The PARITY project has received funding from the EU's Horizon 2020 research and innovation programme under grant agreement No 864319

Project Acronym:	**PARITY**
Project Full Title:	**Pro-sumer AwaRe, Transactive Markets for Valorization of Distributed flexibilITY enabled by Smart Energy Contracts**
Grant Agreement:	**846319**
Project Duration:	**42 months (01/10/2019 – 31/03/2023)**

DELIVERABLE D8.1
Report on activities for system integration

Work Package:	**WP8 – System Integration, Demonstration and Impact Assessment**
Task:	**T8.1 – PARITY System Integration**
Document Status:	**Final v1.0**
File Name:	**PARITY_D8.1_Report on activities for system integration_R1_V1.0_CERTH**
Due Date:	**January 2022**
Submission Date:	**January 2022**
Lead Beneficiary:	**CERTH**

Dissemination Level	
Public	X
Confidential, only for members of the Consortium (including the Commission Services)	

The PARITY project has received funding from the EU's Horizon 2020 research and innovation programme under grant agreement No 864319

Authors List

Leading Author			
First Name	Last Name	Beneficiary	Contact e-mail
Stylianos	Zikos	CERTH	szikos@iti.gr

Co-Author(s)				
#	First Name	Last Name	Beneficiary	Contact e-mail
1	Christos	Timplalexis	CERTH	ctimplalexis@iti.gr
2	Stavros	Koltsios	CERTH	skoltsios@iti.gr
3	Georgios	Skaltsis	CERTH	gskaltsis@iti.gr
4	Giorgos	Fiotakis	HYPERTECH	g.fiotakis@hypertech.gr
5	Andreas	Muñoz	CIRCE	amunoz@fcirce.es
6	Davide	Rivola	HIVE	davide.rivola@hivepower.tech
7	Davide	Strepparava	SUPSI	davide.strepparava@supsi.ch
8	Matteo	Salani	SUPSI	matteo.salani@idsia.ch
9	Marco	Derboni	SUPSI	marco.derboni@idsia.ch
10	Panagiotis	Andriopoulos	QUE	panos@que-tech.com
11	Hamza	Shafique	CWATT	hamza.shafique@checkwatt.se

Reviewers List

Reviewers			
First Name	Last Name	Beneficiary	Contact e-mail
Iulia	**Minda**	E.ON	iulia.minda@eon.se
Panagiotis	Moraitis	QUE	p.moraitis@que-tech.com

Version History

Version	Author	Date	Status
0.1	Stylianos Zikos, CERTH	July 23, 2021	Initial draft (TOC)
0.4	Stylianos Zikos, CERTH	December 21, 2021	Added content in sections 1 and 2
0.5	Stylianos Zikos, CERTH	January 18, 2022	Added content about integration status and activities
0.9	Stylianos Zikos, Christos Timplalexis, CERTH	January 26, 2022	Final draft for internal review
1.0	Stylianos Zikos, CERTH	February 4, 2022	Final version including comments from partners, ready for submission

Legal Disclaimer

Copyright

Executive Summary

This document is a report on the activities that have been performed for system integration within the PARITY project. These activities are important not only for fulfilling the requirements for communication and data exchange among the different components and market participants, but also for documenting the steps and the sequence of actions that need to be performed in order to deploy and setup the PARITY system in a new environment.

Initially, an integration plan has been developed that will guide the interconnection of the developed components. According to the integration plan, two main iterations are foreseen for system integration, and the main objectives in each iteration were defined according to the development requirements and status of the components. The present version of the document describes the progress achieved during the first iteration. The integration process includes both the definition of the required APIs and the formats of exchanged data. Moreover, the methodology for the system integration testing is presented. Integration tests, starting in an early phase at the pre-pilots, will ensure that the communication between different components will be as designed and that no serious problems will appear during the evaluation phase at the pilots.

Lastly, the remaining activities for system integration, the integration tests at small-scale pilot sites and at pilot sites prior to system evaluation, as well as all finalised interfaces will be described in an updated version of this document, which will be delivered in March 2023.

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

Term	Description
API	Application Programming Interface
AS	Ancillary Service
BA	Blockchain Agent
BaaB	Building-as-a-Battery
BESS	Battery Energy Storage System
BMS	Building Management System
BRP	Balance Responsible Party
DERs	Distributed Energy Resources
DHW	Domestic Hot Water
DR	Demand Response
DSO	Distribution System Operator
EV	Electric Vehicle
HTTP	Hypertext Transfer Protocol
HVAC	Heating, ventilation, and air conditioning
IML	Information Management Layer
IoT	Internet of Things
IP	Internet Protocol
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LEM	Local Energy Market
LFM	Local Flexibility Market
OCPP	Open Charge Point Protocol
PFM	Prosumer Flexibility Manager
PV	Photovoltaics
REST	Representational State Transfer
SBM	Stationary Battery Manager
SD	Secure Digital
SIT	System Integration Testing
SLA	Service Level Agreement
SoC	State of Charge
TLC	Traffic Light Concept
TSO	Transmission System Operator
UI	User Interface
URL	Uniform Resource Locator
V2G	Vehicle-to-Grid
VPP	Virtual Power Plant

1. INTRODUCTION

The PARITY project addresses the "structural inertia" of distribution grids by delivering a transactive flexibility framework that will increase durability and efficiency of the electrical grid, while simultaneously enabling the adoption of more Renewable Energy Sources (RES) through enhanced real time control of Distributed Energy Resources (DER) flexibility combined with novel Active Network Management functionalities.

PARITY delivers a local flexibility management platform through the seamless integration of Internet of Things (IoT) and Blockchain technologies. By delivering a smart-contract enabled market platform based on blockchain technology, PARITY will facilitate the efficient deployment of local micro-transactions and reward flexibility in a cost-reflective and symmetric manner, through price signals of higher spatio-temporal granularity based on real-time grid operational conditions.

Finally, by deploying advanced IoT technology PARITY will offer distributed intelligence (DER profiling) and self-learning/self-organization capabilities (automated real-time distributed control), orchestrated by cost reflective flexibility market signals generated by the blockchain-based local energy market platform. Within PARITY, DER will form dynamic clusters that essentially comprise self-organized networks of active DER nodes, engaging in real-time aggregated & P2P energy/flexibility transactions.

1.1 Scope and Objectives of the Deliverable

The PARITY system is composed of several components that have different main objectives, however many interactions are needed to accomplish data storage and exchange, as well as seamless operation and participation in the Local Energy Market (LEM) / Local Flexibility Market (LFM). The purpose of this report is to describe the activities that have taken place for the PARITY system integration and to provide useful insights about the development and integration testing. As a first step, an integration plan has been developed that will guide the interconnection of the developed components. The plethora of dependencies between the modules as well as the complexity of functional requirements, compel the existence of an analytical approach that will facilitate the development of a coherent and efficient communication map. Furthermore, the effort achieved it this task will also be the stepping stone for scaling the adoption of the methodology in the four pilots. The integration procedure includes both the definition of the required APIs and formats of exchanged data, as well as the testing phase to ensure the seamless operation in all foreseen use cases.

1.2 Structure of the Deliverable

The rest of the report is structured according to the following:

- Chapter 2 describes the integration needs of the main components of the PARITY system. Moreover, a detailed description of the integration plan is provided along with an integration overview within each system layer.
- Chapter 3 analyses the utilized components for the implementation of the methodology. More specifically, the presented description includes information related to functionality, interconnectivity and data flow, while the reader is informed about the integration and development status as well as any problems encountered during these phases.
- Chapter 4 describes the integration activities that have been performed at pre-pilot and pilot sites.
- Chapter 5 reports the integration testing activities by providing a description of the followed procedures and an explanatory presentation of the resulted outcomes.

