy storage assets.
- 6c. The Centralized Dispatcher sends setpoints for the operation of the Household Batteries according to the optimization problem that it has solved.
- 6d. The Centralized Dispatcher sends dispatch setpoints for the operation of the BESSs.
7. The asset agents send the aforementioned dispatch signals to the Household BESSs.

UML Sequence Diagram

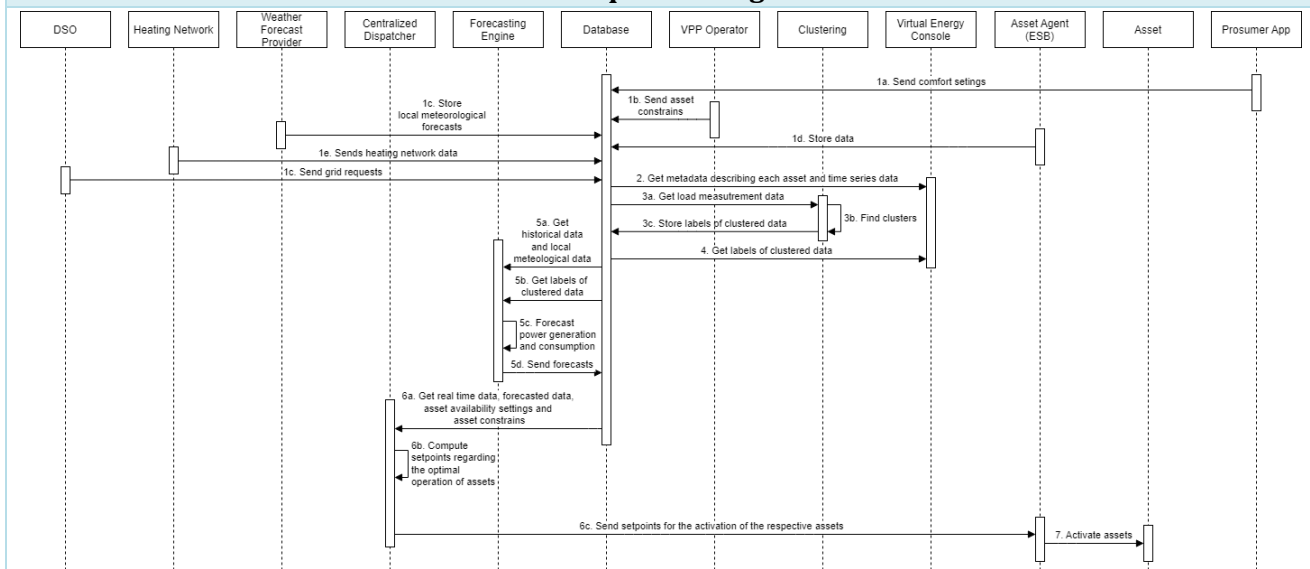


6.1.4 Use Case 4: Demand Side Management and Smart Grid methods to support Power quality and congestion management services

Use Case No.	4
Use Case Name	Demand Side Management and Smart Grid methods to support Power quality and congestion management services
Use Case General Description	
This use case reports the methods to provide slow ancillary services to the power system through demand-side management and smart grid methods. The intelligent Virtual Power Plant (iVPP) performs a global optimization which will consider the 4 following assets: storage assets, fuel cells, hybrid transformer and smart energy router. For each one of these assets, the iVPP computes an optimal setpoint in order to ensure the stability and quality of the power system.	
System Components Used	
<ol style="list-style-type: none"> 1. DSO (External Systems/Cloud Services) 2. Heat Network (External Systems/Cloud Services) 3. Weather Forecast Provider (External Systems/Cloud Services) 4. Centralized Dispatcher (iVPP) (Kiplo, optiMEMS) 5. Forecasting Engine (iVPP) 6. Database (ESB) 7. Aggregation and Intelligent Segmentation (iVPP) 8. Virtual Energy Console (iVPP) 9. Asset Agent (ESB) 10. Assets (Smart Energy Router, Hybrid Transformer, BESS) 	
Workflow of the overall System	
<p>1a. The prosumer sends his/her comfort settings to the database.</p> <p>1b. The Operator of the iVPP sends to the database constraints regarding the operation of the various assets of the system.</p> <p>1c. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database.</p> <p>1d. The asset agents store their data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to the database.</p> <p>1e. Heating network data are stored to the database.</p> <p>1f. The DSO sends grid requests to the database.</p> <p>2. The Virtual Energy Console gets metadata describing each asset and time series data in terms of consumption and generation of each asset from the database where the aggregator can get a clear overview of the assets and their condition.</p> <p>3a. The Aggregation and Intelligent Segmentation component gets load measurement data from the database.</p> <p>3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets of the aggregator into clusters based on a predefined objective.</p> <p>3c. The labels of the clustered data are sent to the database.</p> <p>4. The Virtual Energy Console gets the labels of the clustered data from the database.</p> <p>5a. The Forecasting Engine gets historical data of previous measurements and local meteorological data from to the database.</p> <p>5b. The Forecasting Engine gets the labels of the clustered data from the database.</p> <p>5c. The Forecasting Engine computes forecasts for power generation and consumption</p>	

- 5d. The Forecasting Engine sends the forecasted data to the database.
- 6a. The Centralized Dispatcher gets real time data, grid data, forecasted data and asset constraints that have been stored in the database.
- 6b. The Centralized Dispatcher computes internally the optimal dispatch of the energy assets.
- 6c. The Centralized Dispatcher sends setpoints for the operation of the assets.
7. The asset agents send the aforementioned setpoints to the physical assets.

UML Sequence Diagram



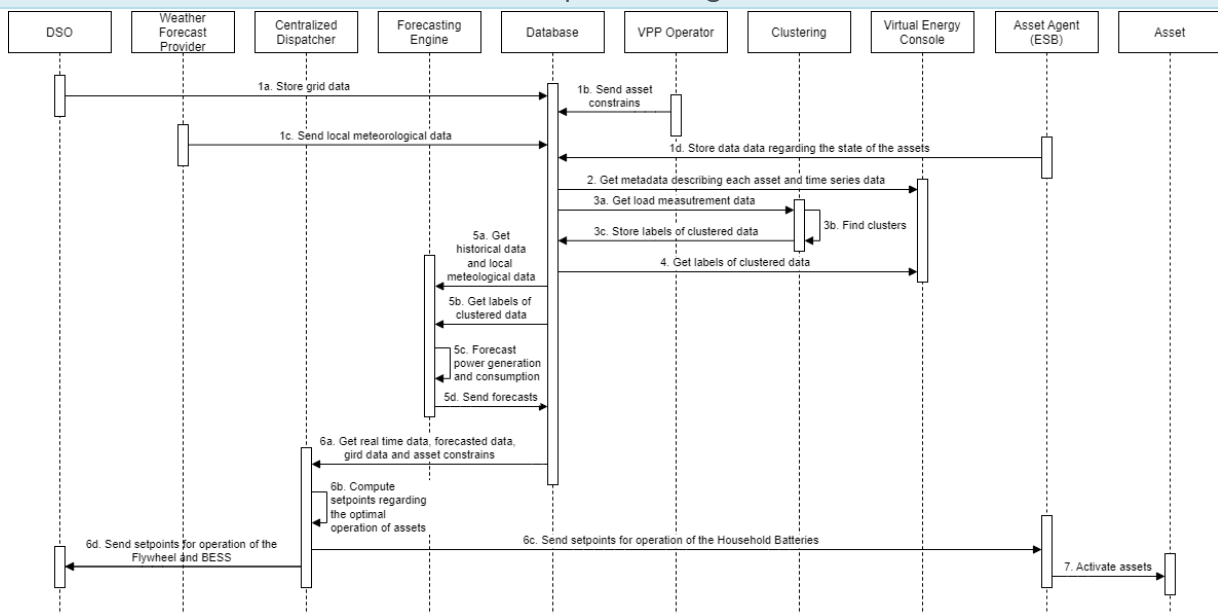
6.1.5 Use Case 5: Decarbonisation of transport and the role of electric mobility in stabilizing the energy system

Use Case No.	5
Use Case Name	Decarbonisation of transport and the role of electric mobility in stabilizing the energy system
Use Case General Description	
This use case focuses on decarbonizing the transport sector by installing energy chargers in the islands. Additionally, using V2G capabilities it demonstrates the provision of grid services utilizing electric vehicle chargers.	
System Components Used	
<ol style="list-style-type: none"> 1. Weather Forecast Provider (External Systems/Cloud Services) 2. DSO (EDA) (External Systems/Cloud Services) 3. Centralized Dispatcher (iVPP)(Kiplo) 4. Forecasting Engine (iVPP) 5. Database (ESB) 6. Aggregation and Intelligent Segmentation (iVPP) 7. Virtual Energy Console (iVPP) 8. Asset Agent (ESB) 9. Asset (V2G Charging station) 	
Workflow of the overall System	
1a. Distribution System Operator (TSO) sends grid data to the Database	



- 1b. The Operator of the iVPP sends to the database constraints regarding the operation of the various assets of the system.
- 1c. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database.
- 1d. The asset agents store their data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to the database.
- 2a. The Virtual Energy Console gets data from the database where the aggregator can get a clear overview of the assets and their condition.
- 3a. The Aggregation and Intelligent Segmentation component gets load measurement data from database.
- 3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets of the aggregator into clusters.
- 3c. The labels of the clustered data are sent to the database.
4. The Virtual Energy Console gets the labels of the clustered data from the database.
- 5a. The Forecasting engine gets historical data of previous measurements and local meteorological data from the database.
- 5b. The Forecasting Engine gets the labels of the clustered data from the database.
- 5c. The Forecasting Engine computes forecasts for energy consumption and production.
- 5d. The Forecasting Engine sends the forecasted data to the database.
- 6a. The Centralized Dispatcher gets the real time data, forecasted data and grid data that have been stored in the database.
- 6b. The Centralized Dispatcher computes internally the optimal dispatch of the V2G charging stations.
- 6c. The Centralized Dispatcher sends setpoints to the V2G Charging stations.
7. The asset agents send the aforementioned setpoints to the V2G charging stations.

UML Sequence Diagram



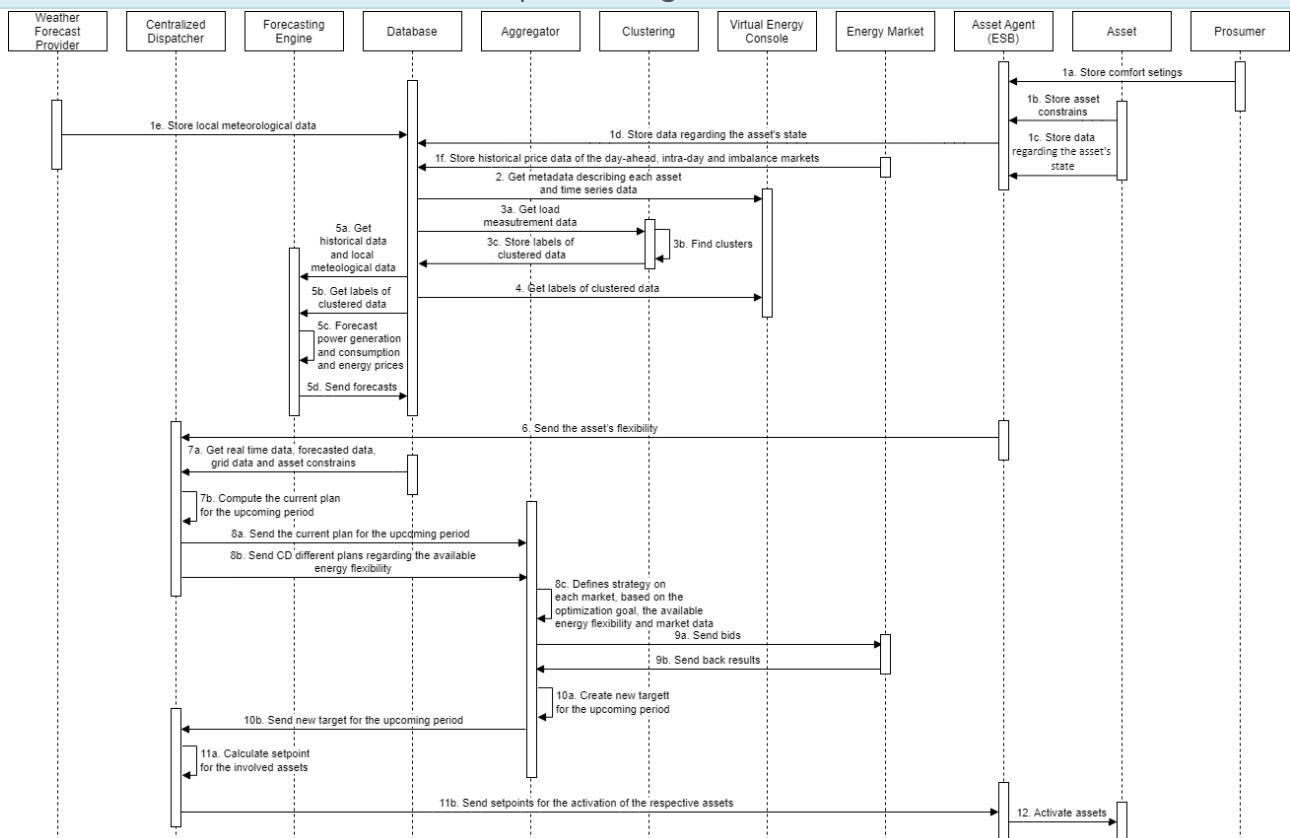
6.2 Ameland pilot island dynamic view

6.2.1 Use Case 1: Community demand-side driven self-consumption maximization

Use Case No.	1
Use Case Name	Community demand-side driven self-consumption maximization
<p>The scope of this Use Case is the optimization of behind-the-meter assets at residential consumer premises to maximize self-consumption from RES and thereby reducing energy curtailment. The ability of monitoring and control loads, PV generation and storage can allow consumers to explore the potential of self-consumption and electricity cost minimization. This Use Case is demonstrated in Local Energy Communities (if LEC already exist in the island) and the optimization of the assets will be performed in a local and a neighbourhood-level.</p>	
System Components Used – Scenario 1	
<ol style="list-style-type: none"> 1. Weather Forecast Provider (External Systems/Cloud Services) 2. Centralized Dispatcher (iVPP) (Reflex) 3. Forecasting Engine (iVPP) 4. Database (ESB) 5. Aggregation and Intelligent Segmentation (iVPP) 6. Virtual Energy Console (iVPP) 7. Energy Market (External Systems/Cloud Services) 8. Asset Agent (ESB) 9. Assets (Solar farm, Hybrid Heat Pumps, Suwotec Battery, Fuel Cell CH₄, Wind Turbines, Residential PV, Battery Pack, Solar Panels, μ CHP) 	
Workflow of the overall System – Scenario 1	
<ol style="list-style-type: none"> 1a. The prosumer stores his/her comfort settings in the Asset Agent. 1b. The assets store their constraints regarding their operation in their respective Asset Agents. 1c. The asset agents store their data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to their Asset Agent. 1d. The asset agents store the asset's data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to the database. 1e. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database 1f. Historical price data of the day-ahead, intra-day and imbalance markets from the energy market are stored in the database. 2. The Virtual Energy Console gets metadata describing each asset and time series data in terms of consumption and generation of each asset from the database. 3a. The Aggregation and Intelligent Segmentation component gets load measurement data from the database. 3b. The Aggregation and Intelligent Segmentation executes clustering algorithms segmenting the assets into clusters based on a predefined objective. 3c. The labels of the clustered data are stored in the database. 4. The Virtual Energy Console gets the labels of the clustered data from the database. 5a. The Forecasting Engine gets the historical data of previous measurements, price data and local meteorological data from the database. 5b. The Forecasting Engine gets the labels of the clustered data from the database. 	

- 5c. The Forecasting Engine computes day-ahead and short-term forecasts for power generation, consumption and price of energy in the day-ahead, intra-day and imbalance markets.
- 5d. The Forecasting Engine sends the forecasted power generation, power consumption and prices to the database.
6. The Asset Agent of each asset sends the asset's flexibility to the Centralized Dispatcher.
- 7a. The Centralized Dispatcher requests real time measurement data, price data, and energy forecasts from the ESB.
- 7b. The Centralized Dispatcher computes the current plan for the upcoming period
- 8a. The Aggregator requests from the CD the current plan for the upcoming period
- 8b. The Aggregator requests from the CD different plans regarding the available energy flexibility
- 8c. The Aggregator defines its strategy on each market, based on the optimization goal, the available energy flexibility and market data (such as the current day-ahead programmed, intra-day data, congestion information)
- 9a. The Aggregator sends bids/schedules towards the relevant energy markets.
- 9b. The Energy Markets send their results back to the aggregator.
- 10a. Based on the outcome of the market interaction, the aggregator creates a new target for the upcoming period.
- 10b. The aggregator sends the new target for the upcoming period to the CD.
- 11a. Based on the received data the CD calculates the setpoints for the involved assets.
- 11b. The CD sends the calculated setpoints to the asset agents for implementation.
12. The asset agents send the aforementioned signals to the Physical Assets to start consuming energy.

UML Sequence Diagram – Scenario 1



System Components Used – Scenario 2

1. Weather Forecast Provider (External Systems/Cloud Services)
2. Centralized Dispatcher (iVPP) (Reflex)
3. DLT-based Transactive Platform(iVPP)
4. Forecasting Engine (iVPP)
5. Database (ESB)
6. Aggregation and Intelligent Segmentation (iVPP)
7. Virtual Energy Console (iVPP)
8. Energy Market (External Systems/Cloud Services)
9. Asset Agent (ESB)
10. Asset
11. Trading Web App

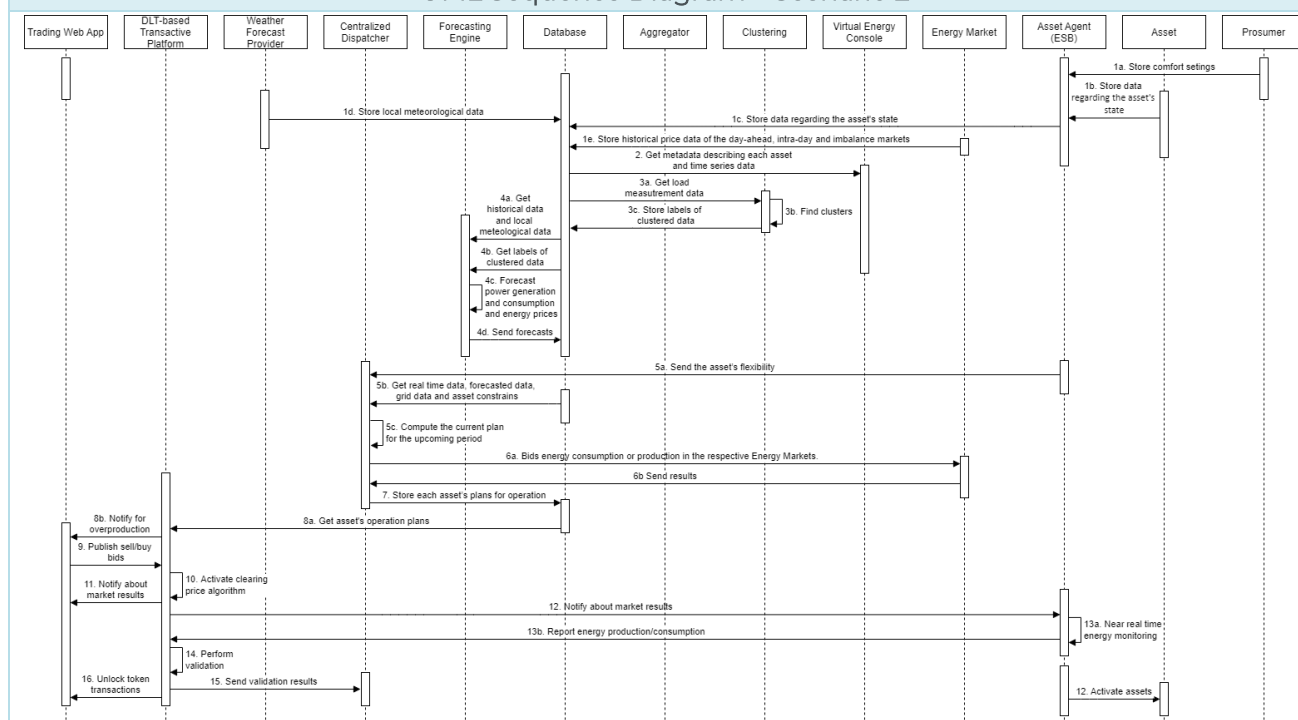
Workflow of the overall System – Scenario 2

- 1a. The prosumer stores his/her comfort settings in the Asset Agent.
- 1b. The asset agents store their data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to their Asset Agent.
- 1c. The asset agents store the asset's data regarding their state (e.g SoC for BESS, Power/Energy output for production assets etc.) to the database.
- 1d. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database
- 1e. Historical price data of the day-ahead, intra-day and imbalance markets from the energy market are stored in the database.
2. The Virtual Energy Console gets metadata describing each asset and time series data in terms of consumption and generation of each asset from the database.
- 3a. The Aggregation and Intelligent Segmentation component gets load measurement data from the database.
- 3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets of the aggregator into clusters based on a predefined objective.
- 3c. The labels of the clustered data are stored in the database.
- 4a. The Forecasting Engine gets historical data of previous measurements, price data and local meteorological data from the database.
- 4b. The Forecasting Engine gets the labels from the clustered data from the database.
- 4c. The Forecasting Engine computes forecasts for power generation, consumption and forecasts the price of energy in the day-ahead, intra-day and imbalance markets.
- 4d. The Forecasting Engine sends the forecasted power generation, power consumption and prices to the database.
- 5a. The Asset Agent of each asset sends the asset's flexibility to the Centralized Dispatcher.
- 5b. The real time data, price data, comfort settings, assets constraints and forecasted data that have been stored in the database are sent to the Centralized Dispatcher.
- 5c. The Centralized Dispatcher computes internally the setpoints regarding the optimal operation of the involved assets.
- 6a. The Centralized Dispatcher based on the optimization process that has solved, bids energy consumption or production in the respective Energy Markets.
- 6b. After gate closure for the Day-Ahead market and during the operation for the Intra-day and Imbalance markets the Energy Markets, send their results back to the Centralized dispatcher.
7. The Centralized Dispatcher stores each asset's plans for operation in the Database.
- 8a. The DLT-based Transactive Platform gets the assets operation plans from the database.
- 8b. DLT-based Transactive Platform notifies the prosumer for overproduction via the Trading Web App.
9. The prosumer publishes sell or buy energy bids to the DLT-based Transactive Platform.
10. The DLT-based Transactive Platform activates the clearing energy price algorithm.
11. The prosumer is notified about the market results from the DLT-based Transactive Platform via the Trading Web App.



12. The DLT-based Transactive Platform also notifies the Asset Agents about the market results.
- 13a. The Asset Agents perform near real time energy monitoring to check whether the energy transactions are being held as agreed.
- 13b. The Asset Agents report the energy production and consumption monitoring back to the DLT-based Transactive Platform.
14. The DLT-based Transactive Platform validates if the energy transactions were held according to the smart contract.
15. The validation results are sent to the Centralized Dispatcher, in order to facilitate any energy transaction deviations that emerged.
16. The DLT-based Transactive Platform unlocks token transactions via the Trading Web App.

UML Sequence Diagram – Scenario 2



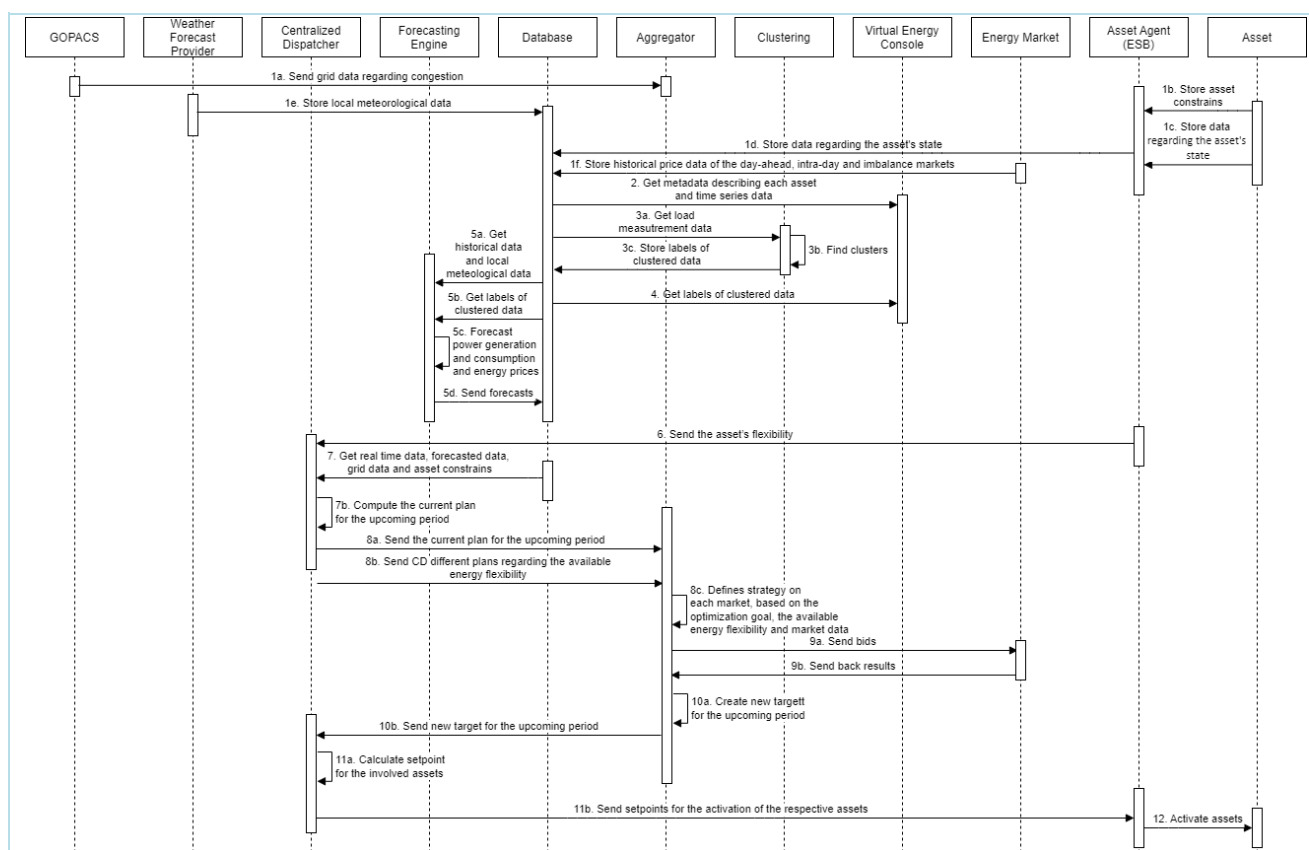
6.2.2 Use Case 2: Community supply-side optimal dispatch and intra-day services provision

Use Case No.	2
Use Case Name	Community supply-side optimal dispatch and intra-day services provision
Use Case General Description	
<p>The present Use Case focuses on using the flexibility on the generation side, for utility-scale assets, to minimize energy curtailment in periods of renewable energy surplus. For this purpose, the intelligent Virtual Power Plant (iVPP) calculates the optimal day ahead dispatch and provides suggestions to the grid-assets with the intention of minimizing curtailment and delivering intra-day services to the grid. Accordingly, the iVPP considers three different types of utility-scale assets for this optimization: i) dispatchable assets, ii) non-dispatchable assets and iii) large-scale storage systems including both BESS and systems producing alternative fuels (electrolyzers), which support the decarbonization of islands with multi-purpose end uses. The iVPP computes the optimal dispatch set-point through provided information and delivers it to the dispatchable assets and large-scale storage systems in order to suggest behaviours to ensure the stability of the power system.</p>	
System Components Used	
<ol style="list-style-type: none"> 1. Electricity Grid (External Systems/Cloud Services) 2. Weather Forecast Provider (External Systems/ Cloud Services) 3. GOPACS (External Systems/Cloud Services) 4. Centralized Dispatcher (iVPP) (Reflex) 5. Forecasting Engine (iVPP) 6. Database (ESB) 7. Aggregation and Intelligent Segmentation (iVPP) 8. Virtual Energy Console (iVPP) 9. Energy Market (External Systems/Cloud Services) 10. Asset Agent (ESB) 11. Assets (Solar Farm, Battery, Electrolyzer) 	
Workflow of the overall System	
<p>1a The Aggregator requests grid data regarding (upcoming) congestion from the GOPACS platform.</p> <p>1b. The assets store their constraints regarding their operation in their respective Asset Agents.</p> <p>1c. The asset agents store their data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to their Asset Agent.</p> <p>1d. The asset agents store the asset's data regarding their state (e.g SoC for BESS, Power/Energy output for production assets etc.) to the database.</p> <p>1e. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database.</p> <p>1f. Historical price data of the day-ahead, intra-day and imbalance markets from the energy market are stored in the database.</p> <p>2. The Virtual Energy Console gets data from the database where the aggregator can get a clear overview of the assets and their condition.</p> <p>3a. The Aggregation and Intelligent Segmentation Component gets load measurement from the database.</p> <p>3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets of the aggregator into clusters based on a predefined objective.</p> <p>3c. The labels of the clustered data are sent to the database.</p>	

4. The Virtual Energy Console gets the labels of the clustered data from the database.
- 5a. The Forecasting Engine gets the historical data of previous measurements, price data and local meteorological data from the database.
- 5b. The Forecasting Engine gets the labels of the clustered data from the database.
- 5c. The Forecasting Engine computes day-ahead and short-term forecasts for power generation, consumption and price of energy in the day-ahead, intra-day and imbalance markets.
- 5d. The Forecasting Engine sends the forecasted power generation, power consumption and prices to the database.
6. The Asset Agent of each asset sends the asset's flexibility to the Centralized Dispatcher.
- 7a. The Centralized Dispatcher requests real time measurement data, price data, and energy forecasts from the ESB.
- 7b. The Centralized Dispatcher computes the current plan for the upcoming period.
- 8a. The Aggregator requests from the CD the current plan for the upcoming period.
- 8b. The Aggregator requests from the CD different plans regarding the available energy flexibility.
- 8c. The Aggregator defines its strategy on each market, based on the optimization goal, the available energy flexibility and market data (such as the current day-ahead programmed, intra-day data, congestion information).
- 9a. The Aggregator sends bids/ schedules towards the relevant energy markets. In case of congestion management through GOPACS, it will attach the GOPACS label to the bids.
- 9b. The Energy Markets send their results back to the aggregator.
- 10a. Based on the outcome of the market interaction, the aggregator creates a new target for the upcoming period.
- 10b. The aggregator sends the new target for the upcoming period to the CD.
- 11a. Based on the received data the CD calculates the setpoints for the involved assets.
- 11b. The CD sends the calculated setpoints to the asset agents for implementation.
12. The asset agents send the aforementioned signals to the Physical Assets to start producing energy.

UML Sequence Diagram





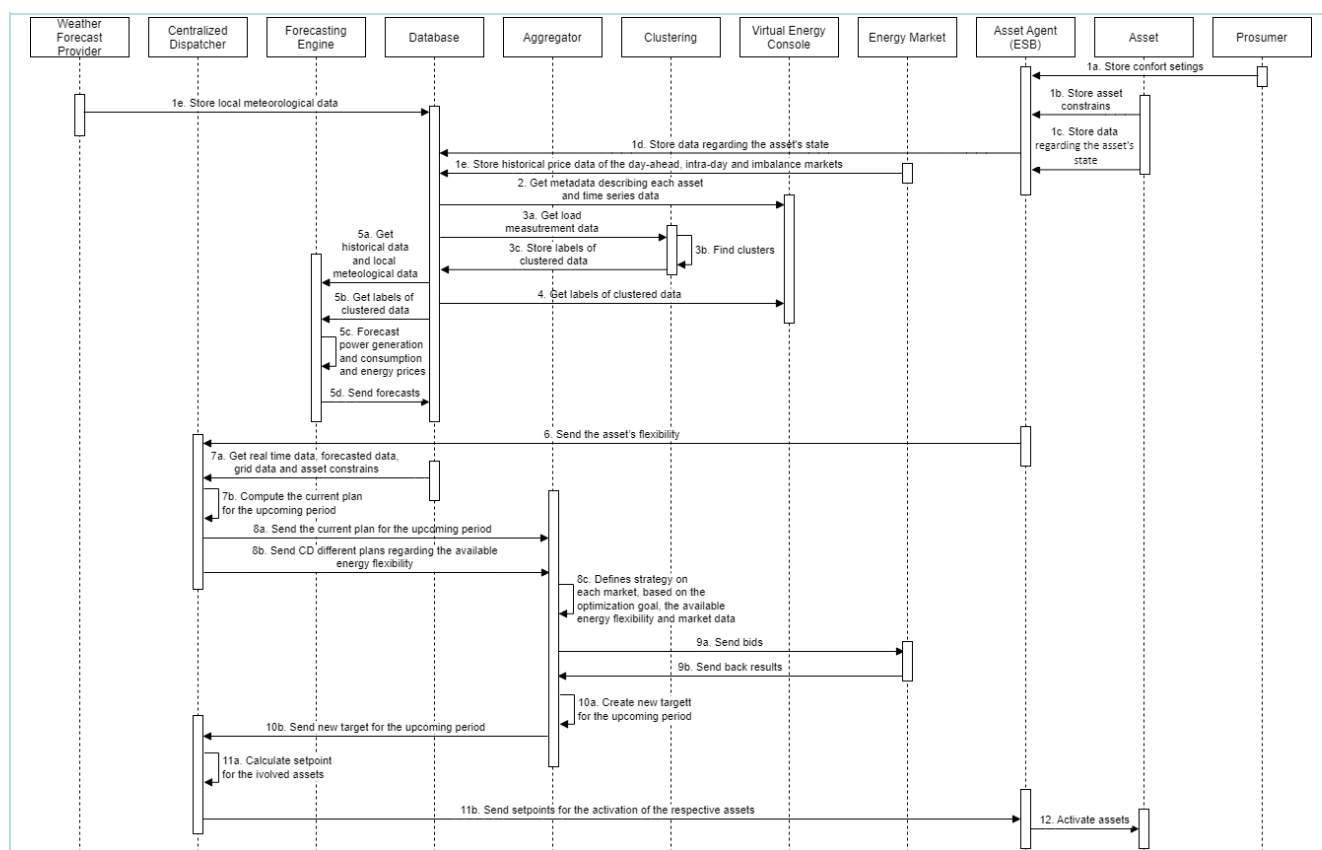
6.2.3 Use Case 3: Island-wide, any-scale storage utilization for fast response ancillary services

Use Case No.	3
Use Case Name	Island-wide, any-scale storage utilization for fast response ancillary services
Use Case General Description	
This Use Case demonstrates the provision of fast ancillary services to the grid, when grid reliability and safety is compromised, through storage systems of any-scale. These storage systems help balancing the power system by either storing energy for later use when there are high levels of energy generation or by providing energy to the grid in periods of high energy demand.	
System Components Used	
<ol style="list-style-type: none"> 1. Weather Forecast Provider (External Systems/Cloud Services) 2. DSO (External Systems/Cloud Services) 3. Centralized Dispatcher (iVPP) (Reflex) 4. Forecasting Engine (iVPP) 5. Database (ESB) 6. Aggregation and Intelligent Segmentation (iVPP) 7. Virtual Energy Console (iVPP) 8. Energy Market (External Systems/ Cloud Services) 9. Asset Agent (ESB) 10. Asset (Battery, Private CHPs, CHP, Fuel Cell) 	
Workflow of the overall System	

- 1a. The prosumer stores his/her comfort settings in the Asset Agent.
- 1b. The assets store their constraints regarding their operation in their respective Asset Agents.
- 1c. The asset agents store their data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to their Asset Agent.
- 1d. The asset agents store the asset's data regarding their state (e.g SoC for BESS, Power/Energy output for production assets etc.) to the database.
- 1e. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database.
- 1f. Historical price data of the day-ahead, intra-day and imbalance markets from the energy market are stored in the database.
2. The Virtual Energy Console gets data from the database where the aggregator can get a clear overview of the assets and their condition.
- 3a. The Aggregation and Intelligent Segmentation Component gets load measurement data from the database.
- 3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets of the aggregator into clusters based on a predefined objective.
- 3c. The labels of the clustered data are sent to the database.
4. The Virtual Energy Console gets the labels of the clustered data from the database.
- 5a. The Forecasting Engine gets the historical data of previous measurements, price data and local meteorological data from the database.
- 5b. The Forecasting Engine gets the labels of the clustered data from the database.
- 5c. The Forecasting Engine computes day-ahead and short-term forecasts for power generation, consumption and price of energy in the day-ahead, intra-day and imbalance markets.
- 5d. The Forecasting Engine sends the forecasted power generation, power consumption and prices to the database.
6. The Asset Agent of each asset sends the asset's flexibility to the Centralized Dispatcher.
- 7a. The Centralized Dispatcher requests real time measurement data, price data, and energy forecasts from the ESB.
- 7b. The Centralized Dispatcher computes the current plan for the upcoming period.
- 8a. The Aggregator requests from the CD the current plan for the upcoming period.
- 8b. The Aggregator requests from the CD different plans regarding the available energy flexibility.
- 8c. The aggregator defines its strategy on the imbalance market, based on the optimization goal, the available energy flexibility and market data (such as the current day-ahead programmed, intra-day data, congestion information).
- 9a. The Aggregator sends flexibility capacity bids/schedules towards the imbalance capacity market.
- 9b. The Energy Markets send their results back to the aggregator.
- 10a. Based on the outcome of the market interaction, the aggregator creates a new target for the upcoming period
- 10b. The aggregator sends the new target for the upcoming period to the CD.
- 11a. Based on the received data the CD calculates the setpoints for the involved assets.
- 11b. The CD sends the calculated setpoints to the asset agents for implementation.
12. The asset agents send the aforementioned signals to the Physical Assets to start their operation.