1.3 Relation to other tasks and deliverables

The activities within T8.1 are related to other tasks and deliverables as follows. The integration procedure will take into account the technical specifications of the system along with the definition of the architecture as developed within T3.5. Furthermore, the utilized communication protocols will follow the privacy and security specifications that were presented in T3.4. All the integrated components derive from the WP5, WP6, and WP7. Lastly, as inputs to T8.1 are considered the T8.3 and T8.4 that are related with the procurement of the hardware infrastructure and their deployment in the pilots, as well as the pre-validation activities, respectively.

T8.1 delivers an integrated platform that facilitates the communication of the components which were developed in WP5, WP6, and WP7. The resulted system will be tested within T8.4 and its deployment and demonstration will take place within T8.5.

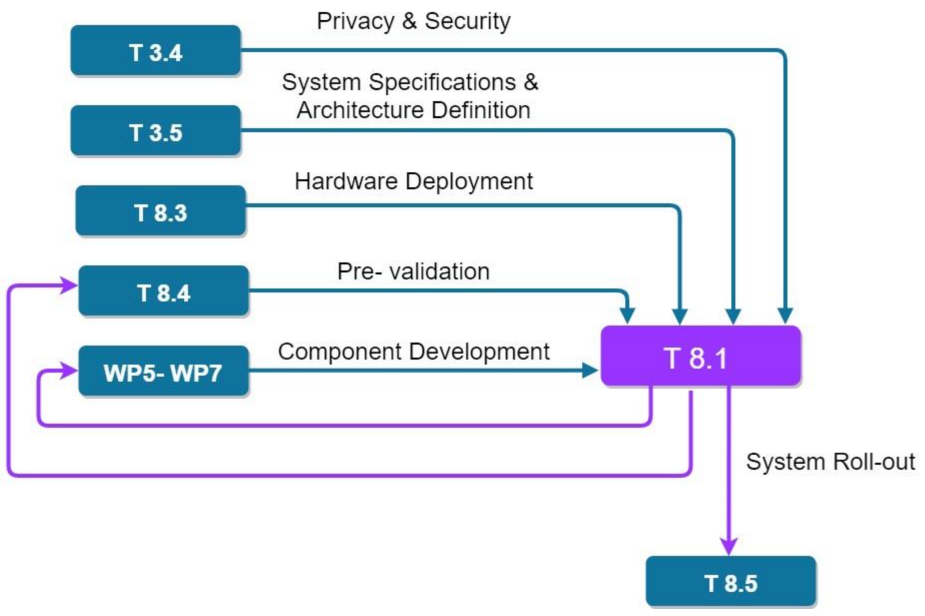

Figure 1. Relation of T8.1 to other PARITY tasks.

2. INTEGRATION NEEDS AND PLAN

In this chapter, the various integration requirements in the platform are described along with the integration plan that aims to fulfil them. The first part of this section explains the reasons for the high integration needs in the PARITY system and provides a list of all the involved components. Then, the integration plan is presented, which determines the implementation process that is being followed during the integration procedure. An overview of integration-related topics is provided for components that lie within one of the four different layers.

2.1 Integration Needs

System integration, or integration at the system level, refers to the integration of components, elements or subsystems, or human interactions in order to realise a system that accomplishes the system objectives [1]. Thus, system integration focuses mainly on the integration of components, subsystem internal/external interfaces and human interactions.

According to the system architecture that has been presented in the deliverable D3.5 – PARITY System Architecture, it is clear that the integration needs for the PARITY system are high. This is due to various reasons that are listed below:

- Several components are developed for different actors within the local grid network: Distribution System Operator (DSO), Market Operator, aggregator, prosumers.
- LEM/LFM market operation relies on, and affects the data flow between components responsible for data collection, processing and storage, as well as the components for market participants (Aggregator Toolset, DSO Toolset, Prosumers' Oracle).
- All PARITY use cases are data-driven and their realisation is based on automated data collection and processing. Moreover, most of the use cases require interactions with the LEM/LFM platform, the DSO Toolset and the Aggregator Toolset. The operation of these three components requires the availability of measurements and data such as forecasts derived after an optimisation process.

The table below lists the main components that need to be integrated to the PARITY final system. It shall be noted that most of the software components are composed of sub-components.

Table 1. List of main PARITY components.

Components
Information Management Layer Cloud
P2H, PV & Battery Managers
EV Profiling and Smart Charging
Oracle
LEM/LFM and Blockchain platform
Aggregator Toolset
DSO Toolset
D-STATCOM device
Prosumer Applications (User Interfaces)

2.2 Integration Plan

According to the integration plan, the integration process will be performed in two main iterations (Figure 2). During the first iteration, the focus is given on the integration of the related components that are within the same level in the architecture, as well as on the integration of components with the LEM/LFM platform. The integration of finalised user interfaces as well as the final integration of all functionalities of DSO Toolset and Aggregator Toolset will be carried out in the second and final iteration.

The main objective is to define the data to be exchanged among components, along their format. The data can be related to the configuration or to the operation of the PARITY system. Moreover, integration testing is foreseen, which is an important process in which individual software modules are combined and tested as a group. It includes the definition of scenarios to be assessed and the analysis of the results towards the identification of any issues that need to be solved or require improvement. The placement of integration testing within the software testing procedure is illustrated in Figure 3. Integration tests are foreseen to be performed in development/controlled environment, at the small-scale pre-pilot deployments and at the final deployments in the pilots. More details about the integration testing are presented in section 5.

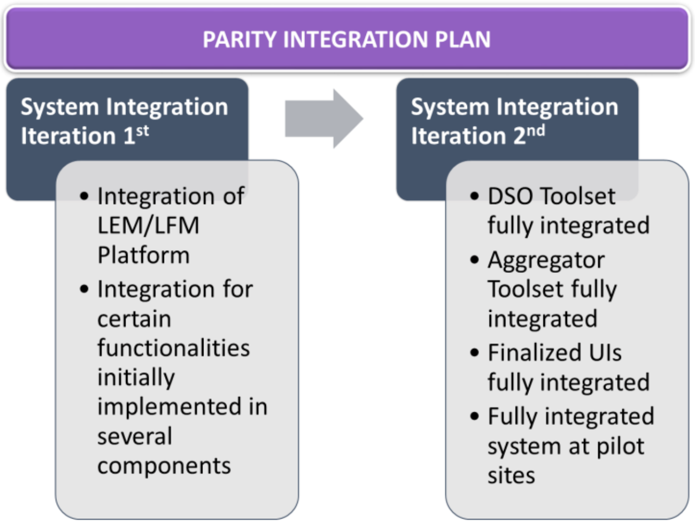
PARITY INTEGRATION PLAN
System Integration Iteration 1st
• Integration of LEM/LFM Platform
• Integration for certain functionalities initially implemented in several components
System Integration Iteration 2nd
• DSO Toolset fully integrated
• Aggregator Toolset fully integrated
• Finalized UIs fully integrated
• Fully integrated system at pilot sites

Figure 2. Outline of PARITY integration plan.

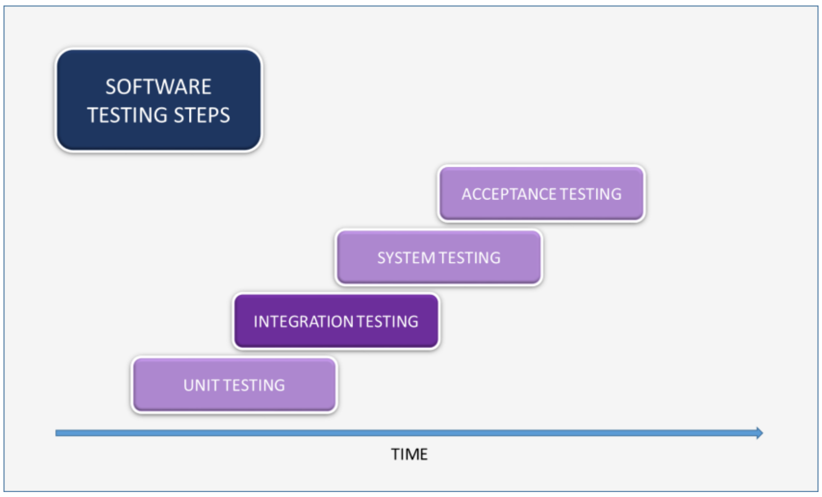

Figure 3. Steps of software testing.