UML Sequence Diagram





6.2.4 Use Case 4: Demand Side Management and Smart Grid methods to support Power quality and congestion management services

Use Case No.	4
Use Case Name	Demand Side Management and Smart Grid methods to support Power quality and congestion management services
Use Case General Description	
This use case reports the methods to provide slow ancillary services to the power system through demand-side management and smart grid methods. The intelligent Virtual Power Plant (iVPP) performs a global optimization which will consider the 4 following assets: storage assets, fuel cells, hybrid transformer and smart energy router. For each one of these assets, the iVPP computes an optimal setpoint in order to ensure the stability and quality of the power system.	
System Components Used	
<ol style="list-style-type: none"> 1. TSO (External Systems/Cloud Services) 2. Heating Network (External Systems/Cloud Services) 3. Weather Forecast Provider (External Systems/Cloud Services) 4. GOPACS (External Systems/Cloud Services) 5. Centralized Dispatcher (iVPP) (Reflex) 6. Forecasting Engine (iVPP) 7. Database (ESB) 	

8. Aggregation and Intelligent Segmentation (iVPP)
9. Virtual Energy Console (iVPP)
10. Energy Market (External Systems/Cloud Services)
11. Asset Agent (ESB)
12. Assets (Battery, CHP, Fuel Cell, Hybrid Heat Pumps, Suwotec Battery)

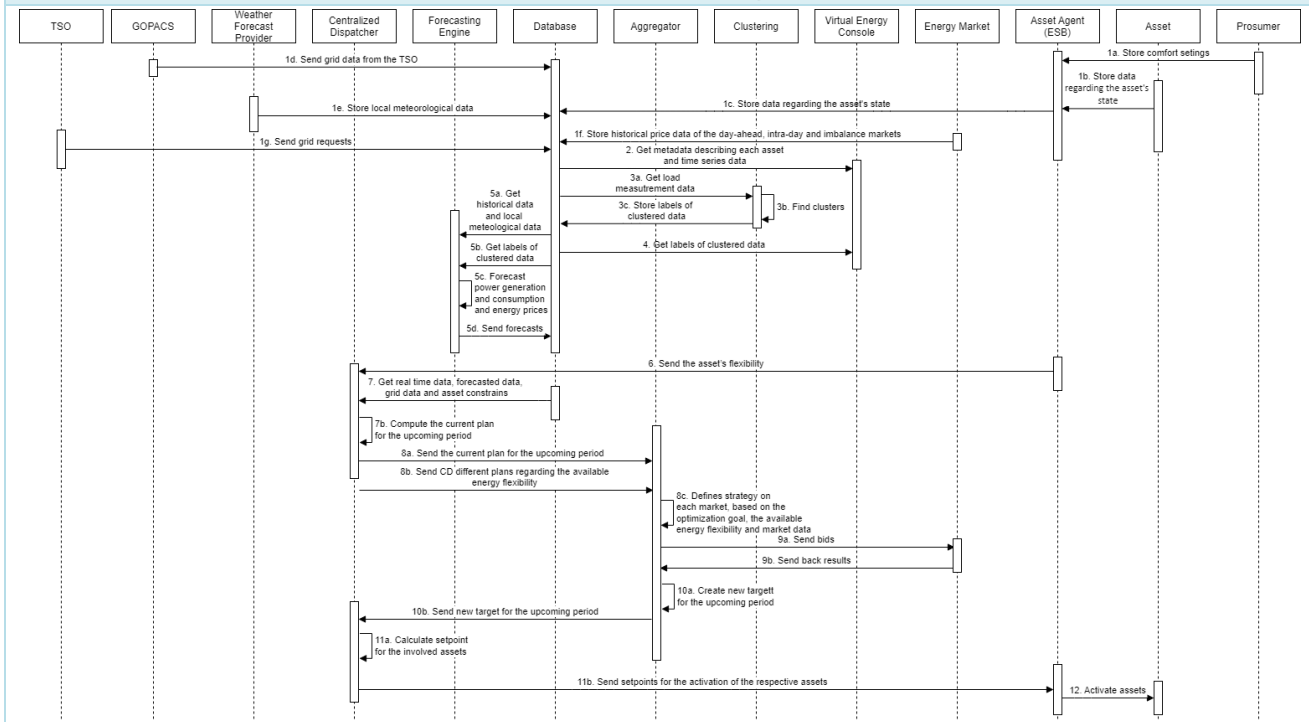
Workflow of the overall System

- 1a. The prosumer stores his/her comfort settings in the Asset Agent.
- 1b. The assets store their data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to their Asset Agent.
- 1c. The asset agents store the asset's data regarding their state (e.g SoC for BESS, Power/Energy output for production assets etc.) to the database.
- 1d. GOPACS sends grid data from the TSO to the Database.
- 1e. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database.
- 1f. Historical price data of the day-ahead, intra-day and imbalance markets from the energy market are stored in the database.
- 1g. The TSO sends grid requests to the database.
- 1h. The Heating Network sends heating network data to the database.
2. The Virtual Energy Console gets data from the database where the aggregator can get a clear overview of the assets and their condition.
- 3a. The Aggregation and Intelligent Segmentation component gets load measurement data from the database.
- 3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets of the aggregator into clusters based on a predefined objective.
- 3c. The labels of the clustered data are sent to the database.
4. The Virtual Energy Console gets the labels of the clustered data from the database.
- 5a. The Forecasting Engine gets the historical data of previous measurements, price data and local meteorological data from the database.
- 5b. The Forecasting Engine gets the labels of the clustered data from the database.
- 5c. The Forecasting Engine computes day-ahead and short-term forecasts for power generation, consumption and price of energy in the day-ahead, intra-day and imbalance markets.
- 5d. The Forecasting Engine sends the forecasted power generation, power consumption and prices to the database.
6. The Asset Agent of each asset sends the asset's flexibility to the Centralized Dispatcher.
- 7a. The Centralized Dispatcher requests real time measurement data, price data, and energy forecasts from the ESB.
- 7b. The Centralized Dispatcher computes the current plan for the upcoming period.
- 8a. The Aggregator requests from the CD the current plan for the upcoming period.
- 8b. The Aggregator requests from the CD different plans regarding the available energy flexibility.
- 8c. The aggregator defines its strategy on the imbalance market, based on the optimization goal, the available energy flexibility and market data (such as the current day-ahead programmed, intra-day data, congestion information).
- 9a. The Aggregator sends flexibility capacity bids/ schedules towards the imbalance capacity market.
- 9b. The Energy Markets send their results back to the aggregator.
- 10a. Based on the outcome of the market interaction, the aggregator creates a new target for the upcoming period.
- 10b. The aggregator sends the new target for the upcoming period to the CD.
- 11a. Based on the received data the CD calculates the setpoints for the involved assets.



11b. The CD sends the calculated setpoints to the asset agents for implementation.
 12. The asset agents send the aforementioned signals to the Physical Assets to start their operation.

UML Sequence Diagram



6.2.5 Use Case 5: Decarbonisation of transport and the role of electric mobility in stabilizing the energy system

Use Case No.	5
Use Case Name	Decarbonisation of transport and the role of electric mobility in stabilizing the energy system
Use Case General Description	
This Use Case aims to define a roadmap to reach decarbonization in the transport sector on the islands. Moreover, it explores the growth potential of EV chargers with or without V2G capabilities and smart charging schemes. The Intelligent Virtual Power Plant (iVPP) will manage the power flows of these chargers in order to ensure the stability of the power system	
System Components Used	
<ol style="list-style-type: none"> 1. Weather Forecast Provider (External Systems/Cloud Services) 2. TSO (External Systems/Cloud Services) 3. Centralized Dispatcher (iVPP) (Reflex) 4. Forecasting Engine (iVPP) 5. Database (ESB) 6. Aggregation and Intelligent Segmentation (iVPP) 7. Virtual Energy Console (iVPP) 8. Energy Market (External Systems/Cloud Services) 9. Asset Agent (ESB) 10. Assets (Electrolyzer, H2 Watertaxi, H2 Infrastructure, Charging Station EV) 	

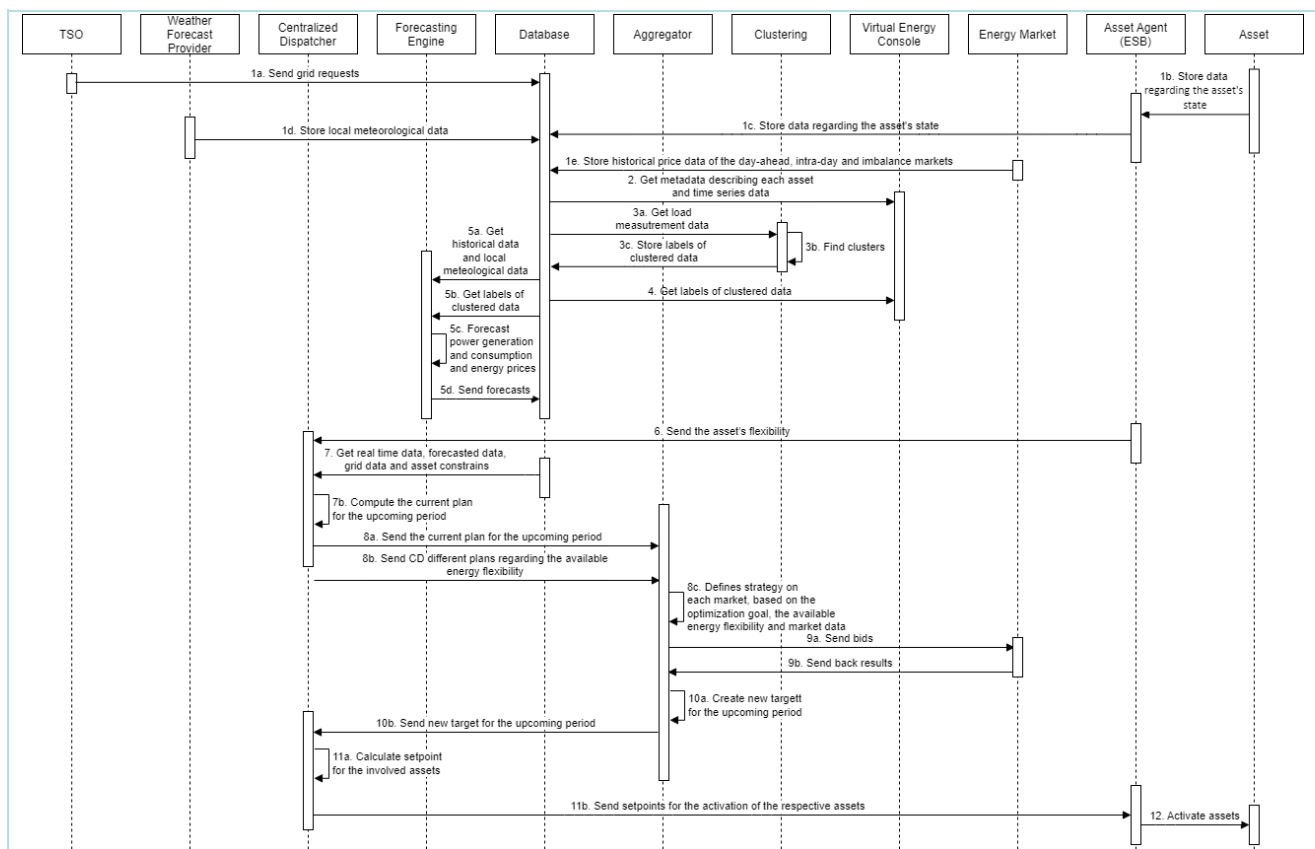


Workflow of the overall System

- 1a. Transmission System Operator (TSO) sends grid requests to the Database.
- 1b. The assets store their data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to their Resource Manager.
- 1c. The asset agents store their asset's data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to the database.
- 1d. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database.
- 1e. Historical price data of the day-ahead, intra-day and imbalance markets from the energy market are stored in the database.
2. The Virtual Energy Console gets data from the database where the aggregator can get a clear overview of the assets and their condition.
- 3a. The Aggregation and Intelligent Segmentation component gets load measurement data from the database.
- 3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets of the aggregator into clusters based on a predefined objective.
- 3c. The labels of the clustered data are sent to the database.
4. The Virtual Energy Console gets the labels of the clustered data from the database.
- 5a. The Forecasting Engine gets the historical data of previous measurements, price data and local meteorological data from the database.
- 5b. The Forecasting Engine gets the labels of the clustered data from the database.
- 5c. The Forecasting Engine computes day-ahead and short-term forecasts for power generation, consumption and price of energy in the day-ahead, intra-day and imbalance markets.
- 5d. The Forecasting Engine sends the forecasted power generation, power consumption and prices to the database.
6. The Asset Agent of each asset sends the asset's flexibility to the Centralized Dispatcher.
- 7a. The Centralized Dispatcher requests real time measurement data, price data, and energy forecasts from the ESB.
- 7b. The Centralized Dispatcher computes the current plan for the upcoming period.
- 8a. The Aggregator requests from the CD the current plan for the upcoming period.
- 8b. The Aggregator requests from the CD different plans regarding the available energy flexibility.
- 8c. The aggregator defines its strategy on the imbalance market, based on the optimization goal, the available energy flexibility and market data (such as the current day-ahead programmed, intra-day data, congestion information)
- 9a. The Aggregator sends flexibility capacity bids/ schedules towards the imbalance capacity market.
- 9b. The Energy Markets send their results back to the aggregator.
- 10a. Based on the outcome of the market interaction, the aggregator creates a new target for the upcoming period.
- 10b. The aggregator sends the new target for the upcoming period to the CD.
- 11a. Based on the received data the CD calculates the setpoints for the involved assets.
- 11b. The CD sends the calculated setpoints to the asset agents for implementation.
12. The asset agents send the aforementioned signals to the Physical Assets to start their operation.

UML Sequence Diagram





6.2.6 Use Case 6: Decarbonizing large industrial continuous loads through electrification and locally induced generation

Use Case No.	6
Use Case Name	Decarbonizing large industrial continuous loads through electrification and locally induced generation
Use Case General Description	
<p>The present use case aims to decarbonize large industrial continuous and power intensive energy consumers, either located in the island or interconnected as in the case of the AWG natural gas platform off the coast of Ameland. The electrification and local renewable generation will be the main drivers to reach decarbonization in this site and will allow the maximization of renewable sources in the local grid.</p>	
System Components Used	
<ol style="list-style-type: none"> 1. Natural gas platform (External Systems/Cloud Services) 2. Electricity grid (External Systems/Cloud Services) 3. Weather Forecast Provider (External Systems/Cloud Services) 4. GOPACS (External Systems/Cloud Services) 5. Centralized Dispatcher (iVPP) (Reflex) 6. Forecasting Engine (iVPP) 7. Database (ESB) 8. Aggregation and Intelligent Segmentation (iVPP) 	



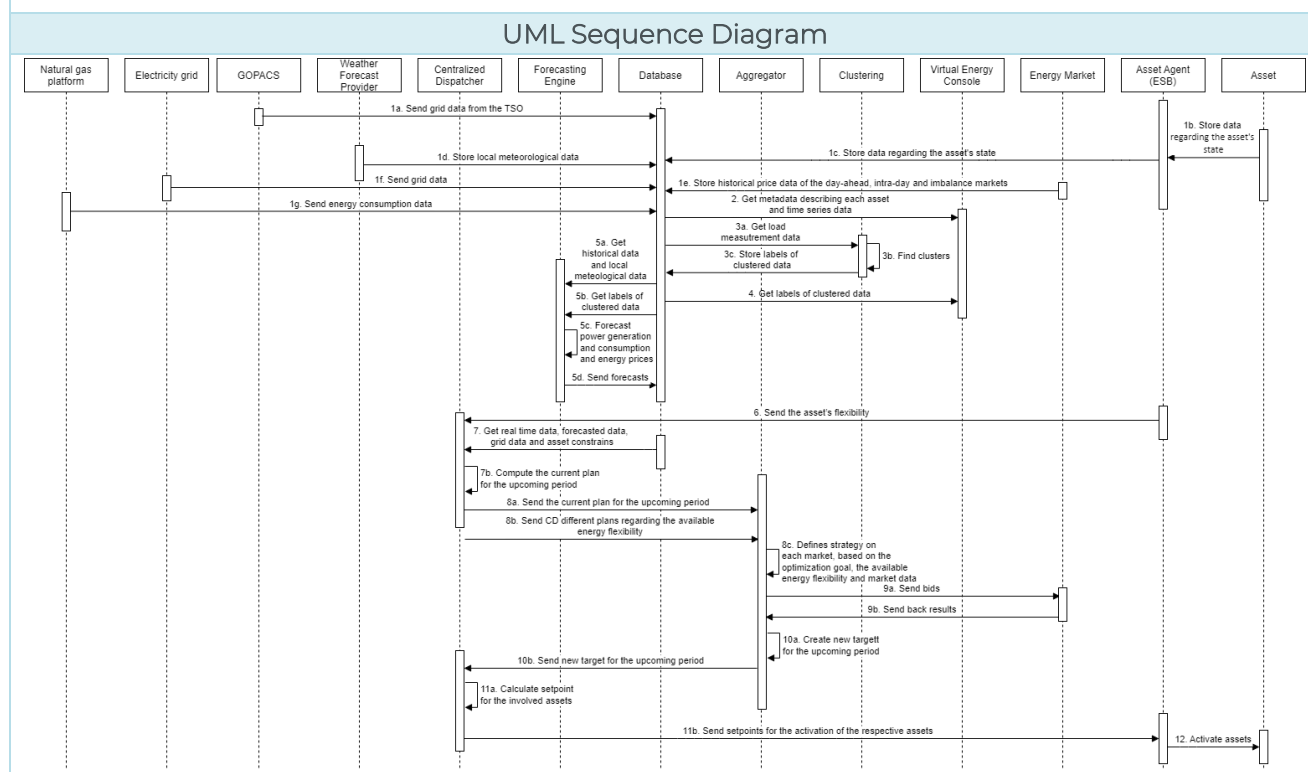
9. Virtual Energy Console (iVPP)
10. Energy Market (External Systems/Cloud Services)
11. Asset Agent (ESB)
12. Assets (Solar Farm 6 MW, Solar Farm 3 MW, 500 kW Tidal Kite, Fuel Cell, Wind Turbines, CHP)

Workflow of the overall System

- 1a. GOPACS sends grid data from the TSO to the Database
- 1b. The assets store their asset's data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to their asset agents.
- 1c. The asset agents store their asset's data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to the database.
- 1d. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database.
- 1e. Historical price data of the day-ahead, intra-day and imbalance markets from the energy market are stored in the database.
- 1f. The Electricity grids send grid data to the database
- 1g. The natural gas platform sends energy consumption data to the database.
2. The Virtual Energy Console gets data from the database where the aggregator can get a clear overview of the assets and their condition.
- 3a. The Aggregation and Intelligent Segmentation component gets load measurement data from the database.
- 3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets of the aggregator into clusters according to a predefined objective.
- 3c. The labels of the clustered data are sent to the database.
4. The Virtual Energy Console gets the labels of the clustered data from the database.
- 5a. The Forecasting Engine gets the historical data of previous measurements, price data and local meteorological data from the database.
- 5b. The Forecasting Engine gets the labels of the clustered data from the database.
- 5c. The Forecasting Engine computes day-ahead and short-term forecasts for power generation, consumption and price of energy in the day-ahead, intra-day and imbalance markets.
- 5d. The Forecasting Engine sends the forecasted power generation, power consumption and prices to the database.
6. The Asset Agent of each asset sends the asset's flexibility to the Centralized Dispatcher.
- 7a. The Centralized Dispatcher requests real time measurement data, price data, and energy forecasts from the ESB.
- 7b. The Centralized Dispatcher computes the current plan for the upcoming period.
- 8a. The Aggregator requests from the CD the current plan for the upcoming period.
- 8b. The Aggregator requests from the CD different plans regarding the available energy flexibility.
- 8c. The aggregator defines its strategy on the imbalance market, based on the optimization goal, the available energy flexibility and market data (such as the current day-ahead programmed, intra-day data, congestion information).
- 9a. The Aggregator sends flexibility capacity bids/ schedules towards the imbalance capacity market.
- 9b. The Energy Markets send their results back to the aggregator.
- 10a. Based on the outcome of the market interaction, the aggregator creates a new target for the upcoming period.
- 10b. The aggregator sends the new target for the upcoming period to the CD.
- 11a. Based on the received data the CD calculates the setpoints for the involved assets.
- 11b. The CD sends the calculated setpoints to the asset agents for implementation.



12. The asset agents send the aforementioned signals to the Physical Assets to start their operation.

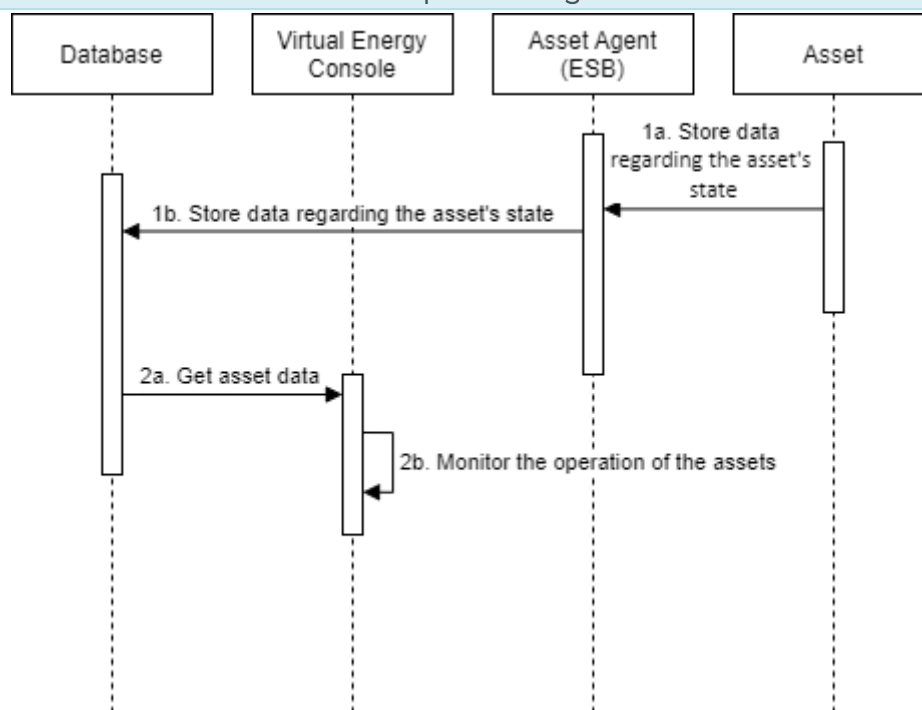


6.2.7 Use case 7: Circular economy, utilization of waste streams and gas grid decarbonization

Use Case No.	7
Use Case Name	Circular economy, utilization of waste streams and gas grid decarbonization
Use Case General Description	
<p>The present use case describes how waste streams are used to produce renewable energy and help to decarbonize the local grid, either for electricity production and/or heating purposes, using green natural gas. Therefore, a demonstration of a digester will occur at Ameland to exploit the potential of converting organic waste into green natural gas, while hydrogen produced from the Electrolyser (using excess of RES) can be used to upgrade the remaining CO₂ in the digester to natural gas. Moreover, an investigation regarding the potential of technologies to process biomass for using the remaining streams is also performed.</p>	
System Components Used	
<ol style="list-style-type: none"> 1. Database (ESB) 2. Virtual Energy Console (iVPP) 3. Asset Agent (ESB) 4. Assets (Electrolyzer) 	
Workflow of the overall System	
<p>1a. The assets store their data regarding their state (e.g SoC for BESS, Power/Energy output for production assets etc.) to their asset agent.</p>	

- 1b. The asset agent sends the asset's data to the database
 2a. The Virtual Energy Console gets the asset's data from the database
 2b. The Virtual Energy Console monitors the operation of the assets

UML Sequence Diagram



6.2.8 Use case 8: Decarbonization of heating network

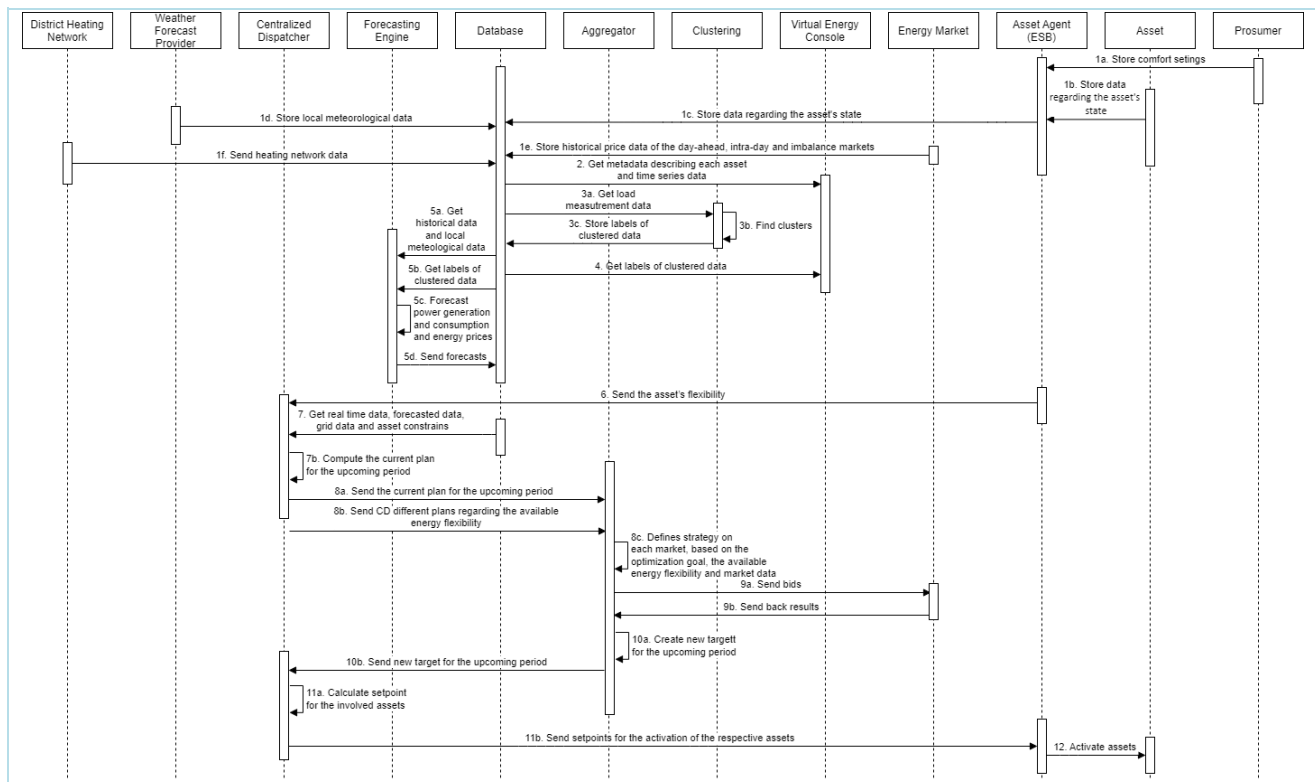
Use Case No.	8
Use Case Name	Decarbonization of heating network
Use Case General Description	
The scope of this Use Case is to decarbonize the heating network in Ameland which currently runs on natural gas. For this purpose, this Use Case focuses on the installation of equipment that allows the reduction of emissions such as hybrid heat pumps to be powered by local RES. Moreover, it also explores further possibilities to phase-out natural gas of certain villages.	
System Components Used	
<ol style="list-style-type: none"> 1. District Heating Network (External Systems/Cloud Services) 2. Weather Forecast Provider (External Systems/Cloud Services) 3. Centralized Dispatcher (iVPP) (Reflex) 4. Forecasting Engine (iVPP) 5. Database (ESB) 6. Aggregation and Intelligent Segmentation (iVPP) 7. Virtual Energy Console (iVPP) 8. Energy Market (External Systems/Cloud Services) 9. Asset Agent (ESB) 10. Assets (Fuel Cell, Suwotec Battery, Hydrogen Storage, Hybrid Heat Pumps) 	
Workflow of the overall System	
1a. The prosumer stores his/her comfort settings in the Asset Agent.	



- 1b. The assets store their data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to their asset agent.
- 1c. The asset agents store their asset's data regarding their state (e.g. SoC for BESS, Power/Energy output for production assets etc.) to the database.
- 1d. Local Meteorological Forecast data from the Weather Forecast Provider are stored in the Database.
- 1e. Historical price data of the day-ahead, intra-day and imbalance markets from the energy market are stored in the database.
- 1f. The district heating network send heating network data to the database.
- 2. The Virtual Energy Console gets data from the database where the aggregator can get a clear overview of the assets and their condition.
- 3a. The Aggregation and Intelligent Segmentation component gets load measurement data from the database.
- 3b. The Aggregation and Intelligent Segmentation component executes clustering algorithms segmenting the assets of the aggregator into clusters based on a predefined objective.
- 3c. The labels of the clustered data are sent to the database.
- 4. The Virtual Energy Console gets the labels of the clustered data from the database.
- 5a. The Forecasting Engine gets the historical data of previous measurements, price data and local meteorological data from the database.
- 5b. The Forecasting Engine gets the labels of the clustered data from the database.
- 5c. The Forecasting Engine computes day-ahead and short-term forecasts for power generation, consumption and price of energy in the day-ahead, intra-day and imbalance markets.
- 5d. The Forecasting Engine sends the forecasted power generation, power consumption and prices to the database.
- 6. The Asset Agent of each asset sends the asset's flexibility to the Centralized Dispatcher.
- 7a. The Centralized Dispatcher requests real time measurement data, price data, and energy forecasts from the ESB.
- 7b. The Centralized Dispatcher computes the current plan for the upcoming period.
- 8a. The Aggregator requests from the CD the current plan for the upcoming period.
- 8b. The Aggregator requests from the CD different plans regarding the available energy flexibility.
- 8c. The aggregator defines its strategy on the imbalance market, based on the optimization goal, the available energy flexibility and market data (such as the current day-ahead programmed, intra-day data, congestion information).
- 9a. The Aggregator sends flexibility capacity bids/schedules towards the imbalance capacity market.
- 9b. The Energy Markets send their results back to the aggregator.
- 10a. Based on the outcome of the market interaction, the aggregator creates a new target for the upcoming period
- 10b. The aggregator sends the new target for the upcoming period to the CD.
- 11a. Based on the received data the CD calculates the setpoints for the involved assets.
- 11b. The CD sends the calculated setpoints to the asset agents for implementation.
- 12. The asset agents send the aforementioned signals to the Physical Assets to start their operation.

UML Sequence Diagram





6.2.9 Use case 9: Active Citizen and LEC Engagement into Decarbonization Transition

Table 6-9: UC9 dynamic view table

Use Case No.	9
Use Case Name	Active Citizen and LEC Engagement into Decarbonization Transition
Use Case General Description	
<p>This Use Case aims to promote an active role and engagement of the community in the island's energy transition. Accordingly, it uses Local Energy Cooperatives to fulfil this purpose where various strategies will be applied, such as involving the community in DSM programs and raising the customer's environmental and energy efficiency awareness through dissemination actions for local homeowners and young people.</p>	

7 Deployment View

In the following section the deployment view of IANOS system architecture is going to be provided, which presents the deployment of software into hardware, as well as detailed data interfaces and the system's general deployment state. It presents the different physical equipment that are going to be used to assist the systems functionality. Additionally, UML deployment views of the system architecture is going to be presented.

Table 7-1: IANOS system detailed data interfaces

Component	Interface/ Data Item	Read/ Write	Related data model/ standard	Data Received from	Data Sent to	Data Format	Measureme nt Unit/ Sampling Rate	Communic ation Interface
VERIFY-D platform tool (LCA and LCC module)	KPIs relevant to the LCA/LCC analysis	W	HTTP(REST , MQTT		IEPT, CrowdEqu ity platform	JSON	Every 60 min	HTTP REST services for data exchange and MQTT protocol for communica tion purposes with third parties.
CrowdEqui ty Component	Input: • LCC performance • Tokenized Energy data	W	HTTP(REST , MQTT		IEPT			HTTP REST services for data exchange and MQTT protocol for communica tion purposes with third parties.
CBA Component	Input: • LCC performance • KPIs calculated by Grid- Optimizer • KPIs calculated by System Modeller	W	HTTP(REST)		IEPT	JSON		HTTP REST services for data exchange
System Modeller	Input: • Descriptio n of Energy System as ESDL	W/R	HTTP(REST)	Input: - File system, ESDL Drive Output: - Simulation storage	ESSIM IETP Suite & MapEdit or	ESDL (XML) JSON	-	HTTP REST to start ESSIM simulation. Read results from the Simulation

Component	Interface/ Data Item	Read/ Write	Related data model/ standard	Data Received from	Data Sent to	Data Format	Measureme nt Unit/ Sampling Rate	Communic ation Interface
	<ul style="list-style-type: none"> Simulation configuration Output: Simulation results 							storage and view in the MapEditor or IETP Suite
Grid-Oriented Optimizer – INTEMA.grid	KPIs relevant to energy analysis	W	HTTP(REST)		IEPT	JSON	Every 60 min	HTTP REST services for communication purposes with the rest of the platform
Aggregation and Classification	Input: <ul style="list-style-type: none"> Raw sensor data Output: <ul style="list-style-type: none"> Clusters of energy portfolio 	R/W	HTTP(REST), AMQP	ESB	iVPP	JSON XML CSV	Labels of clusters	HTTP REST services for communication purposes with the rest of the platform A publish-subscribe mechanism (AMQP) for communication with third parties may be supported.
Forecasting Engine	Input: <ul style="list-style-type: none"> Historical demand/generation data Price data Weather data Output: <ul style="list-style-type: none"> Production, consumption, price forecast 	R/W	HTTP(REST), AMQP	ESB External systems cloud services	iVPP	JSON XML CSV	kWh/ Every 15 min to 1 day	HTTP REST services for communication purposes with the rest of the platform A publish-subscribe mechanism (AMQP) for communication with third parties may be supported.
KIPLO core Platform	Input: <ul style="list-style-type: none"> Production/consumption forecasts Real-time data 	R/W	HTTP (REST), MQTT	ESB, Forecasting Engine, Aggregation and Classification, OptiMEMS, External systems cloud services	IVPP Virtual Energy Console	JSON	15 min	HTTP REST Service for communication purposes with the rest of the platform