2.2.1 Field-level components / Information management layer

Integration activities for the components in this category mainly refer to the integration of components that have data handling and storage as primary objective, such as the Information Management Layer cloud with the LEM/LFM Repository for storing the collected and pre-processed data from the IoT network. Moreover, the integration of the Electric Vehicle (EV) charging stations with the EV Profiling and Smart Charging component, and the integration of equipment such as the stationary batteries and Photovoltaics (PV) with the LEM/LFM Repository is done through the Stationary Battery Manager and the PV Manager. The integration of assets with the locally deployed IoT Gateways in buildings as well as the integration of the D-STATCOM device at the DSO side are also necessary.

2.2.2 Market-related components / Market layer

The integration between the components responsible for the Blockchain-enabled LEM/LFM market operation is necessary. This is focused on the communication among the Blockchain Agents, Oracle, LEM/LFM Repository and off-chain services. Additionally, the integration of Aggregator Toolset with interfaces of external markets (Ancillary services market, Wholesale market) managed by a Transmission System Operator (TSO) or Balance Responsible Party (BRP), will also be performed.

2.2.3 Services-related components / Decision-making, optimisation and control layer

These components make use of input data mainly stored to the LEM/LFM Repository, in order to create models and optimise the operation of assets or generate load and flexibility forecasts. In case the produced outputs of these components need to be subsequently utilised by other PARITY component(s), they are sent to be stored to the LEM/LFM Repository.

2.2.4 User Interfaces / Application layer

Prosumer Applications and LEM/LFM Management Application are standalone user interface applications that need to communicate with the LEM/LFM Repository or utilize Application Programming Interfaces (APIs) provided by other components in order to acquire data or send commands and user settings. Furthermore, the user interfaces of DSO Toolset and Aggregator Toolset will be part of the respective components and will retrieve the information to display from internal sub-components or directly from the LEM/LFM Repository.

3. INTEGRATION STATUS

This section presents the integration status of each corresponding PARITY component. In particular, for each component, a short functional description is initially provided, followed by details about the connections with other components in a table. Then, the data flow concerning the component is described. Integration and development status (progress and pending actions), as well as any problems encountered, are presented in the three last tables.

3.1 Information Management Layer (IML) cloud

3.1.1 Functional Description

The Information Management Layer (IML) cloud is the system component in charge of gathering real-time data from the buildings that participate in the PARITY trials. The IML cloud handles, in a secure way, all the data generated by downstream available IoT devices (sensors, controllers, measuring devices, etc.) streamed through the IoT gateway installed at prosumers' premises. More specifically, the IML cloud performs the following key functions:

- Facilitates/enables the communication (and information flow) between the IoT infrastructure installed at participants' premises and other PARITY components, such as the PARITY LEM/LFM Repository, the Building-as-a-Battery (BaaB) Application, and the Prosumer Flexibility Manager (PFM), with all of which it interfaces.
- Stores, securely handles, processes and applies cleansing (outlier detection and outlier treatment) and transformation techniques to data from available prosumer-level IoT infrastructure, passed to the IML cloud through the IoT gateway.

The IML is part of the integrated Hypertech PARITY solution (the integration with the IoT gateway is already established, tested and validated prior to any pre-validation tests and pilot rollouts and has already been described as part of PARITY's deliverable D7.1). As such, the integration of the IML cloud with other PARITY components will be further detailed below. Such components are: i) the LEM/LFM repository, ii) the BaaB App component, and iii) the Prosumer Flexibility Manager component, which is part of the Aggregator Toolset.

3.1.2 Connection with other Components and Interfaces

Component	Connection type	API protocol	Data type	Comments
LEM/LFM Repository	TCP/IP	HTTP/HTTPs (REST)	JSON	The IML cloud communicates with the LEM/LFM Repository to send real-time information on the status of the end-user devices.
BaaB App	TCP/IP	HTTP/HTTPs (REST/MQTT)	JSON	Through the BaaB App, the end-user sends control signals to applicable/available downstream controllable BaaB devices through the IML cloud.
Prosumer Flexibility Manager (Aggregator Toolset)	TCP/IP	HTTP/HTTPs (REST)	JSON	The PFM sends control signals concerning available prosumer-level controllable assets to the IML cloud (which are then passed to relevant downstream DER devices).

3.1.3 Data Flow

The IML cloud receives real-time status information from all available prosumer-level controllable devices, which are then passed to the PARITY LEM/LFM Repository where all this information is kept (all historical data on prosumer-level assets can then be retrieved by any PARITY component from the LEM/LFM Repository).

The BaaB Application component interfaces with the IML cloud in order to pass any control actions/signals (on/off signals, setpoints, etc.) that the user has selected downstream, i.e., to available controllable BaaB devices, namely the stationary battery, the HVAC systems lighting and the domestic hot water boilers. This enables the remote control of such controllable devices by the PARITY participating users. It should be noted that this interfacing is preferable to the originally described interface between the BaaB App and the IoT gateway in D3.5, as it streamlines and simplifies communications, and, as such, is the adopted way forward for the PARITY purposes.

Similarly, post prosumer-level optimisation control actions (determined by the Prosumer Flexibility Manager) are sent by the Prosumer Flexibility Manager (PFM) to the IML cloud and from there to the relevant downstream controllable end-user assets, such as the HVAC, DHW boilers and lighting. It should be noted that there is a deviation between the originally foreseen interfacing between the PFM component and the IoT gateway, which safeguards and simplifies the communication between the Aggregator Toolset and prosumer-level systems.

3.1.4 Integration Status

The following table describes the integration status between the IML cloud and the LEM/LFM Repository:

Integration Status	☐ **Final** ☒ **Under development**
Format for Integration	Service integration using RESTful web services
Progress up to date	• Development of initial interface Common Data Model • Testing of initial Common Data Model internally with mock services endpoints
Pending Integration Actions	• Integration with actual/real endpoints, • Correction of any Common Data Model misalignments that may occur • End-to-end integration test

The following table describes the integration status between the IML cloud and the BaaB App:

Integration Status	☐ **Final** ☒ **Under development**
Format for Integration	Service integration using RESTful web services and / or MQTT message broking
Progress up to date	• Development of initial interface Common Data Model • Testing of initial Common Data Model internally with mock services endpoints
Pending Integration Actions	• Integration with actual/real endpoints • Correction of any Common Data Model misalignments that may occur • End-to-end integration test

The following table describes the integration status between the IML cloud and the Prosumer Flexibility Manager component:

Integration Status	☐ **Final** ☒ **Under development**
Format for Integration	Service integration using RESTful web services
Progress up to date	• Development of initial interface Common Data Model • Testing of initial Common Data Model internally with mock services endpoints
Pending Integration Actions	• Integration with actual/real endpoints • Correction of any Common Data Model misalignments that may occur • End-to-end integration test

3.1.5 Development Status

The following table provides a brief overview of the development status of the interfaces among the IML cloud and the aforementioned three components, namely the LEM/LFM Repository, the BaaB App and the PFM.

Development Status	☐ **Final** ☒ **Under development**
Programming Language	JAVA
Progress up to date	• Development of initial interface Common Data Model • Testing of initial Common Data Model internally with mock services endpoints
Pending Development Actions	• Integration with actual/real endpoints • End-to-end integration test

3.1.6 Problems Encountered

No major problems were encountered so far with regards to the integration among the IML cloud and the LEM/LFM Repository, the BaaB App and the PFM. Minor adjustments were made to the data models of the aforementioned services.