Component	Interface/ Data Item	Read/ Write	Related data model/ standard	Data Received from	Data Sent to	Data Format	Measurement Unit/ Sampling Rate	Communication Interface
	<ul style="list-style-type: none"> Price data Constraints End-user preferences <p>Output:</p> <ul style="list-style-type: none"> Setpoints for optimal dispatch schedule 							
OptiMEMS	<p>Input:</p> <ul style="list-style-type: none"> Production/consumption forecasts Real-time data Price data Constraints End-user preferences <p>Output:</p> <ul style="list-style-type: none"> Setpoints for optimal dispatch schedule 	R/W	HTTP(REST), AMQP	ESB, Forecasting Engine, External systems cloud services	iVPP	JSON XML CSV	Setpoints/ Every 15, 30, 60 min.	HTTP REST services for communication purposes with the rest of the platform A publish-subscribe mechanism (AMQP) for communication with third parties may be supported.
Reflex	<p>Input:</p> <ul style="list-style-type: none"> Flexibility of device Forecasts Market information <p>Output</p> <ul style="list-style-type: none"> Dispatch Market allocation 	R/W	HTTP(REST)	<p>Flexible Devices, Forecast service/ESB Markets</p> <p>ReFlex</p>	<p>ReFlex</p> <p>Flexible devices Markets</p>	<p>S2 (JSON) JSON JSON</p> <p>S2 (JSON) JSON</p>	<p>Event based, Configurable in the S2 protocol</p> <p>Event based</p>	<p>Via HTTP REST S2 messages (EN 50491-12-2) are send to ReFlex. Via ESB or HTTP REST forecast information is received for Reflex Market information send from markets to ReFlex S2 interface to flexible device for dispatch Market allocation</p>

Component	Interface/ Data Item	Read/ Write	Related data model/ standard	Data Received from	Data Sent to	Data Format	Measurement Unit/ Sampling Rate	Communication Interface
								interface to market
DLT-based Transactive Platform	Input: <ul style="list-style-type: none"> • Overproduction notification • Historical production / consumption data Output: <ul style="list-style-type: none"> • P2P market results 	R/W	HTTP REST	Forecasting Engine ESB	Centralized Dispatcher (iVPP)	JSON	TBD	HTTP REST services
GOPACS	Trader to Market	R/W	HTTP (REST)	GOPACS	Trader (ReFlex)	JSON	TBD	GOPACS REST API to read congestion information and place a bid.
Virtual Energy Console	UI Dashboard for monitoring the whole VPP operation	R/W	HTTP (REST)	Kiplo Core	Client Machine	JSON	Selected by the operator	HTTP REST Service for communication with Kiplo Core
IANOS Secured Enterprise Service Bus	Input/Output: <ul style="list-style-type: none"> • Energy field devices • Historical data • Real time DB information • IoT Broker Service information 	R/W	HTTP (REST)	Kiplo Core, Aggregation and Classification, Forecasting Engine, Weather, Blockchain, Opti MEMS, Network Energy Assets, dEF-Pi	Kiplo Core, Aggregation and Classification, Forecasting Engine, Weather, Blockchain, Opti MEMS, Network Energy Assets, dEF-Pi	JSON, CSV	Dependent on the information sent/received (TBD)	The platform supports different technologies for the bulk data ingestion: API, REST, MQTT, AMQP, WS DDP, NATS, CoAP, Cron ETL, EFI
dEF-Pi	Input: <ul style="list-style-type: none"> • Production/consumption forecasts • Real-time data 	R/W	HTTP(Rest), MQTT, OCPI	The energy-devices itself	dEF-Pi platform, Reflex, Gopacs	Preferably JSON	Depends on the capabilities of the energy devices	

Component	Interface/ Data Item	Read/ Write	Related data model/ standard	Data Received from	Data Sent to	Data Format	Measureme nt Unit/ Sampling Rate	Communic ation Interface
	Output: • S2/EFI- interface							
BackEnd Agent	Input/Output: • Historical data • Real-time data • Set-points	R/W	HTTP(Rest) , MQTT	ESB, iVPP components	ESB, iVPP components	JSON	Dependent on the information sent/received	HTTP REST services for data exchange and MQTT protocol for communication purposes with third parties.

Table 7-2: IANOS system general deployment state

Component	Involved partners	Current TRL	Hardware requirements	Software requirements	Licence	Installatio n location	Other comme nts/ Installati on require- ments
VERIFY-D platform tool (LCA and LCC module)	CERTH	6	i7 CPU, 16GB RAM, 2TB hard disk, Ethernet connection	Front-end: HTML5, CSS Back-end: Ruby 2.7.0, Rails 6.0.3, Python 3.8.6, Javascript		Physical Server	
CrowdEquity Component	CERTH	3	2 processor cores, 2.2 GHz processor, 16 GB RAM	Blockchain technology, platform services	Propriet ary	Physical Server	
System Modeller	TNO	6	Cloud environment	Docker	Apache 2.0	Cloud server	
Grid- Oriented Optimizer – INTEMA.grid	CERTH	7	i7 CPU, 16GB RAM, 2TB hard disk, Ethernet connection	Front-end: HTML5, CSS Javascript Back-end: Ruby, Rails, Python, Modelica		Physical Server	
CBA Component	UBE	3	i7 CPU, 16GB RAM, 2TB hard disk, Ethernet connection	Back-end: Python 3.8.6	Propriet ary	Cloud or Physical Server	
Aggregation and Classification	CERTH	6	4 Processor cores, 3GHz processor, 32 GB RAM	Python libraries and runtime environment.	Propriet ary	Cloud server	
Forecasting Engine	CERTH	6	4 Processor cores, 2.2GHz processor, 32 GB RAM	Python libraries and runtime environment.	Propriet ary	Cloud server	



Component	Involved partners	Current TRL	Hardware requirements	Software requirements	Licence	Installation location	Other comments/Installation requirements
KIPLO core Platform	CWD	6	VCPU: 4 RAM: 16GB Disk: 500GB	OS: Microsoft Windows Server 2016 64bits, running IIS DB: Microsoft SQL Server 2019 Standard	Proprietary	Cloud or Physical Server	
OptiMEMS	CERTH	7	4 Processor cores, 3.2GHz processor, 16 GB RAM, 512MB HDD, 128MB SSD	Python libraries and runtime environment.	Proprietary	Cloud or Physical Server	
Reflex	TNO, NEROA	6	Fast server hardware	Docker	Proprietary	Cloud or Physical Server	
DLT-based Transactive Platform	ENG	5	4 Processor cores, 16GB RAM	Back-end: Nodejs, Express, Mongoose, Solidity, Truffle Front-end: Vuejs Deploy: Docker	Proprietary	Cloud or Physical Server	
Virtual Energy Console	CWD	6	Access via any web browser or mobile app	Any operating System (Windows, Linux, Android or iOS)	Proprietary	Client Machine	
IANOS Secured Enterprise Service Bus	ETRA	6	Cloud environment	The different microservices are integrated via API REST	Proprietary	Docker SWARN in a LINUX server PC	
dEF-Pi	Neroa	5	Raspberry Pi	Any operating system (Windows or Linux family) with Docker framework	Open-source, Apache 2	Cloud or Physical Server	Additional Hardware requires depending on connected devices
BackEnd Agent	CERTH	5	4 Processor cores, 2.2GHz processor, 8 GB RAM	Python libraries and runtime environment.	Proprietary	Physical Server	

7.1 Business Layer

7.1.1 VERIFY-D platform tool (LCA and LCC module)

The VERIFY-D is a web-based platform tool that performs dynamic environmental and economic life cycle assessment activities. The methodological approach follows the ISO14044, including the manufacturing (i.e., CO2 emissions and capital expenditures) and operational (e.g., consumption, production) stages of the involved energy-related technologies or systems in the LH islands. The environmental and cost assessment platform tool will be installed in a dedicated server machine, providing continuous access from and to the IEPT as well as to the Crowdequity component. The server machine must be always connected to the internet through Ethernet cable.

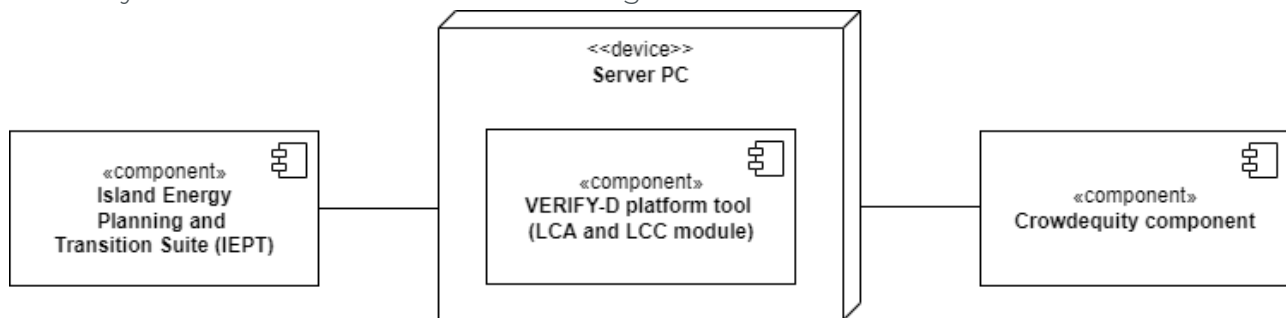


Figure 7-1: VERIFY – LCA/LCC toolkit's deployment view.

7.1.2 CrowdEquity Component

The CrowdEquity component will be developed under a web application framework scheme. For the development of the front-end of the platform, HTML and CSS programming languages will be utilized, while the framework that will be utilized for the development of the front-end of the platform is Angular. The reference language of the back-end of IANOS CrowdEquity platform, in order to facilitate its functionalities, will be Python, and the frameworks will be Django, Flask and PostgreSQL for the database.

The platform will be installed on a server machine, providing continuous access from and to the related IEPT components, as well as the DLT-based Transactive Platform component. The platform requires a stable internet connection to maintain service at any time.

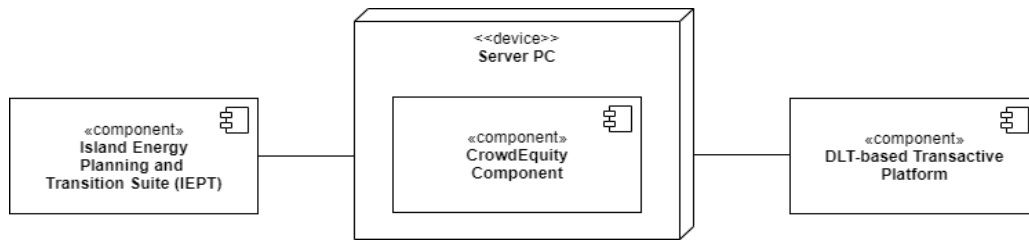


Figure 7-2: CrowdEquity component deployment diagram.

7.1.3 System Modeller

ESSIM is based on a microservices architecture, in which all components are provided by container images. Therefore, ESSIM needs a cloud infrastructure running Docker or Kubernetes for deployment.

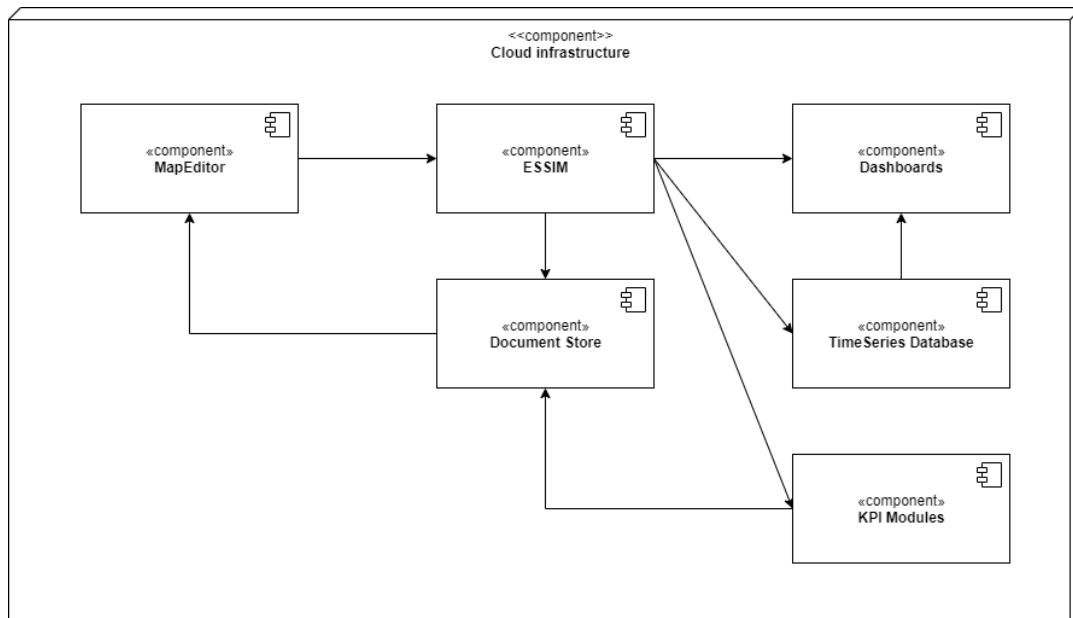


Figure 7-3: ESSIM Deployment diagram.

7.1.4 Grid-Oriented Optimizer - INTEMA.grid

The INTEMA.grid module will be developed as a web application. The back-end server will be based on Ruby, and custom Python programs will be utilized as an interface with the Modelica code generation, simulation, data visualization and post-processing (validation, KPIs calculation etc.) activities, to offer a GUI. The full-stack will be installed on a server machine connected to the internet, providing continuous access from and to the IEPT.

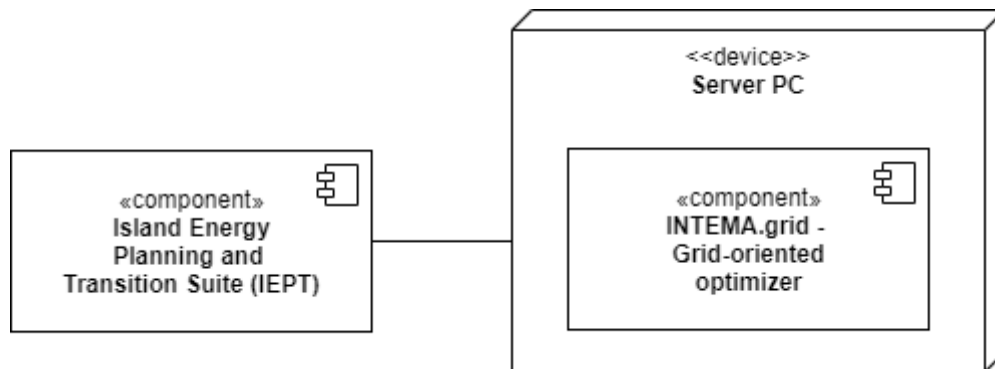


Figure 7-4: INTEMA.grid – grid-oriented optimizer's deployment view.

7.1.5 CBA component

The CBA component is going to be developed as a web application. Python will be used as a back-end in order to conduct all the calculations needed, based on the defined CBA methodology. The full stack will be installed on a server machine connected to the internet, providing continuous access from and to the IEPT.

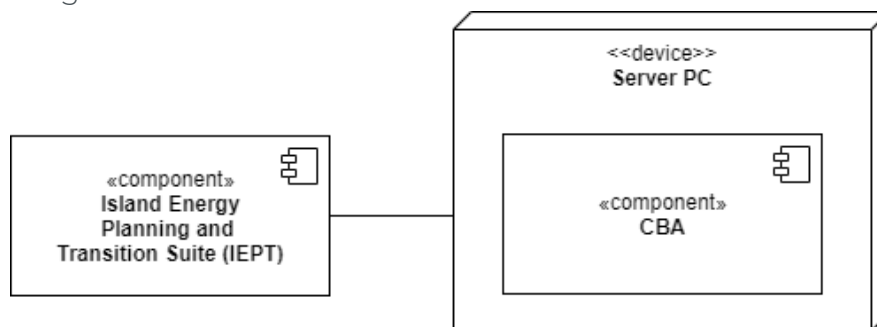


Figure 7-5: CBA component deployment diagram.

7.2 Decision Making Layer

7.2.1 Aggregation and Intelligent Segmentation

The Aggregation and Classification component will be able to execute commands in Python programming language, aiming to provide clusters of the project's energy assets. The component will be installed on the iVPP server machine, providing continuous access from and to the related iVPP components, as well as the IANOS Secure Enterprise Service BUS (ESB).

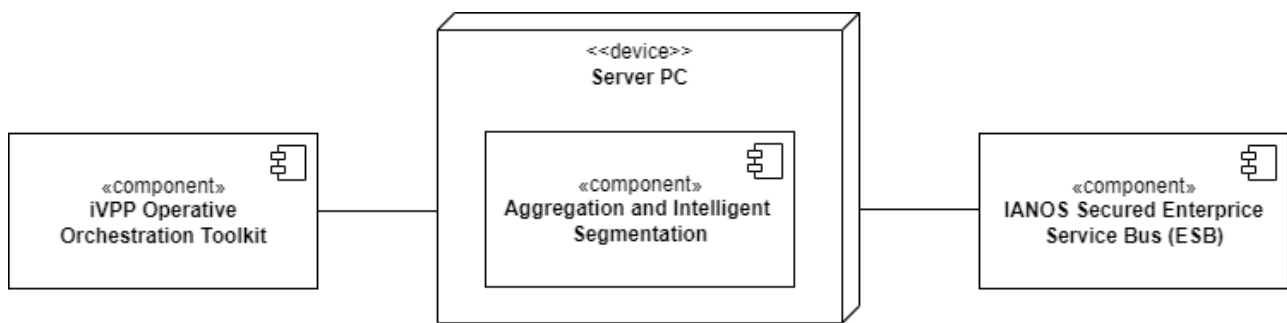


Figure 7-6: Aggregation and Classification deployment diagram.

7.2.2 Forecasting Engine

The Forecasting Engine component will be able to execute commands in Python programming language, in order to provide consumption, production and price forecasts. The component will be installed on the iVPP server machine, providing continuous access from and to the related iVPP components, as well as the IANOS Secure Enterprise Service BUS (ESB) and external systems cloud services, namely Weather Service and Energy Market.

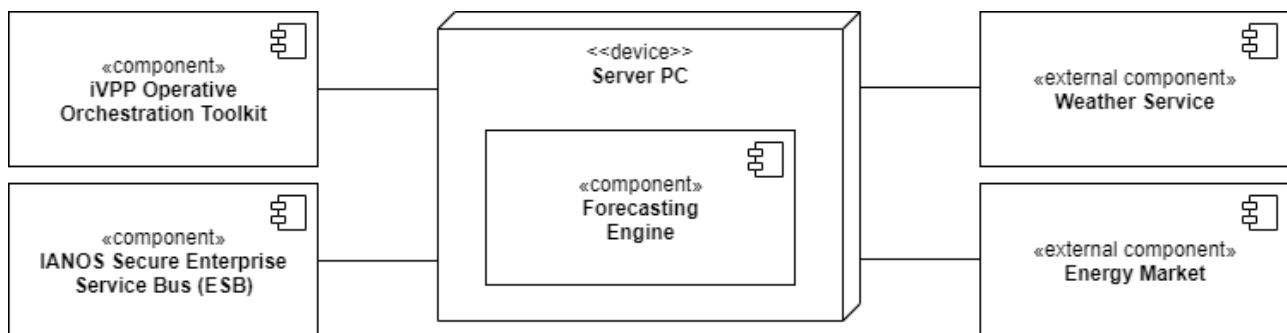


Figure 7-7: Forecasting Engine deployment diagram.