3.2 LEM/LFM Repository and off-chain services

3.2.1 Functional Description

The LEM/LFM Repository is part of the LEM/LFM platform, and is responsible for storing measurements, information about users as well as off-chain data that are needed for the market operation. It provides a set of web-services that have been described in the deliverable D5.3 – *PARITY Off-chain Components*. Data models have been defined in JSON format for the representation of several entities are shown in the figure below.

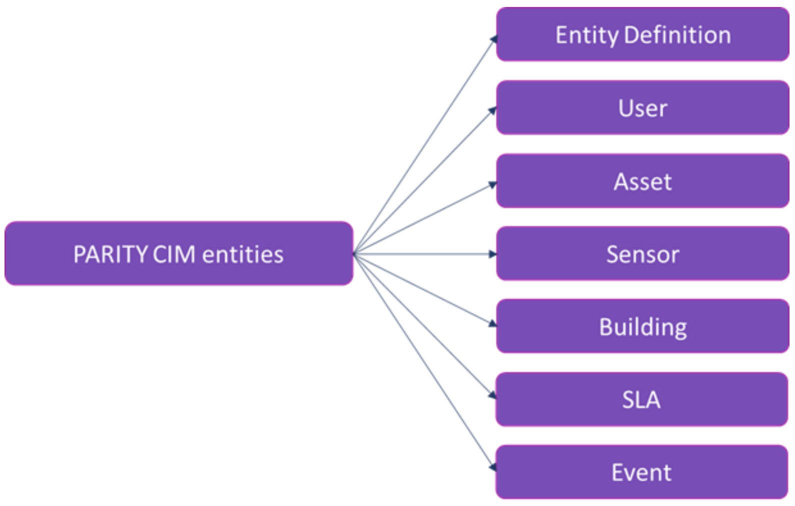

Figure 4. Entities of PARITY Common Information Model.

Additional off-chain services provide anonymization services to Blockchain Agents, as well as other two main functionalities, which are the addition of new prosumers and their assets into the portfolio, and access to user wallets.

3.2.2 Connection with other Components and Interfaces

Component	Connection type	API protocol	Data type	Comments
IML cloud	TCP/IP	HTTP(S)	JSON	Measurement and device status data are received from the IML cloud
Blockchain Agent	TCP/IP	HTTP(S)	JSON	Generated pseudonyms are returned to Blockchain Agent on request

Oracle	TCP/IP	HTTP(S)	JSON	Oracle retrieves various data regarding users, Service Level Agreements (SLAs), measurements, and anonymised IDs
EV profiling and smart charging	TCP/IP	HTTP(S)	JSON	Static data and dynamic events related to EVs and usage of EV charging stations
Stationary Battery Manager	TCP/IP	HTTP(S)	JSON	Retrieves stored measurements to be used as input
PV Manager	TCP/IP	HTTP(S)	JSON	Use of LEM/LFM Repository API for receiving, sending and storing data
Aggregator Toolset	TCP/IP	HTTP(S)	JSON	Retrieves data about prosumers (users), assets, load and flexibility forecasts
DSO Toolset	TCP/IP	HTTP(S)	JSON	Stores events related to grid state and retrieves data related to active network management and LEM/FLM
User Interfaces (Prosumer applications & LEM/LFM Management application)	TCP/IP	HTTP(S)	JSON	Use of LEM/LFM Repository API to retrieve already stored data, check user logins, or store new entities (user, asset types) or user preferences

3.2.3 Data Flow

The data flow concerning the LEM/LFM Repository and off-chain services is bi-directional and is initiated by received requests from the other PARITY components. The requests are performed on-demand, concern data retrieval or storage, and are initiated by the other PARITY components. Some of the requests for data storage and retrieval are performed periodically in 15-minute intervals, according to market operation interval and produced load/generation measurements and forecasting events.

3.2.4 Integration Status

Integration Status	☐ **Final** ☒ **Under development**
Format for Integration	Service integration using REST web-services
Progress up to date	• All web services that have been developed are online and operational • Basic tests of the web services have been made
Pending Integration Actions	• Finalize integration with external weather data service • Finalize integration with the PARITY Blockchain Agent

3.2.5 Development Status

Development Status	☐ Final ☒ Under development
Programming Language	Java
Progress up to date	• Implementation of all required web services and their interactions with the database management system (MongoDB) • Deployment of the application on a server for internal testing
Pending Development Actions	• Finalize authentication / authorization support for all web services • Enable the use of secure connections (SSL/TLS) • Finalize data models

3.2.6 Problems Encountered

Failure description	Failure cause	Countermeasure
Could not fetch weather historical data from external weather API	Connection timeout is returned in some cases	Modify connection timeout limit or consider using an alternative weather data service

3.3 Blockchain Agent

3.3.1 Functional Description

The detailed functionalities of the Blockchain Agent are explained in the paragraphs 2.3 – 2.8 of deliverable D5.4 – *PARITY LFM-enabling Smart Contracts.*

3.3.2 Connection with other Components and Interfaces

Component	Connection type	API protocol	Data type	Comments
Oracle	TCP/IP	HTTP	JSON	-
Aggregator Toolset	TCP/IP	HTTP	JSON	-
Off-chain services component Interface	TCP/IP	HTTP	JSON	-
DSO Toolset	TCP/IP	HTTP	JSON	-

3.3.3 Data Flow

The Blockchain Agent receives data from the PARITY components described in the previous sub-section every fifteen minutes. The acquired information is then saved in the sidechain and used to periodically solve the markets, i.e. calculate buying and selling prices based on power generation and demand to perform market clearing. After the aforementioned solution, the PARITY components that are interacting with the Blockchain Agent will be able to download the information related to the results from the sidechain. The reader can refer to deliverable D5.4 – *PARITY LFM-enabling Smart Contracts* for additional details.

3.3.4 Integration Status

Integration Status	☐ **Final** ☒ **Under development**
Format for Integration	Service integration using REST web-services
Progress up to date	• Integration with the Off-chain pseudonymizer service
Pending Integration Actions	• Finalize integration with the DSO toolset • Finalize integration with the Aggregator Toolset • Finalize integration with the Oracle

3.3.5 Development Status

Development Status	**☐ Final** **☒ Under development**
Programming Language	Python/Go
Progress up to date	• Implemented sidechain smart contracts • Implemented Anonymizer application module • Implemented Market Engine application module • Implemented SLA Checker application module
Pending Development Actions	• Implementation of Balance Controller application module • Implementation of Payment Engine application module

3.3.6 Problems Encountered

No major problems were encountered so far.

3.4 Oracle

3.4.1 Functional Description

The Oracle is the component responsible for gathering and providing reliable data to the Blockchain Platform in order to execute the Smart Contracts and run the Market Engine in a secure and trustful way. It acts as a middleware with a bridge functionality between the isolated ecosystem of the Blockchain and the real world. More specifically the Oracle performs the following key functions:

- Facilitates the information flow between the LFM/LEM Repository and the Blockchain Platform.
- Retrieves KPIs from the Smart Contracts for the registered users, requests the necessary metering and sensing data from the LFM/LEM Repository, calculates the KPIs and provides this information to the Blockchain Platform.
- Provides measurements and forecasts to feed the Market Engine of the Blockchain Platform.

The Oracle is part of the integrated PARITY solution and shares two interfaces with other components. The first interface is established between the Oracle and the LEM/LFM Repository and communicates data regarding registered users, anonymization, SLAs, sensing and metering data originating from the IoT ecosystem. The second interface, which is between the Oracle component and the blockchain agent, is a unidirectional interface that facilitates the secure transmission of KPIs, measurements and forecasts to the BA. The two interfaces are described in detail below.

3.4.2 Connection with other Components and Interfaces

Component	Connection type	API protocol	Data type	Comments
LEM/LFM Repository	TCP/IP	HTTP/HTTPs (REST)	JSON	The Oracle communicates with the LEM/LFM Repository to communicate data regarding registered users, anonymization, SLAs, sensing and metering data.
Blockchain Agent	TCP/IP	HTTP/HTTPs (REST)	JSON	Bidirectional interface that facilitates the secure transmission of KPIs, measurements and forecasts regarding the operation of Smart Contracts and the Market Engine.

3.4.3 Data Flow

The LEM/LFM Repository database is the component responsible for holding various information regarding the registered users. This information, which is essential for the operation of Oracle is the user’s role, the available equipment, the SLAs, pseudonym association, and all the measurements and forecasts deriving from the IoT equipment.

At first, the Oracle, through a dedicated Configuration sub-component requests user information such as the available and the already registered equipment using the Web Service Endpoint of the LEM/LFM Repository. When this information is retrieved, then it requests information regarding the SLAs that are associated with the given user. The SLAs as a form of a digital contract holds information such as the ID of the agreement, the involved parties, the starting and the ending date of the contract and the KPIs that need to be calculated, which are extracted from the Oracle. For every SLA that is valid for the given time period, the SLA Manager subcomponent of the Oracle, identifies which metrics are needed for the calculation of the KPIs, and then sends a requests for the associated data to the LEM/LFM Repository. The response LEM/LFM Repository includes various types of metrics such as power generation, ambient conditions, energy consumption, or forecasts that are used for the calculation of the KPIs. This information is anonymised and communicated from the Oracle to the Blockchain Platform for the activation of the Smart Contracts. A very similar approach is followed for the provision of the measurements and forecasts that are necessary for the operation of the Blockchain Market Engine mechanism. The Oracle retrieves the energy consumption and production forecasts, and measurements which are collected from the Web Service Endpoint of the LEM/LFM Repository, and sends them anonymised to the Blockchain Agent.

3.4.4 Integration Status

The following table describes the integration status between the Oracle and the LEM/LFM Repository:

Integration Status	☐ Final ☒ Under development
Format for Integration	Service integration using RESTful web services
Progress up to date	• Development of the first version of the interface Common Data Model • Testing of the first version of the Common Data Model internally with mock services endpoints
Pending Integration Actions	• Refinement and finalization of Common Data Model • Integration with actual/real endpoints • End-to-end integration test

The following table describes the integration status between the Oracle and the Blockchain Agent:

Integration Status	☐ Final ☒ Under development
Format for Integration	Service integration using RESTful web services
Progress up to date	• Development of the first version of the interface Common Data Model • Testing of initial Common Data Model • Integration with actual/real endpoints
Pending Integration Actions	• Refinement and finalization of Common Data Model • End-to-end integration test

3.4.5 Development Status

Development Status	☐ Final ☒ Under development
Programming Language	Node.js
Progress up to date	• Development of initial interface Common Data Model • Testing of initial Common Data Model internally with real services endpoints
Pending Development Actions	• Refinement and corrections of Common Data Model • End-to-end integration test

3.4.6 Problems Encountered

No major problems were encountered so far with regards to the integration between the Oracle with the LEM/LFM repository and the Blockchain Agent.

3.5 EV Profiling and Smart Charging

3.5.1 Functional Description

The Optimization Engine is the main sub-component of the EV Profiling and Smart Charging component. As it is shown in the figure below, it is composed of several modules in order to implement all required functionalities.

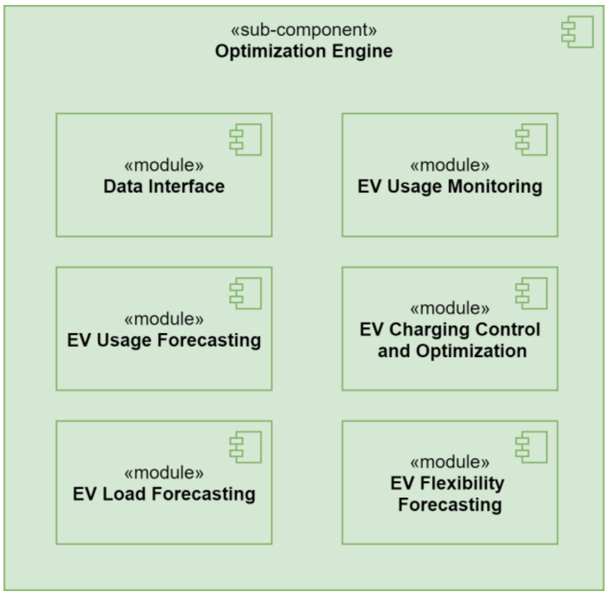

Figure 5. Optimization Engine of the EV Profiling and Smart Charging component.

The modules and their functionalities are the following:

- Data Interface: this module is responsible for exchanging data with the LEM/LFM Repository. This module is capable of interfacing and handling external data sources, used to test the Optimization Engine component. For example:
    - Dataset provided by CIRCE to evaluate different future scenarios of EV penetration in the pilots' networks;
    - ACN-Data dataset [2]: public dataset provided by Caltech with real EV charging sessions
- EV Usage Monitoring: this module is responsible for monitoring data coming from EV chargers. In case new events (connection, input from user, disconnection, state of charge update) are stored in the LEM/LFM Repository, it collects and provides them to the EV Usage Forecasting and EV Charging Control and Optimization modules.
- EV Usage Forecasting: this module is responsible to estimate the state of charge at plug time (SOC0) of the EV, whether it is not provided by EV chargers, and to estimate the time when

the user will unplug the EV (T1). It meets the functional requirements R1 and R5 presented in D7.3.

- EV Charging Control and Optimization: this module is responsible for managing a set of EV charging stations with the objective of servicing the majority of charge requests as a primary objective while either minimizing the charging costs according to a dynamic tariff scheme or the impact that such charges have on the electric grid implementing peak shaving. It meets the functional requirements R9, R11 and R12 presented in D7.3. Both V1G and V2G schemes are available.
- EV load forecasting: this module is able to predict the forecasted load on the EV charging station for a configurable time horizon, based on data from past charging sessions. Seasonal patterns (hourly, daily) are taken into consideration.
- EV flexibility forecasting: this module is responsible for estimating the flexibility potential of the charging station. User preferences are taken into consideration and the flexibility potential for a specific time horizon is calculated for each session individually.

Apart from the Optimisation Engine, this component will also provide an API in order to receive load management requests from the DER dispatch module of the Aggregator Toolset.

3.5.2 Connection with other Components and Interfaces

Component	Connection type	API protocol	Data type	Comments
LEM/LFM Repository	TCP/IP	HTTP	JSON	API of LEM/LFM Repository is used to retrieve input data and store
Aggregator Toolset	TCP/IP	HTTP	JSON	HTTP REST API will be provided to receive on demand signals from the DER dispatch module of the Aggregator Toolset
EV charging station interface	TCP/IP	HTTP or OCPP	JSON	This interface will be used for retrieving events and storing them to the LEM/LFM Repository, or sending commands to the EV charging stations