7.2.3 Centralized Dispatcher

7.2.3.1 Terceira View

Kiplo Core Platform

Kiplo Core Platform will be installed on a pair of Cloud Servers (an application server and a database server), providing continuous access from and to the related iVPP external components, as well as to the IANOS Secure Enterprise Service BUS (ESB), from where the main sources of information will flow.

The platform does not require any third-party licenses to operate besides the operating system (Microsoft Windows Server) and the database (Microsoft SQL Server).

As mentioned, the platform is a set of modules implemented as a collection of services that communicate via an internal bus. A comprehensive REST API exposes all the platform functionalities.

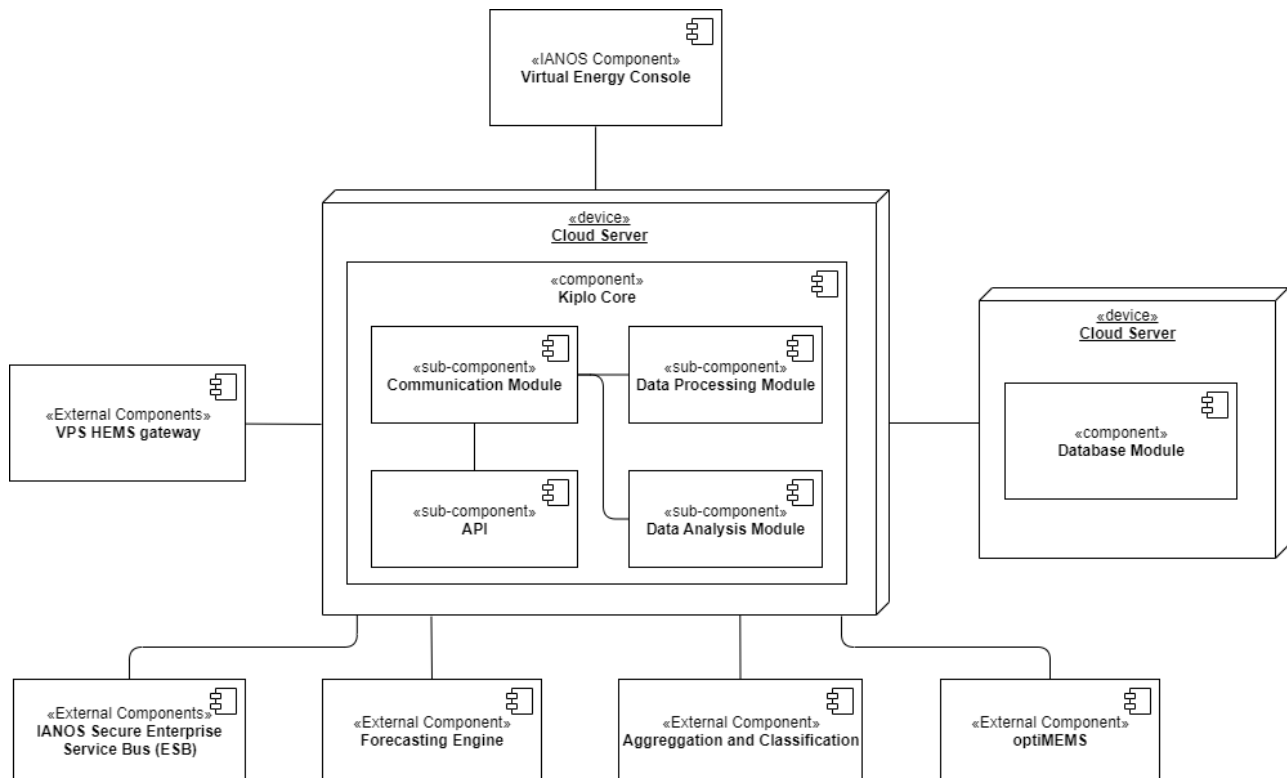


Figure 7-8: Kiplo Core Platform deployment diagram.

OptiMEMS

OptiMEMS component will be able to execute commands in Python programming language, to provide the optimal dispatch setpoints for the IANOS asset's functionality. The component will be installed on the iVPP server machine, providing continuous access from and to the related iVPP components, such as KIPLO Core Platform and Forecasting engine, as well as the IANOS Secure Enterprise Service BUS (ESB) and external systems cloud services, namely the Energy Market.

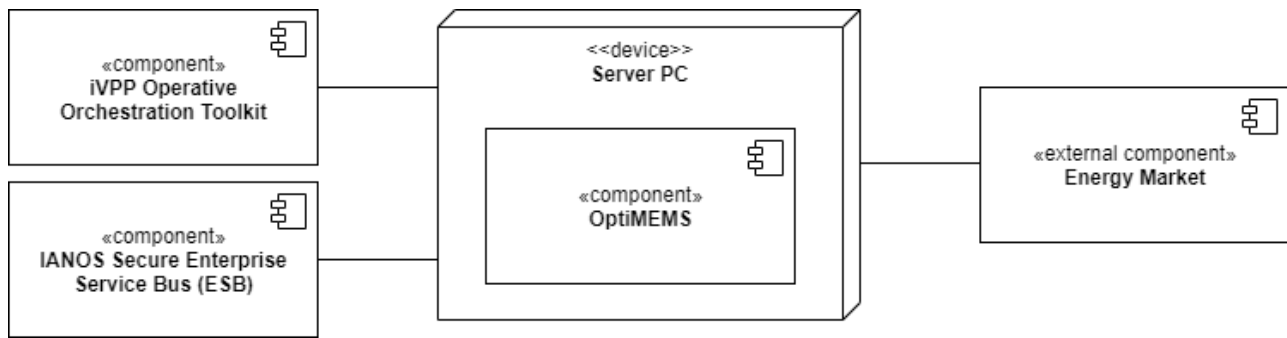


Figure 7-9: OptiMEMS deployment view.

7.2.3.2 Ameland View

Reflex

The figure below depicts the ReFlex deployment view. On the left side, input is provided by the Forecasting Engine and dEF-Pi as communication platform to provide the required flexibility information. On the right side connections are made to the energy markets and the IANOS enterprise service bus (ESB). ReFlex itself is run on server hardware.

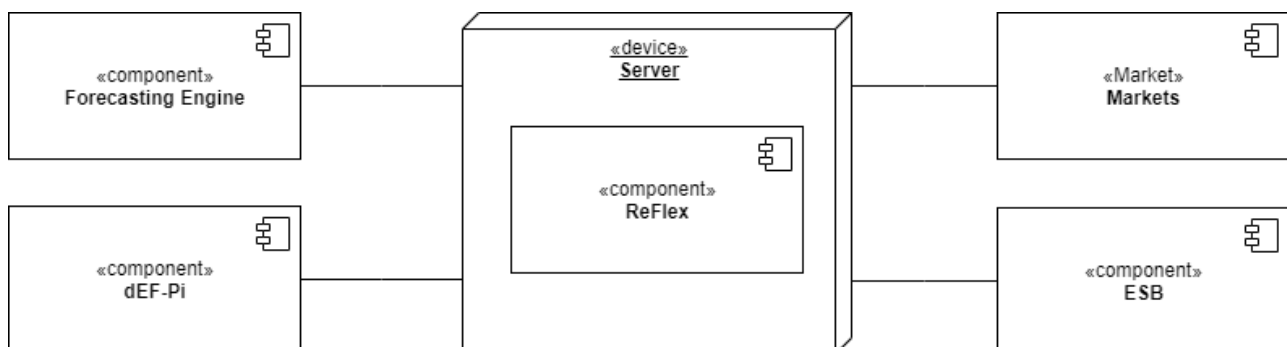


Figure 7-10: ReFlex deployment diagram.

7.2.4 DLT-based Transactive Platform

The components of the DLT-based transactive platform will be deployed using Docker. Figure 7.11 shows the deployment diagram of the system. Four containers are created and running in a host of Engineering's datacenters: the system back-end (yellow), the Ethereum blockchain (green), the dashboard (red) and the database (purple). Only the back-end can communicate with the other components, using the specific APIs. The containers are associated with the related images. A Docker image is a read-only template with instructions for creating a Docker container. Containers and images are managed by the Docker daemon. All the images are stored in a private Docker Registry

running in a separate host machine. The images are pulled, build, and executed by the Docker daemon. The system is accessible externally via browser or via REST APIs.

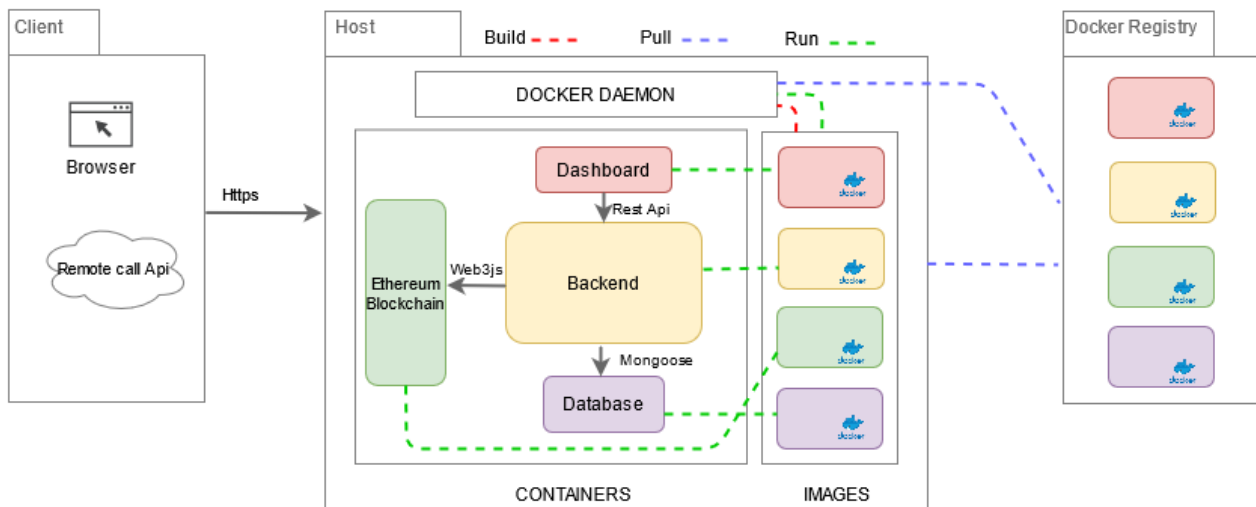


Figure 7-11: DLT-based transactive platform deployment diagram

7.3 User Interface Layer

7.3.1 Virtual Energy Console

As already referred, the Virtual Energy Console will consist of a user-friendly monitoring console (UI - User Interface) for VPP operators to effectively assess energy flows, properly manage, visualize and dispatch their energy assets. The decisions will originate at the Centralized Dispatcher using the algorithms of its different components. On the Virtual Energy Console, the VPP operator can change the dispatch setpoints of certain energy assets. The new setpoints will be sent to the assets through the Enterprise Service Bus (using Kiplo Core Communications and/or API Modules).

The Virtual Energy Console is thus the user interface for VPP operators, deployed on a cloud-based server, accessible via any web browser, not requiring any additional hardware, and which will not take any automated decision on the operations of the assets.

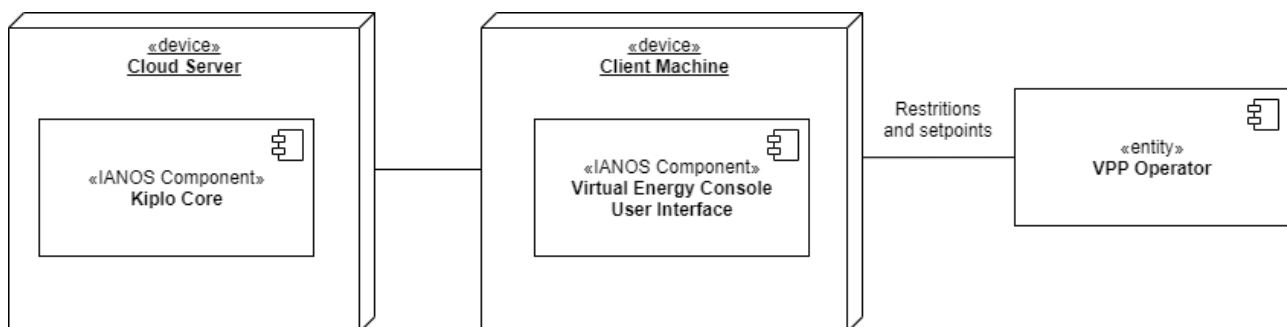


Figure 7-12: Virtual Energy Console deployment diagram.

7.4 Communication Layer

7.4.1 IANOS Secured Enterprise Service Bus

ESB will have the form of a set of microservices and components that are configured to interact among each other's and with the external components to provide the required functionalities.

For the deployment of the microservices, the Docker container technology has been used. Docker is an open-source virtualization technology known a containerization platform for software containers. These containers provide a means of enclosing an application, including its own filesystem, into a single, replicable package. Docker containers wrap up software and its dependencies into a standardized unit for software development that includes everything it needs to run: code, runtime, system tools and libraries. This guarantees that your application will always run the same and makes collaboration as simple as sharing a container image.

Having the microservices containerized has several advantages:

- The microservices always run in an appropriate isolated environment with no conflicts of program versions or dependencies.
- The deployment is much easier, because installing and running a microservice means basically deploying a pre-configured image.
- Version control is also very easy, because the upgrades of the microservices are just re-deployment of the images with a new version, but the specific configuration and historical values could be stored in external volumes, persistent and independent of the software version installed.
- Different versions of the microservices can co-exists, and it is easy to start or stop specific versions, no matter if they have different dependencies.
- Microservices are truly isolated, this has some benefits:
 - The containers can be configured with specific resource limitations (memory, disk, CPU) so that they cannot drain all the server requirements and affect other processes or the whole server.
 - The eventual failure of the microservices in the containers does not affect the rest of the microservices.
- Microservices containers can be clustered, so that they can use in a server farm, with automatic replication and fault tolerance, allowing.
- The Docker containers are much smaller in size, and resources needed than the virtual machine alternative.

This is the deployment diagram of the ESB platform:

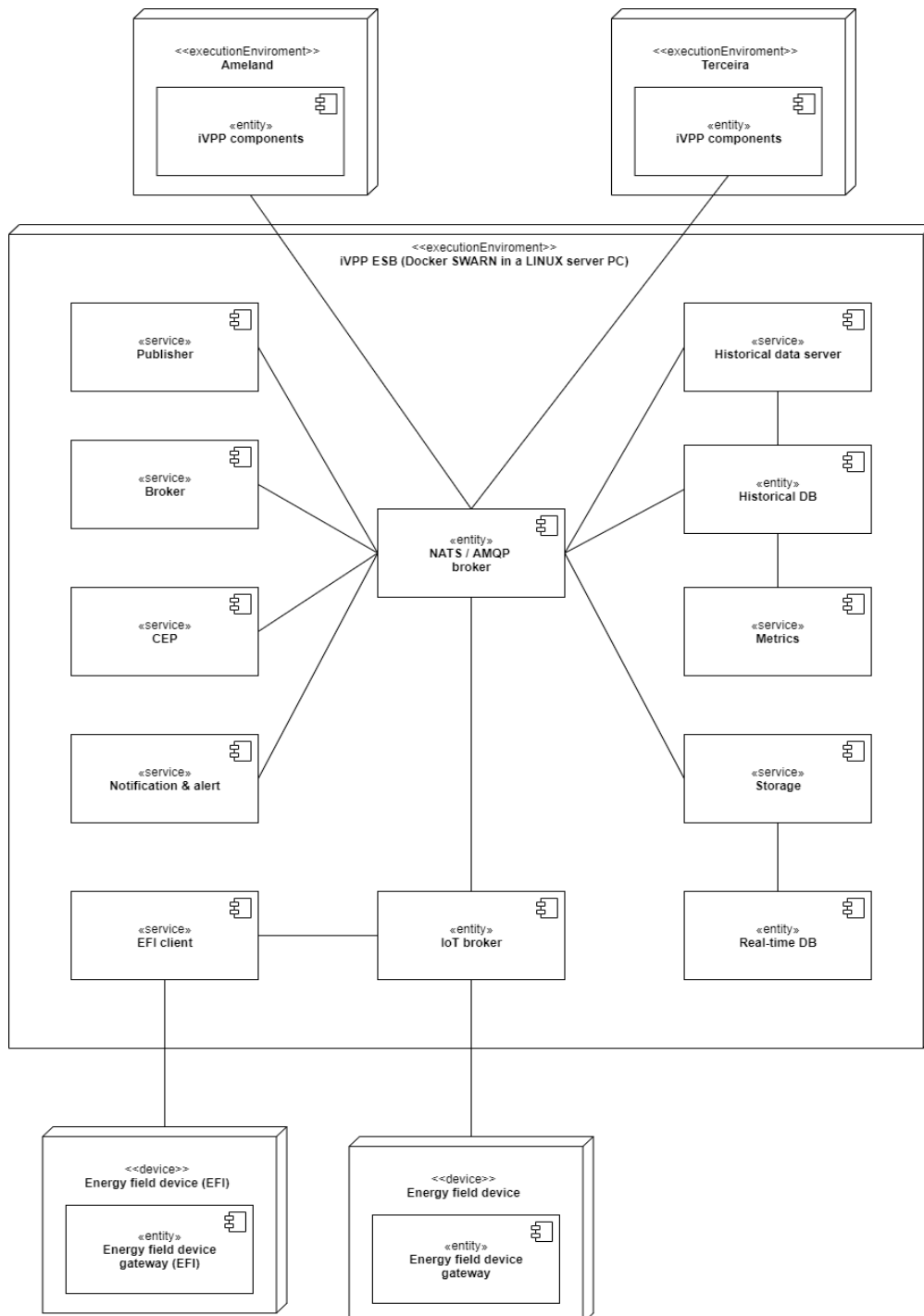


Figure 7-13: iVPP ESB deployment diagram.

All the services will be programmed in Javascript/NodeJS. The real time database will be MongoDB and the historical database will be ElasticDB. Regarding the brokers, RabbitMQ will be used for the IOT communication and NATS for inter-iVPP communications.