3.5.3 Data Flow

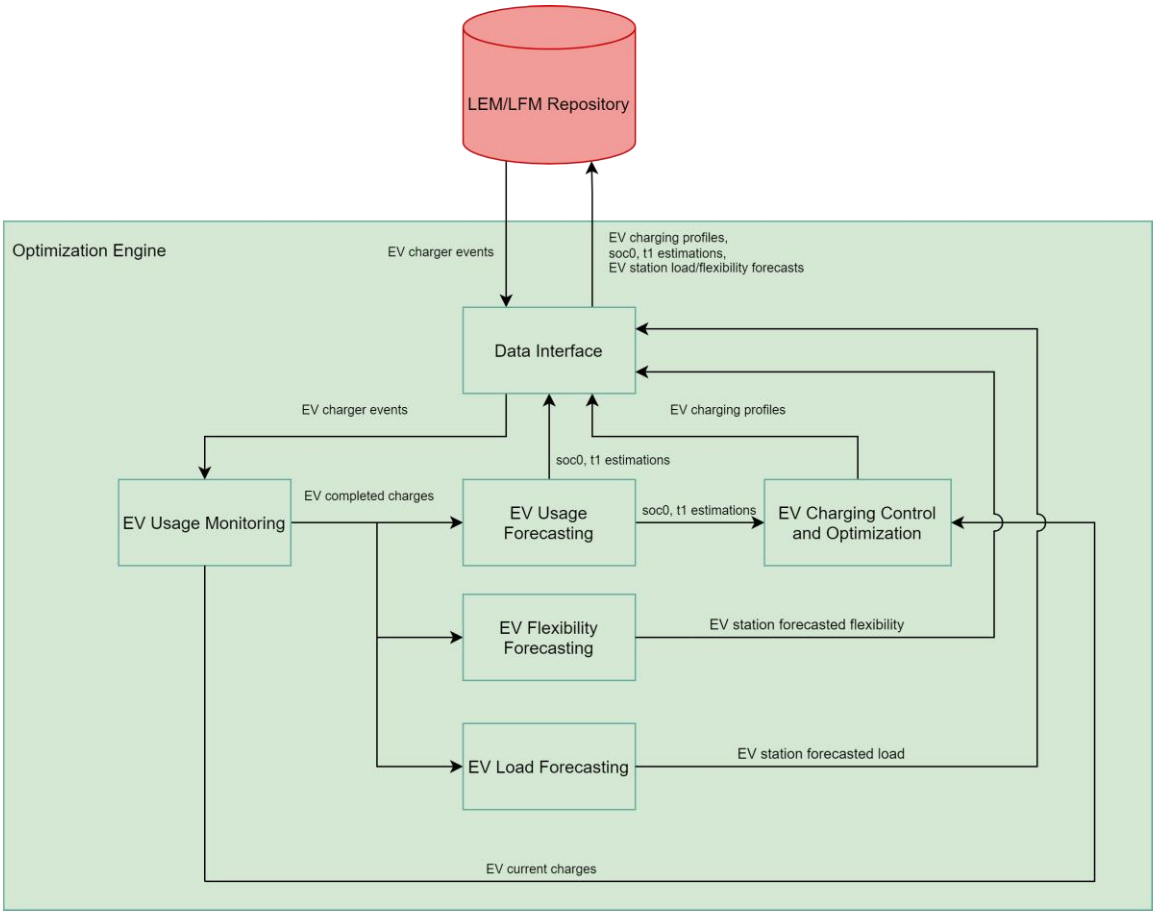

Figure 6. Information exchange diagram for the Optimization Engine of the EV Profiling and Smart Charging component.

EV charging events are stored in the LEM/LFM Repository according to the data models designed and implemented within T7.3. Events are retrieved by the Optimization Engine component (Figure 6) through the Data Interface and analysed by the EV Usage Monitoring module, which is responsible for providing the list of completed charging sessions to the EV Usage Forecasting and the list of active charging sessions and the related details to the EV Charging Control and Optimization module.

EV Usage Forecasting gets the list of all the completed charging sessions from the EV Usage Monitoring module, and provides the estimations of soc0 and t1 to the EV Charging Control and Optimization module. This data is also stored in the LEM/LFM Repository through the Data Interface.

EV Charging Control and Optimization generates the charging schedules, given the set of all the estimated and active charging requests (defined by their soc0, t1 and required charging amounts). The EV Charging Control and Optimization considers as input data (depending also on the selected objective): a dynamic price profile for both charging and discharging power (in case V2G is enabled), an estimated load profile for each charging station and the running observed maximum power, in case power-based tariffs are defined. These are also stored in the LEM/LFM Repository through the Data Interface.

EV load forecasting module gets the list of the active charging sessions and the user preferences of each session. Past values are also retrieved as they are used to the predictive model. Additional features are extracted from the timestamp, as seasonal patterns are also taken into consideration to the predictive model. Forecasted values are predicted and finally stored back to the LFM repository.

The flexibility forecasting module is able to periodically calculate the flexibility potential of the charging station, so that it can be used by the Aggregator when necessary. Flexibility can be provided either by modifying the current charging schedule of the EVs, and more specifically by time shifting their charging session, or by activating V2G mode and discharge a vehicle's battery, injecting energy back to the grid. The capability of each vehicle to participate in the flexibility request is determined by the preferences set by the user (requested kWh, requested departure time). If it is impossible to satisfy those user preferences, then the vehicle is excluded from the estimation of the potential flexibility and continues its charging session according to the initial schedule. Once the amount of potential flexibility is estimated for a certain time window, this information is stored to the LEM/LFM repository. If the Aggregator finally decides to activate this amount of flexibility, the participating EVs will need to modify their charging schedule accordingly.

3.5.4 Integration Status

Integration Status	☐ **Final** ☒ **Under development**
Format for Integration	Currently under discussion
Progress up to date	• Designed and implemented data models to collect data in the LEM/LFM Repository from EV charging stations, EV prosumer application and Optimization engine components, and to share with other components. • Data Interface with the LEM/LFM Repository • EV Charging Control and Optimization module has been developed. It accounts for power-based and dynamic tariffs, to deal with pilot use cases. It also accounts for peak shaving use cases.
Pending Integration Actions	• Testing Data Interface with the LEM/LFM Repository • Testing modules on the field

3.5.5 Development Status

Development Status	☐ **Final** ☒ **Under development**
Programming Language	Python
Progress up to date	Design and developed SW infrastructure • EV Data Interface module implemented and tested with ACN-Data public dataset • EV Usage Monitoring module implemented and tested with ACN-Data public dataset • EV Charging Control and Optimization implementation • EV load forecasting module implemented and tested with ACN-Data public dataset

Pending Development Actions	• EV Usage Forecasting implementation • Refine modules with real data coming from pilots • EV V2G smart charging mode implementation

3.5.6 Problems Encountered

Failure description	Failure cause	Countermeasure
No data has been used from the EV chargers at pilots	Not all needed parameters were available and the activities are ongoing	A public dataset with real charging sessions (ACN-Data, provided by Caltech) has been used for implementing and testing the Optimization Engine modules

3.6 PV Manager

3.6.1 Functional Description

The PV Manager is monitors and controls the PV installation at the site. It has two sub-components, i.e., PV Forecasting Engine and PV Monitor.

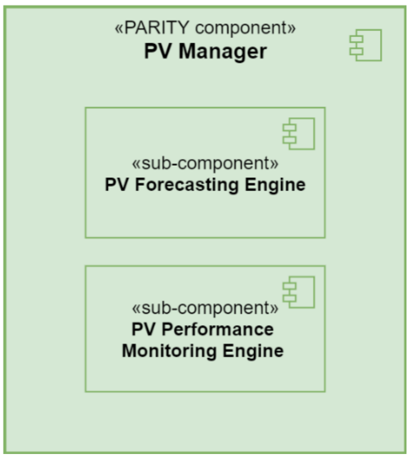

Figure 7. PV Manager component diagram.

The PV Forecasting engine creates PV power output generation forecasts using historical PV production data and weather data. The PV Monitor keeps track the PV generation that can be used by the PV Forecasting Engine or other PARITY components trough the LEM/LFM Repository.

3.6.2 Connection with other Components and Interfaces

Component	Connection type	API protocol	Data type	Comments
LEM/LFM Platform (Repository)	TCP/IP	HTTP	JSON	Use of Repository API for receiving, sending and storing data.
DER Dispatch Module	TCP/IP	HTTP	JSON	For correct flexibility dispatch operation

3.6.3 Data Flow

The PV Manager gets site and weather data from the LEM/LFM Repository for its PV Forecasting Engine subcomponent which can create PV generation forecasts. The data from the PV Manager is sent back to the LEM/LFM Repository for updates and to the DER Dispatch Module of Aggregator Toolset so it can perform appropriate adjustments regarding flexibility delivery.

3.6.4 Integration Status

Integration Status	☐ **Final** ☒ **Under development**
Format for Integration	Currently under discussion
Progress up to date	• Integration validation has been successfully performed on non-PARITY sites
Pending Integration Actions	• Pending Integration on the PARITY sites.

3.6.5 Development Status

Development Status	☐ **Final** ☒ **Under development**
Programming Language	C# and Python
Progress up to date	• PV Monitor development is complete
Pending Development Actions	• Some development required on the PV forecasting Engine

3.6.6 Problems Encountered

No major problems were encountered so far.

3.7 Stationary Battery Manager

3.7.1 Functional Description

The Stationary Battery Manager (SBM) takes input from the LEM/LFM repository where applicable and from the meters installed on site. The local site condition is important for the SBM to optimize its operation and implement Peak Shaving locally and interact with the aggregator tool set for provision of flexibility services.

The SBM has three engines dedicated at performing their designated function, i.e. Peak Shaving Forecasting Engine, Peak Shaving Real Time Engine and Flexibility Response Real Time Engine.

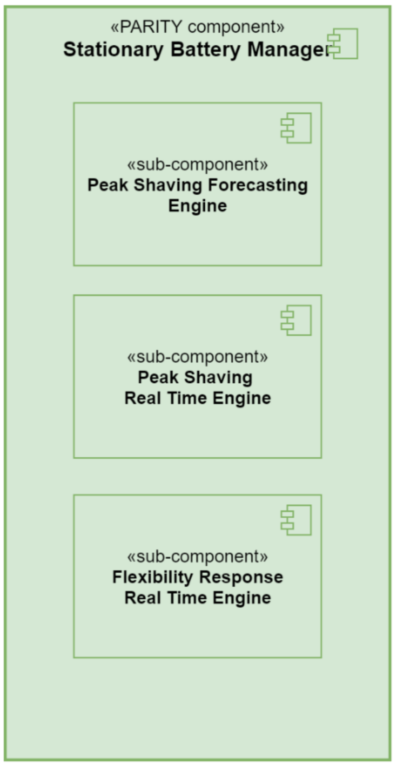
«PARITY component»
Stationary Battery Manager
«sub-component»
Peak Shaving Forecasting Engine
«sub-component»
Peak Shaving Real Time Engine
«sub-component»
Flexibility Response Real Time Engine

Figure 8. Stationary Battery Manager component diagram.

The Peak Shaving Forecasting Engine uses the LEM/LFM Repository and its local meter data store to generate a forecast for the expected Peak Shaving activity ahead of time. A power threshold limit is set beyond above which the power is expected to be shaved. This forecast allows for an organized dispatch planning of the SBM and provides information to the Aggregator Toolset with the possible flexibility service delivery options. It also avoids the conflict of having to perform Peak Shaving and deliver on flexibility service at the same time.

The Peak Shaving Realtime Engine performs the Peak shaving service in real-time as the designated time arrives. It uses the information available from the Peak Shaving Forecasting Engine and the real-time data available from the onsite meters to perform the Peak Shaving service by sending charge/discharge commands to the BMS.

The Flexibility Response Real Time Engine uses the inputs from the Aggregator Toolset for the charge/discharge level of the BESS and sends these commands to the BMS for the delivery of the service.

3.7.2 Connection with other Components and Interfaces

Component	Connection type	API protocol	Data type	Comments
LEM/LFM Platform (Repository)	TCP/IP	HTTP	JSON	Use of Repository API
DER Dispatch Module	TCP/IP	HTTP	JSON	To Aggregator Toolset for flexibility dispatch operation

3.7.3 Data Flow

The onsite meter data is stored at the LEM/LFM repository. This data is used to perform the functions within the Peak Shaving Forecasting Engine. The results are again stored in the LEM/LFM repository and later retrieved by Peak Shaving Real Time Engine and Flexibility Response Real Time Engine as per the requirement. The Flexibility Response Real Time Engine gets the information from the Aggregator Toolset with regards DER Dispatch. All the information from the real-time operations is passed to the BMS controlling the BESS which can either charge or discharge the BESS depending on the instructions.

3.7.4 Integration Status

Integration Status	☐ **Final** ☒ **Under development**
Format for Integration	Currently under discussion
Progress up to date	• The SBM has been developed • Peak shaving has been integrated and validated on non-PARITY sites • Frequency Response pre-qualification is under process. Should be completed soon
Pending Integration Actions	• Integration of SBM on PARITY sites

3.7.5 Development Status

Development Status	☐ **Final** ☒ **Under development**
Programming Language	C# and Python
Progress up to date	• Peak Shaving Real Time Engine development complete • Frequency Response Real Time Engine development complete
Pending Development Actions	• Partial development remaining on Peak Shaving Forecasting Engine

3.7.6 Problems Encountered

Failure description	Failure cause	Countermeasure
Communication problems	Incorrect protocol	Corrected the communication protocol

3.8 Aggregator Toolset

3.8.1 Functional Description

In smart grids the Aggregator is responsible for grouping small and medium size Distributed Energy Resources (DERs) of final prosumers, in order to optimally coordinate their operation and facilitate participation in ancillary services (AS) and wholesale markets (WS). Additionally, in the context of PARITY, the Aggregator Toolset aims to optimize the management of available flexibility resources in order to provide services to the Distribution System Operator (DSO) of the grid based on grid network's condition. The DSO makes use of a Traffic Light Concept (TLC) so as to reflect the grid's needs to certain actions by the Aggregator.

In PARITY, the condition of the grid and the corresponding action required on Aggregators' level can be described by the following three scenarios. In cases that the grid is not under any stress, it is under green state, which translates to having an active Local Energy Market (LEM) as well as to the Aggregator being able to participate in AS and WS markets. In cases that some stability violations are foreseen within the grid, the DSO forwards a yellow state, which translates to activation of the implicit Local Flexibility Market (LFM) as well as deactivation of participation of the Aggregator in AS and WS markets. Finally, in cases when the grid is in emergency, the DSO alerts a red state during which any other market-based activity is paused and the Aggregator is responsible to serve instant DR signals requested by the DSO, based on bilateral contracts.

The overall operation of the Aggregator Toolset requires communication with other components and flow of information in a certain format. These components, together with details about the integration and development status, are presented in the following subsections.

3.8.2 Connection with other Components and Interfaces

Component	Connection type	API protocol	Data type	Comments
Blockchain Agent	TCP/IP	HTTP	JSON	API is provided by the Blockchain Agent side
DSO Toolset	TCP/IP	HTTP	JSON	API is provided by the Aggregator side
BRP / TSO Interface	TCP/IP	HTTP	JSON	API is utilized by the Aggregator Toolset to interact with external markets (emulated)
IoT Gateway	TCP/IP	HTTP	JSON	API is utilized for sending control commands
EV profiling and smart charging	TCP/IP	HTTP	JSON	API is utilized by the DER dispatch module of the Aggregator Toolset
Stationary Battery Manager	TCP/IP	HTTP	JSON	API is utilized for sending control commands
LEM/LFM Repository	TCP/IP	HTTP	JSON	API of LEM/LFM Repository is utilized to retrieve various data: users, measurements, forecasts and other

3.8.3 Data Flow

Communication between the Aggregator Toolset and other subcomponents is supported by API protocols as described in the table of the previous subsection. However, in the context of evaluating the current integration status of the PARITY project it is useful to evaluate the overall flow of information related to the Aggregator Toolset.

The Aggregator Toolset consists of subcomponents that facilitate its functionalities. For example, two core subcomponents are the VPP Manager and the Prosumer Flexibility Manager. Moreover, information exchange is required with other actors such as the Blockchain Agent, the TSO and BRP interface, the LEM/LFM Platform, the DSO Toolset and the IoT Gateway. It should be noted that, in all of these cases, information can be exchanged via the use of HTTP REST APIs.