7.4.2 dEF-Pi

The hardware that is needed to implement dEF-Pi, depends on the assets. It is possible that some of the assets are already being monitored for maintenance purposes. If this is the case, an API can be used to enclose that specific asset on the platform. If this is not the case, Raspberry Pi's are needed to deploy a node on this asset. This raspberry pi functions as a gateway to the platform. In addition to this hardware, some hardware is needed for storage. This could be in a cloud, which would mean that even for the storage no hardware is needed in the project but is also possible to choose for a physical server (in this case for example on the island itself).

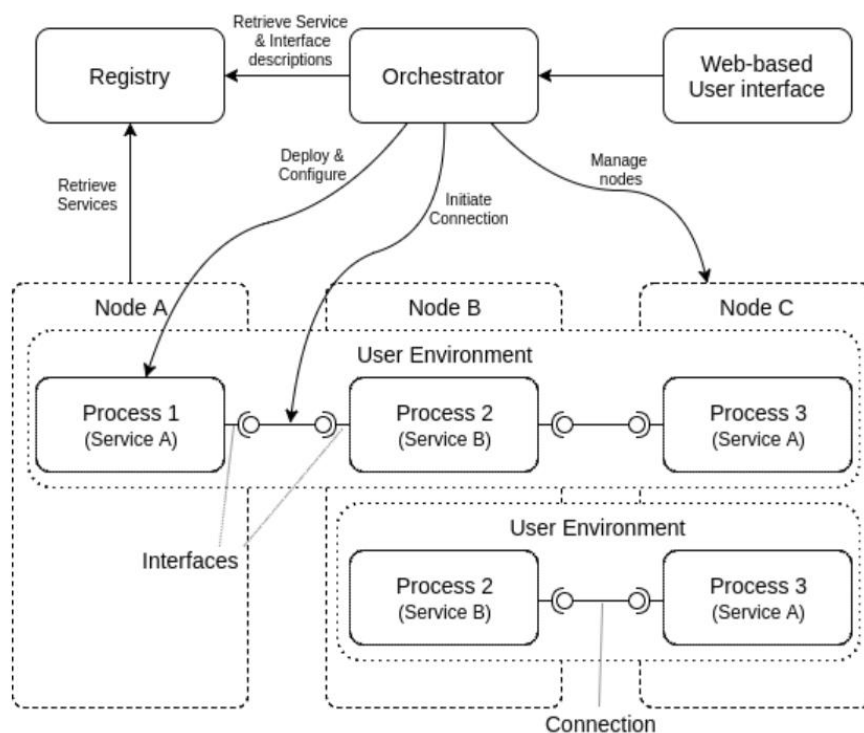


Figure 7-14: dEF-Pi deployment view.

7.4.3 Back-end Agent

The Back-end Agent component will be installed on a server PC, where it will be able to execute commands in Python programming language, aiming to provide communication with between the iVPP components and the IANOS Secure Enterprise Service Bus (ESB). Continuous access will be provided from and to the related iVPP components, as well as the IANOS Secure Enterprise Service BUS (ESB).

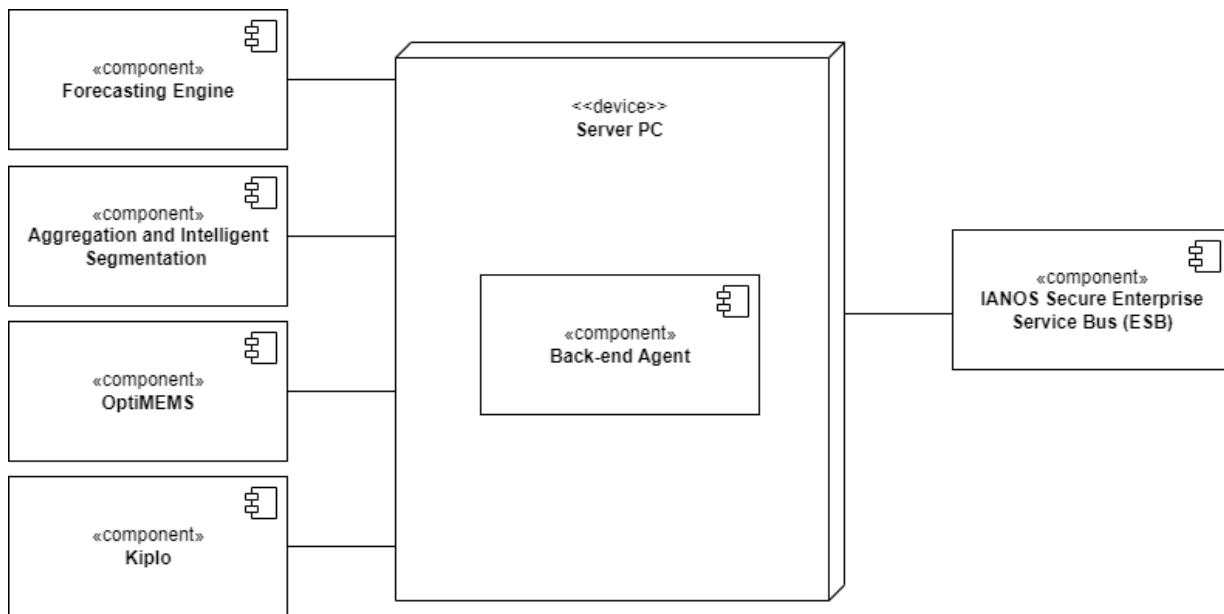


Figure 7-15: Back-End Agent Deployment View

8 Conclusions

This document constitutes the third Deliverable of Task 2.5: «System Architecture». This deliverable describes the overall architecture of the components that are going to be deployed in the context of IANOS project and mainly its ICT and software components. In order to better understand and describe the System Architecture, the 4+1 architectural view was adopted which describes each component from a Logical, Development, Process and Physical view as well as the different scenarios in which the components are used.

In this Deliverable D2.15, the main changes that were made regard the addition of the updated Data Model of IANOS, the update and alignment of the Use Case descriptions to D2.3 and finally the inclusion of component “Back-end Agent”.

As the deployment of certain equipment in Ameland is not yet finalized, another version (D2.16) might be needed to describe the developments in this pilot island.



Annex I: IANOS Common Data Model

Each island will have different topics, and depending on it, the message will have to follow or not some specific data model so that the database can guarantee the data is in the right format. This section contains the final version of the data model for the IANOS project in which 3 different data models that differ from some specific fields, but are almost identical overall are presented.. . The final data model will be presented in the second version of the deliverable “iVPP secure data monitoring and governance”, and in particular, in deliverable D4.2, which is linked to task T4.1. The monitoring processes for energy assets communications expect the messages to have the data model provided here. Nevertheless, the current data model for the ESB communication will be structured in the following sections:

- Measurements
- Status
- Cluster

Each of them linked to their corresponding topic (which has on of these names in the mqtt topic path). All the fields that will be specified are mandatory, extra fields are not allowed and some fields must be a fixed value type.

Measurement data model

- asset id: with /{island}/{locationId}/{assetId} (string, assetId being the same as id)
- id: This field will be automatically filled with the corresponding routing key of the mqtt topic (string, not required)
- measurements message: Array with all the different measurements, only 1 value per measurement allowed at a time (array of objects)
 - measurement name: Type of measurement (string)
 - measurement unit: Unit of the measurement (string)
 - measurements: Array with ONE measurement value (array of 1 object)
 - value: The corresponding value of the measurement (float)
 - timestamp: Date of the measurement (ISO format string)

An example of a valid message is indicated as follows:

```
1 {
2   "asset_id": "/ameland/house03/device_002",
3   "_id": "device_002",
4   "measurements_message": [{
5     "measurement_name": "activePower",
6     "measurement_unit": "kW",
7     "measurements": [{
8       "value": 87.643,
9       "timestamp": "2019-01-01T12:00:00.000"
10    }]
11  }, {
12    "measurement_name": "reactivePower",
13    "measurement_unit": "kW",
14    "measurements": [{
15      "value": 22.413,
16      "timestamp": "2019-01-01T12:00:00.000"
17    }]
18  }]
19 }
```

Status data model

- asset id: with /{island}/{locationId}/{assetId} (string, assetId being the same as id)
- id: This field will be automatically filled with the corresponding routing key of the mqtt topic (string, not required)
- status message: Array with all the different status, only 1 value per measurement allowed at a time (array of objects)
 - measurement name: Type of measurement (string, should be "measurement")
 - status: Array with ONE status value (array of 1 object)
 - value: The corresponding value of the status (string)
 - timestamp: Date of the measurement (ISO format string)

An example of a valid message is indicated hereafter:

```
1 {
2   "asset_id": "/ameland/house03/device_002",
3   "_id": "device_002",
4   "status_message": [
5     {
6       "measurement_name": "status",
7       "status": [
8         {
9           "value": "ok",
10          "timestamp": "2019-01-01T12:00:00.000"
11        }
12      ]
13    }
14  ]
15 }
```


Cluster Data Model

- asset id: with `/[island]/[locationId]/[assetId]` (string, `assetId` being the same as `id`)
- id: This field will be automatically filled with the corresponding routing key of the mqtt topic (string, not required)
- cluster message: Array with all the different measurements, only 1 value per measurement allowed at a time (array of objects)
 - measurement name: Type of measurement (string, should be cluster)
 - clusters: Array with ONE element (array of 1 object)
 - o category: Category of the cluster (string)
 - o timestamp: Date of the measurement (ISO format string)

An example of a valid message is indicated as follows:

```
1  {
2    "asset_id": "/ameland/house03/device_002",
3    "_id": "device_002",
4    "cluster_message": [
5      {
6        "measurement_name": "cluster",
7        "clusters": [
8          {
9            "category": "3",
10           "timestamp": "2019-01-01T12:00:00.000"
11          }
12        ]
13      }
14    ]
15  }
```

Annex II: Secured ESB communication data model

Rationale

The data related to energy assets can be fed into the ESB message broker (RabbitMQ) in MQTT or AMQP protocols, in a IoT fashion (bulk or very frequent data ingest that is processed, scaled, and balanced in a way that prevents the system from being overflowed)

The information exchange among ESB clients or components in the business and decision layers can be done using the previously mentioned protocols or (preferably) using NATS, that is better suited for the request/response communication pattern.

Data format for field devices messages

The processes that monitor the energy assets messages are expecting the messages to have the specific data model described here.

Examples of systems or energy assets that may use this mechanism for sending messages could be: Smart meters, RES dispatching centres, EMS SCADAs, PV or simulation algorithms that produce fake data periodically.

The messages must be valid JSON documents that adhere to the JSON schema described:

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://etraid-id.com/dataMessage.json",
  "definitions": {
    "complexValue": {
      "$id": "#complexValue",
      "type": "object",
      "properties": {
        "value": {
          "type": "string",
          "$id": "value",
          "pattern": "^(.*)$"
        },
        "timestamp": {
          "type": "string",
          "$id": "timestamp",
          "pattern": "^(.*)$"
        },
        "quality": {
          "type": "integer",
```

```

        "default": 1
      }
    },
    "required": [
      "value",
      "timestamp"
    ]
  }
},
"type": "object",
"title": "The Root Schema",
"required": [
  "id"
],
"properties": {
  "id": {
    "type": "string",
    "pattern": "^(.*)$"
  },
  "states": {
    "$ref": "#/definitions/complexValue",
    "$id": "#states"
  },
  "statesTimestamp": {
    "type": "string",
    "pattern": "^(.*)$"
  },
  "measurements": {
    "$ref": "#/definitions/complexValue",
    "$id": "#measurements"
  },
  "measurementsTimestamp": {
    "type": "string",
    "pattern": "^(.*)$"
  }
}
}
}

```

An example of a valid JSON message could be:

```

{
  "id": "device_id",
  "states": {
    "communicationState": {
      "value": "OK",
      "timestamp": "2019-11-08T12:45:40.035Z",
      "quality": 1.0
    }
  },
  "statesTimestamp": "2019-11-08T12:45:40.035Z",
  "measurements": {
    "activePower": {
      "value": 50.53363,
      "timestamp": "2019-11-08T12:45:40.035Z",
      "quality": 1.0
    }
  }
}

```

```
{
  "reactivePower": {
    "value": 12.363,
    "timestamp": "2019-11-08T12:45:40.035Z",
    "quality": 1.0
  },
  "frequency": {
    "value": 50.002,
    "timestamp": "2019-11-08T12:45:40.035Z",
    "quality": 1.0
  }
},
"measurementsTimestamp": "2019-11-08T12:45:40.035Z"
}
```

Basically, the JSON message has:

- An «id» field with the identifier of the device that the measures and states are related.
- A «states» object holding the states of the device. More than one state can be identified simultaneously, e.g. the communication state, the working mode, «door open» flag, etc. The values are normally predefined values (tags) or flags.
- A «measurements» object holding the representation of different measured magnitudes associated with the device, e.g. active power, voltage, etc.
- Two fields with the timestamp of the last state or measurement reported. This is optional and can be omitted

The list of measurements and states names is defined in the next section.

For every state or measurement, the value is actually represented by an object with three fields: Actual value, timestamp and quality of the measurement:

- The «quality» field is a float number between 0 and 1, being 1 the highest quality value. Normally, lowest values are associated to values calculated captured far away in time and probably not accurate. This field is optional and if not provided, 1 is assumed.
- The «value» field is a float number with the actual value.
- The «timestamp» field is a string representation of the measurement capture timestamp, in the [ISO 8601 extended format](#).

The measurements received are treated as accumulative. A received measurement will be valid in our system even though other messages could have been received for this device with no update for this specific measurement. The system could reduce the quality of the measurement in case it is not updated for a long time, but the new measurements messages do not replace the old measurements unless they overwrite them.

Data model measurements and states names

Data model

CERTH	Measurement name	Unit
active_power	activePower	kW
temp	temp	°C
load_forecast	loadForecast	MW
cloud_cover	cloudCoverage	% (0..100)
wind_spd	windSpeed	m/s
pv_forecast	pvForecast	kWh
wind_dr	windDirection	degress from north
wind_forecast	windForecast	kWh
da_price	daPrice	EUR/MWh
da_load	daLoad	MWh
da_gen	daGen	MWh
da_price_for	daPriceForecast	EUR/MWh
intra_price	intraPrice	EUR/MWh
intra_price_for	intraPriceForecast	EUR/MWh
imb_price	imbPrice	EUR/MWh
imb_price_for	imbPriceForecast	EUR/MWh
fcr	fcr	EUR/MW/ISP
fcr_for	fcrForecast	EUR/MW/ISP
label	Not measurement	-
soc_res	soc	% (0..100)
opt_bess	¿?	-
active_energy+	activeEnergyImported	kWh
active_energy-	activeEnergyExported	kWh
reactive_power	reactivePower	kVAr
power	activePower	kW

UNINOVA	Measurement name	Unit
active_power	activePower	kW
pv_forecast	pvForecast	kW
load_forecast	loadForecast	kW

CWD HEM	Measurement name	Unit
active_energy	activeEnergyImported	kWh



reactive_power	reactivePower	kvAr
----------------	---------------	------

ENG P2P	Measurement name	Unit
forecastEnergy	forescastEnergy	kWh
energy	activeEnergyImported/activeEnergyGenerated	kWh

CWD HB	Measurement name	Unit
avg_temp	temp	°C
soc	soc	% (0..100)

CWD ElectBattery	Measurement name	Unit
Grid Voltage	voltage	V
Grid Frequency	frequency	Hz
AC Out Current	current	A
AC Out Power	activePower¿?	kW
Battery Voltage	dc_voltage	V
Battery Current	dc_current	A
Battery State of charge	soc	% (0..100)

TNO	Measurement name	Unit
active_power	activePower	kW
state	Not measurement	

Efacec SRV	Measurement name	Unit
RTD1_CALC	probe1_temp	°C
RTD2_CALC	probe2_temp	°C
RTD4_CALC	oil_temp	°C
PRESSURE1	pressure1	mBAR
PRESSURE2	pressure2	mBAR
PRESSURE3	pressure3	mBAR
VIBRA_O	vibration	¿?
RUIDO	noise	dB
COM	Not measurement	-
LEVEL_BOUND	oilLevel	mm
OIL_PPM	oilMoisture	ppm



CORRENTE1	currentL1	A
CORRENTE2	currentL2	A
CORRENTE3	currentL3	A

EFACEC EV_Chargers	Measurement name	Unit
energy_active_export_interval	activeEnergyExported	kWh
energy_active_import_interval	activeEnergyImported	kWh
power_active_export > active_power_export	activePower-	kW
power_active_import	activePower+	kW
frequency	frequency	hz
maximum_power_offered		
dc_output_voltage		
dc_output_current		
soc	soc	% (0..100)
temperature	Is it necessary?	°C

BEON	Measurement name	Unit
powerLevel	Not measurement	
energy	dischargeAvailable	¿?

SEAURRENT	Measurement name	Unit
power output		

TERALOOP	Measurement name	Unit
soc	soc	% (0..100)
energy_usable	dischargeAvailable	kWh
energy_total	chargeAvailable	kWh
grid_power_instantaneous	activePower	kW
grid_voltage	voltage	V

Measurements names

States names

Measurement name	Symbol(s)	Units	Comments
activeEnergyImported	AEimp	(kWh)	Acummulated active energy imported
activeEnergyImportedL1	AEimp1, AEimp[0]	(kWh)	Acummulated active energy imported, Phase 1
activeEnergyImportedL2	AEimp2, AEimp[1]	(kWh)	Acummulated active energy imported, Phase 2
activeEnergyImportedL3	AEimp3, AEimp[2]	(kWh)	Acummulated active energy imported, Phase 3
activeEnergyImportedDelta		(kWh)	Delta active energy imported
activeEnergyExported	AEexp	(kWh)	Acummulated active energy exported
activeEnergyExportedL1	AEexp1, AEexp[0]	(kWh)	Acummulated active energy exported, Phase 1
activeEnergyExportedL2	AEexp2, AEexp[1]	(kWh)	Acummulated active energy exported, Phase 2
activeEnergyExportedL3	AEexp3, AEexp[2]	(kWh)	Acummulated active energy exported, Phase 3
activeEnergyExportedDelta		(kWh)	Delta active energy exported
activeEnergyGenerated	AEgen	(kWh)	Acummulated active energy generated
activeEnergyGeneratedL1	AEgen1, AEgen[0]	(kWh)	Acummulated active energy generated, Phase 1
activeEnergyGeneratedL2	AEgen2, AEgen[1]	(kWh)	Acummulated active energy generated, Phase 2
activeEnergyGeneratedL3	AEgen3, AEgen[2]	(kWh)	Acummulated active energy generated, Phase 3
activeEnergyRemanent	AEremanent	(kWh)	Remanent active energy (e.g. in a battery)
reactiveEnergyQi	REQi, REq[0]	(kvarh)	Reactive energy 1st Quadrant
reactiveEnergyQii	REQii, REq[1]	(kvarh)	Reactive energy 2nd Quadrant
reactiveEnergyQiii	REQiii, REq[2]	(kvarh)	Reactive energy 3rd Quadrant
reactiveEnergyQiv	REQiv, REq[3]	(kvarh)	Reactive energy 4th Quadrant
reactiveEnergyQiL1	REQi1, REqphase[0][0]	(kvarh)	Reactive energy 1st Quadrant, Phase 1
reactiveEnergyQiiL1	REQii1, REqphase[0][1]	(kvarh)	Reactive energy 2nd Quadrant, Phase 1
reactiveEnergyQiiiL1	REQiii1, REqphase[0][2]	(kvarh)	Reactive energy 3rd Quadrant, Phase 1
reactiveEnergyQivL1	REQiv1, REqphase[0][3]	(kvarh)	Reactive energy 4th Quadrant, Phase 1
reactiveEnergyQiL2	REQi2, REqphase[1][0]	(kvarh)	Reactive energy 1st Quadrant, Phase 2
reactiveEnergyQiiL2	REQii2, REqphase[1][1]	(kvarh)	Reactive energy 2nd Quadrant, Phase 2
reactiveEnergyQiiiL2	REQiii2, REqphase[1][2]	(kvarh)	Reactive energy 3rd Quadrant, Phase 2
reactiveEnergyQivL2	REQiv2, REqphase[1][3]	(kvarh)	Reactive energy 4th Quadrant, Phase 2

reactiveEnergyQiL3	REQi3, REqphase[2][0]	(kvarh)	Reactive energy 1st Quadrant, Phase 3
reactiveEnergyQiiL3	REQii3, REqphase[2][1]	(kvarh)	Reactive energy 2nd Quadrant, Phase 3
reactiveEnergyQiiiL3	REQiii3, REqphase[2][2]	(kvarh)	Reactive energy 3rd Quadrant, Phase 3
reactiveEnergyQivL3	REQiv3, REqphase[2][3]	(kvarh)	Reactive energy 4th Quadrant, Phase 3
reactiveEnergyCapacitive	REcap	(kvarh)	Reactive energy capacitive, Q2+Q4
reactiveEnergyInductive	REind	(kvarh)	Reactive energy inductive, Q1+Q3
reactiveEnergyPositive	REplus	(kvarh)	Reactive energy positive, Q1+Q2
reactiveEnergyNegative	REminus	(kvarh)	Reactive energy negative, Q3+Q4
activePower	P	(kW)	Active power
activePowerL1	P1, Pphase[0]	(kW)	Active power Phase 1
activePowerL2	P2, Pphase[1]	(kW)	Active power Phase 2
activePowerL3	P3, Pphase[2]	(kW)	Active power Phase 3
activePowerImported	Pimp	(kW)	Active power imported
activePowerExported	Pexp	(kW)	Active power exported
activePowerInput	Pin, Pinput	(kW)	Active power input
activePowerOutput	Pout, Poutput	(kW)	Active power output
activePowerRated	ratedP	(kW)	Rated active power
reactivePower	Q	(kvar)	Reactive power
reactivePowerL1	Q1, Qphase[0]	(kvar)	Reactive power Phase 1
reactivePowerL2	Q2, Qphase[1]	(kvar)	Reactive power Phase 2
reactivePowerL3	Q3, Qphase[2]	(kvar)	Reactive power Phase 3
reactivePowerCapacitive	Qcap	(kvar)	Reactive power capacitive
reactivePowerInductive	Qind	(kvar)	Reactive power inductive
reactivePowerQi	Qi, Qq[0]	(kvar)	Reactive power 1st Quadrant
reactivePowerQii	Qii, Qq[1]	(kvar)	Reactive power 2nd Quadrant
reactivePowerQiii	Qiii, Qq[2]	(kvar)	Reactive power 3rd Quadrant
reactivePowerQiv	Qiv, Qq[3]	(kvar)	Reactive power 4th Quadrant
reactivePowerQiL1	QiL, Qqphase[0][0]	(kvar)	Reactive power 1st Quadrant, Phase 1
reactivePowerQiiL1	QiiL, Qqphase[0][1]	(kvar)	Reactive power 2nd Quadrant, Phase 1

reactivePowerQiiiL1	Qiii1, Qqphase[0][2]	(kvar)	Reactive power 3rd Quadrant, Phase 1
reactivePowerQivL1	Qiv1, Qqphase[0][3]	(kvar)	Reactive power 4th Quadrant, Phase 1
reactivePowerQiL2	Qi2, Qqphase[1][0]	(kvar)	Reactive power 1st Quadrant, Phase 2
reactivePowerQiiL2	Qii2, Qqphase[1][1]	(kvar)	Reactive power 2nd Quadrant, Phase 2
reactivePowerQiiiL2	Qiii2, Qqphase[1][2]	(kvar)	Reactive power 3rd Quadrant, Phase 2
reactivePowerQivL2	Qiv2, Qqphase[1][3]	(kvar)	Reactive power 4th Quadrant, Phase 2
reactivePowerQiL3	Qi3, Qqphase[2][0]	(kvar)	Reactive power 1st Quadrant, Phase 3
reactivePowerQiiL3	Qii3, Qqphase[2][1]	(kvar)	Reactive power 2nd Quadrant, Phase 3
reactivePowerQiiiL3	Qiii3, Qqphase[2][2]	(kvar)	Reactive power 3rd Quadrant, Phase 3
reactivePowerQivL3	Qiv3, Qqphase[2][3]	(kvar)	Reactive power 4th Quadrant, Phase 3
apparentPower	S	(kVA)	Apparent power
apparentPowerL1	S1, Sphase[0]	(kVA)	Apparent power, Phase 1
apparentPowerL2	S2, Sphase[1]	(kVA)	Apparent power, Phase 2
apparentPowerL3	S3, Sphase[2]	(kVA)	Apparent power, Phase 3
apparentPowerRated	ratedS	(kVA)	Rated apparent power
current	I	(A)	In three-phase installations, average of the three phases
currentL1	I1, Iphase[0]	(A)	Current, Phase 1
currentL2	I2, Iphase[1]	(A)	Current, Phase 2
currentL3	I3, Iphase[2]	(A)	Current, Phase 3
currentNeutral	In	(A)	Neutral current
currentInput	Iinput	(A)	Input current
currentAngle	Iang	(deg)	Current angle
currentAngleL1	Iang1, Iangphase[0]	(deg)	Current angle, Phase 1
currentAngleL2	Iang2, Iangphase[1]	(deg)	Current angle, Phase 2
currentAngleL3	Iang3, Iangphase[2]	(deg)	Current angle, Phase 4
voltage	V	(V)	Voltage
voltageL1	V1, V[0]	(V)	Voltage, Phase 1
voltageL2	V2, V[1]	(V)	Voltage, Phase 2
voltageL3	V3, V[2]	(V)	Voltage, Phase 3
voltageAngle	Vang	(deg)	Voltage angle

voltageAngleL1	Vang1, Vangphase[0]	(deg)	Voltage angle, Phase 1
voltageAngleL2	Vang2, Vangphase[1]	(deg)	Voltage angle, Phase 2
voltageAngleL3	Vang3, Vangphase[2]	(deg)	Voltage angle, Phase 3
voltageL1L2	U12, U[0]	(V)	Voltage difference between Phase 1 and Phase 2
voltageL2L3	U23, U[1]	(V)	Voltage difference between Phase 2 and Phase 3
voltageL3L1	U31, U[2]	(V)	Voltage difference between Phase 3 and Phase 1
powerFactor	PF	0..1	Power factor
powerFactorL1	PF1, PFphase[0]	0..1	Power factor, Phase 1
powerFactorL2	PF2, PFphase[1]	0..1	Power factor, Phase 2
powerFactorL3	PF3, PFphase[2]	0..1	Power factor, Phase 3
frequency	f	(Hz)	Frequency
voltageTotalHarmonicDistortion	THDV	% (0..100)	Voltage Total Harmonic Distortion
voltageTotalHarmonicDistortionL1	THDV1, THDV[0]	% (0..100)	Voltage Total Harmonic Distortion, Phase 1
voltageTotalHarmonicDistortionL2	THDV2, THDV[1]	% (0..100)	Voltage Total Harmonic Distortion, Phase 2
voltageTotalHarmonicDistortionL3	THDV3, THDV[2]	% (0..100)	Voltage Total Harmonic Distortion, Phase 3
voltageHarmonic1	HV1, HV[0]	(V)	Voltage Harmonic 1
voltageHarmonic2	HV2, HV[1]	% (0..100)	Voltage Harmonic 2
voltageHarmonicN	HVN, HV[N-1]	% (0..100)	Voltage Harmonic N (2 >= N >= 42)
currentTotalHarmonicDistortion	THDI	% (0..100)	Current Total Harmonic Distorsion
currentTotalHarmonicDistortionL1	THDI1, THDI[0]	% (0..100)	Current Total Harmonic Distorsion, Phase 1
currentTotalHarmonicDistortionL2	THDI2, THDI[1]	% (0..100)	Current Total Harmonic Distorsion, Phase 2
currentTotalHarmonicDistortionL3	THDI3, THDI[2]	% (0..100)	Current Total Harmonic Distorsion, Phase 3
currentHarmonic1	HI1, HI[0]	(A)	Current Harmonic 1
currentHarmonic2	HI2, HI[1]	% (0..100)	Current Harmonic 2
currentHarmonicN	HIN, HI[N-1]	% (0..100)	Current Harmonic N (2 >= N >= 42)
Measurement name	Symbol(s)	Units	Comments
stateOfCharge	SoC	% (0..100)	State of charge
stateOfHealth	SoH	% (0..100)	State of health
capacity	capacity	(kWh)	Capacity
chargeAvailable	AEcharge	(kWh)	Charge available
dischargeAvailable	AEdischarge	(kWh)	Discharge available
maxBatteryCellTemperature		°C	Max battery cell temperature
minBatteryCellTemperature		°C	Min battery cell temperature
inverterTemperature		°C	Inverter temperature
numberOfCycles	numCycles	–	Number of cycles

State name	Possible values	Comments
communicationStatus	0: ok	Asset connectivity
	1: noCommunication	

Storage specific measurements names

Measurement name	Units	Comments
stateOfCharge	%	State of charge
stateOfHealth	%	State of health
chargeAvailable	(kW)	
dischargeAvailable	(kW)	
maxBatteryCellTemperature	°C	Max battery cell temperature
minBatteryCellTemperature	°C	Min battery cell temperature
inverterTemperature	°C	Inverter temperature

Storage specific status names

State name	Possible values	Comments
status	0: disconnected	Battery/storage status
	1: connected	
	2: charge	
	3: discharge	
	4: standby	
	5: error	
	6: busy	
	7: islanding	
workingMode	0: Standard	Battery/storage working mode
	1: Manual	
	2: Alarm	
	3: Backup	



	4: Test	
	5: Schedule	
	6: Config	
controlMode	0: System Manually set inverter target power	Battery/storage control model (closed loop)
	1: Load Manually set target power at meter	
	2: PvsF Frequency regulation	
	3: PvsV Voltage regulation	
	4: QvsV Voltage regulation	

Data communication for field device messages

The generated JSON messages created according to the previous definition must be sent to the ESB message broker in MQTT or AMQP protocols.

Both protocols provide an endpoint for the clients to “send” the data, and both protocols require the messages to be “tagged” or identified with a topic (or routing key in AMQP nomenclature).

Topics are structured in a hierarchy similar to folders and files in a file system using a delimiter (the forward slash ‘/’ in MQTT, and the dot ‘.’ In AMQP).

The hierarchy of the topics helps identifying the nature of the message and the format must be the following:

<SYSTEM_NAME>.<ASSET_TYPE>.<ASSET_UNIQUE_IDENTIFIER>

Examples of these could be:

AMELAND.METER.31245

TERCEIRA/FARM/VELEBIT

Properties of the topic names:

- Are Case sensitive
- Use UTF-8 strings.
- Must consist of at least one character to be valid.

The asset type and identifier must be agreed between the relevant partners and ETRA as ESB developer.

Platform security

Regarding the security, different mechanisms will be established to enforce security:

- All connections to the ESB broker will require valid credentials
- The access to specific topics will be restricted by user, so that only the clients with specific credential will be able to publish or subscribe to data or restricted topics.



The configuration of the permissions for each credential will be configured at the ESB.

- The ESB will allow clients to authenticate by using either user & password, long-time tokens or certificates
- All the communication channels will use the secure version using SSL (MQTTS, HTTPS, AMQPS)



Annex III: Secured ESB weather information communication data model

Rationale

The data related to current and forecasted weather information is taken from online weather information server weatherbit.io by the ESB message broker and offered to ESB client by different protocols (AMQP and NATS)

The different ESB clients can subscribe to weather information in different locations, and the ESB will periodically forward them the most up-to data or the requested weather forecasting.

Data model for weather information messages

The weather information messages will adhere to a specific data model described here. The messages will be valid JSON documents that adhere to the JSON schema described here:

Real time weather messages data model

```
{
  "location": {
    "name": "Valencia,ESP",
    "latitude": 39.469917,
    "longitude": -0.3763
  },
  "timestamp": "2019-11-20T13:50:00Z",
  "atmosphericPressure": 1004.1, // Pressure (mbar)
  "seaLevelPressure": 1008.3, // Sea level pressure (mbar)
  "windSpeed": 3.13, // Wind speed (m/s)
  "windDirection": 222, // Wind direction (degrees)
  "windDirectionCardinal": "SW", // Abbreviated wind direction
  "temperature": 18.3, // Temperature (Celcius)
  "apparentTemperature": 18.4, // Apparent temperature (Celcius)
  "relativeHumidity": 37, // Relative humidity (%)
}
```



```
"dewPoint": 3.4, // Dew point (Celcius)
"cloudCoverage": 33, // Cloud coverage (%)
"visibility": 5, // Visibility (km)
"precipitationAccumulation": 0, // Liquid equivalent precipitation rate (mm/hr)
"snowfallAccumulation": 0, // Snowfall (mm/hr)
"ultravioletIndex": 4.34596, // UV Index (0-11+)
"airQualityIndex": 0, // Air Quality Index [US - EPA standard 0 - +500]
"diffuseHorizontalSolarIrradiance": 85.14, // Diffuse horizontal solar irradiance (W/m^2) [Clear Sky]
"directNormalSolarIrradiance": 729.53, // Direct normal solar irradiance (W/m^2) [Clear Sky]
"globalHorizontalSolarIrradiance": 363.91, // Global horizontal solar irradiance (W/m^2) [Clear Sky]
"solarRadiation": 357.2, // Estimated Solar Radiation (W/m^2)
"elesolarElevationAngle": 23.05, // Solar elevation angle (degrees)
"solarHourAngle": 54 // Solar hour angle (degrees)
}
```

Forecasted weather messages data model

```
[{
  "location": {
    "name": "Valencia,ESP",
    "latitude": 39.469917,
    "longitude": -0.3763
  },
  "timestamp": "2019-11-20T15:00:00Z",
  "atmosphericPressure": 1007.61,
  "seaLevelPressure": 1008.73,
  "windSpeed": 2.74225,
  "windDirection": 247,
  "windDirectionCardinal": "WSW",
  "windGustSpeed": 6.59272, // Wind gust speed (m/s)
  "temperature": 16.4,
  "apparentTemperature": 16.4,
  "relativeHumidity": 34,
  "dewPoint": 0.4,
  "cloudCoverage": 33,
  "cloudCoverageHighLevel": 0, // High-level (>5km AGL) cloud coverage (%)
  "cloudCoverageMidLevel": 9, // Mid-level (~3-5km AGL) cloud coverage (%)
}
```




```
"cloudCoverageLowLevel": 25, // Low-level (~0-3km AGL) cloud coverage (%)
"visibility": 24.135,
"precipitationAccumulation": 0,
"precipitationProbability": 0, // Probability of Precipitation (%)
"snowfallAccumulation": 0,
"snowDepth": 0, // Snow Depth (mm)
"ultravioletIndex": 1.79043,
"diffuseHorizontalSolarIrradiance": 69.5,
"directNormalSolarIrradiance": 616.06,
"globalHorizontalSolarIrradiance": 226.87,
"solarRadiation": 222.945,
"ozone": 331.598 // Average Ozone (Dobson units)
},
... // One message per hour
]
```

Request API

The forecasted weather information can be queried by sending request messages to the NATS endpoint of the ESB. The request messages must be sent with the following routing keys:

IANOS.weather

Or

IANOS.weatherforecast

The body of the request message must contain the request parameters formatted as a JSON object with the following data model:

```
{
  "cityName": "Valencia", // (Mandatory) Name of the place, used as a kind of ID.
  "lat": "39.469917", // (Query option 1) Latitude.
  "lon": "-0.376300", // (Query option 1) Longitude.
  "city": "Valencia", // (Query option 2) Name of the city, e.g. "Valencia", "Valencia,ES". Can be
combined with the property "country".
  "postalCode": "46014", // (Query option 3) Postal code.
  "country": "ES", // (Query options 2 and 3) Country in ISO 3166-1 alpha-2 code format, e.g. "ES".
It is mandatory for query option 3 and optional for query option 2.
  "hours": 48, // (Optional, only for forecast) Number of hours in the future to retrieve. Default
(and maximum) value: 48
}
```



Subscribe to real time weather data

There will be the possibility to subscribe to weather data changes and receive the information proactively. This can be done by subscribing using AMQP to a specific 'Queue' and configuring a routing rule from the IANOS 'exchange' to this queue. A different routing rule based on the message topics could be defined to select What information to receive in the selected queue. Examples of such rules could be:

IANOS.VALENCIA.weather

Or

IANOS.46013.weatherforecast

Or

IANOS.myPoint.weather

The topic can be splitted by dots. The central part is the identifier of the piece of weather data (that could be configured using a web form). The final part of the topic will allow to indicate the willingness to receive real time or forecasted data.

The messages received after subscribing will have the same data model as the one described in the request API.