3.8.4 Integration Status

Integration Status	**☐ Final** **☒ Under development**
Format for Integration	Web services
Progress up to date	• Architecture of communication of the Aggregator Toolset with DSO Toolset, Prosumer Flexibility Manager and LEM/LFM Platform has been decided
Pending Integration Actions	• The existing APIs will be tested and evaluated in an end-to-end scenario. Specifically, the behaviour of DSO Toolset, Aggregator Toolset, LEM/LFM Platform and Prosumer Flexibility Manager can be evaluated via an end-to-end scenario that involves all three states of the TLC • The format of communication with the DSO only needs to be finalised • Blockchain Agent and IoT Gateway need to be integrated

3.8.5 Development Status

Development Status	**☐ Final** **☒ Under development**
Programming Language	Python
Progress up to date	• The API for communication between the Aggregator Toolset and LFM Platform has been developed • The way that DR signals are to be communicated from DSO has been decided, and endpoint for DR event publishing has been established • API for assets'/users' registration, as well as flexibility events has been developed on the Aggregator Toolset side
Pending Development Actions	• APIs that are still under development need to be finalised and then tested and evaluated

3.8.6 Problems Encountered

No major problems were encountered so far.

3.9 DSO Toolset

3.9.1 Functional Description

As described in the architecture definition of PARITY (D3.5), the DSO Toolset is the component of the PARITY architecture that will be in charge of allowing the DSO to monitor the grid, applying the necessary management actions and interacting with the LEM/LFM. Furthermore, the Toolset will be responsible for providing grid constraints to be integrated with market conditions to reduce harmful effects on the local grid from the flexibility activation actions. It is composed of the following sub-components shown in Figure 9:

• **User Interface**: The User Interface will display information that will allow the DSO user to monitor network state and make specific requests to procure flexibility.

• **Smart Marketplace Module**: It will include the requests that are needed for sending and retrieving data to/from the LEM/LFM Platform. Returned information will then be visualized through the UI.

• **Network Monitoring Tool**: It will be responsible for monitoring grid parameters related to power quality, consumption, and generation. Grid network state DER-setpoints will be sent according to the traffic light approach, based on the monitored parameters.

• **Active Network Management**: ANM tool will monitor and manage the distribution networks under normal operation conditions and under unforeseen events. It will capture any technical issue early on, before it escalates and achieve a reduction in the likelihood of costly interruptions to grid operation.

• **DER Dispatch Module**: It will be responsible for sending fully automated control signals to any flexibility assets controlled directly by the DSO, previously calculated by the ANM.

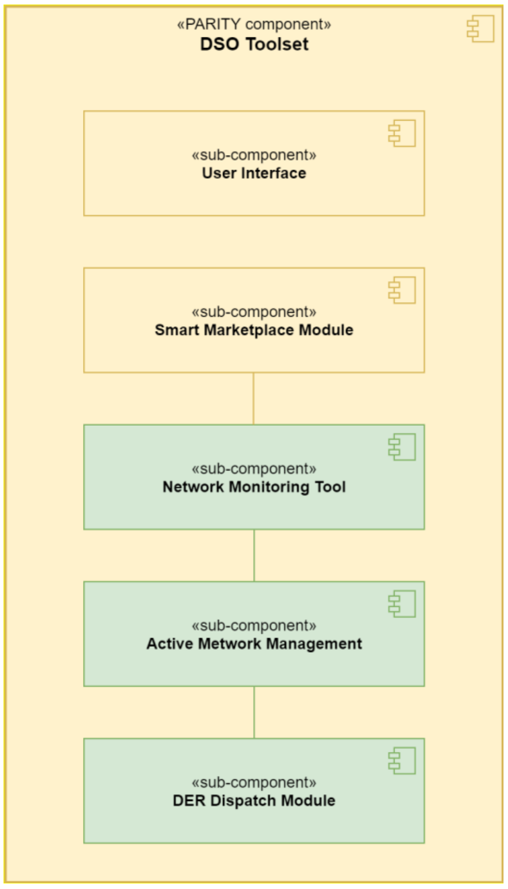

Figure 9. DSO Toolset component diagram.

3.9.2 *Connection with other Components and Interfaces*

Component	Connection type	API protocol	Data type	Comments
D-STATCOM	TCP/IP	HTTP/HTTPS (REST/MQTT)	JSON	TBD
Blockchain Agent	TCP/IP	HTTP/HTTPS (REST/MQTT)	JSON	TBD
LEM/LFM Platform	TCP/IP	HTTP/HTTPS (REST/MQTT)	JSON	TBD
Aggregator Toolset	TCP/IP	HTTP/HTTPS (REST/MQTT)	JSON	TBD
DSO Network Monitoring System	TCP/IP	HTTP/HTTPS (REST/MQTT)	JSON	TBD
Stationary Battery Management	TCP/IP	HTTP/HTTPS (REST/MQTT)	JSON	TBD
EV Profiling and Smart Charging	TCP/IP	HTTP/HTTPS (REST/MQTT)	JSON	TBD
TSO Interface	TCP/IP	HTTP/HTTPS (REST/MQTT)	JSON	TBD

3.9.3 Data Flow

DSO Toolset communicates with various components through its subcomponents that are mentioned next. The communication with the LEM/LFM Platform is performed via the Smart Marketplace Module, which communicates, via Off chain Tools, using an HTTP REST API for presenting information related to DSO engagement in the implicit LFM market, such as the SLA with the Aggregator and acquired flexibility.

Other communication with the LEM/LFM is performed via the Network Monitoring Tool to acquire information about local consumption and generation from the LEM/LFM Repository. This subcomponent also receives grid monitoring information from the external DSO Network Monitoring System and communicates with D-STATCOM device to retrieve information about its state. The Network Monitoring Tool also sends information such as grid constraints to the market through the Blockchain Agent.

DSO Active Network Management sub-component will be able to send signals for direct load control to the Aggregator Toolset when network is in an emergency state. Alternatively, DSO DER Dispatch Module will be able to perform DER controls (only for DERs that are controlled directly by the DSO) using APIs.

3.9.4 Integration Status

Integration Status	☐ **Final** ☒ **Under development**
Format for Integration	To be discussed
Progress up to date	• The DSO Toolset is still under development, being the Machine-To-Machine (M2M) communications one of the developments still to be implemented. Currently only the communication with the D-STATCOM is already clarified.
Pending Integration Actions	• Discussion with the rest of the components to define the format of the communications • Development of the solution agreed with the partners

3.9.5 Development Status

Development Status	☐ **Final** ☒ **Under development**
Programming Language	.Net, Python
Progress up to date	• User Interface defined and pending to be programmed. • The backend of the platform is developed, the rest of the subcomponents have to be integrated with it.
Pending Development Actions	• Integration of the subcomponents in the platform. • Development and test of the user interface. • Definition and development of the communications.

3.9.6 Problems Encountered

No major problems were encountered so far.

3.10 D-STATCOM

3.10.1 Functional Description

The D-STATCOM is a regulating device used on alternating current electricity, a power electronics converter that acts as either a source or sink of reactive power in the electric grid.

A STATCOM provides variable reactive power, inductive or capacitive, to adjust the voltage of the busbar at the point of common coupling.

The proposed development has an additional connection to the classical three-leg solutions in the market, allowing the current flow through the neutral wire. Neutral current expands the work range to unbalanced loads, allowing a single-phase reactive power control. Additionally, the main advantage of the STATCOM's unbalanced loads capability is that the active power regulation between phases is allowed if the three-phase power sum is equal to zero.

The four-legs STATCOM presented could be useful in low voltage grids, where unbalanced loads or distributed energy resources could produce unbalanced power flows. In addition, a centralized controller coordinating several 4-leg STATCOMs over the low voltage distribution grid, is a new tool able to provide support to the grid in a smooth way. This new capability allows the DSO a better control over the low voltage distribution grid and its influence over the medium voltage grid, regulating the contribution in the transformation centre connection point.

3.10.2 Connection with other Components and Interfaces

Component	Connection type	API protocol	Data type	Comments
DSO Toolset	TCP/IP	HTTP/HTTPS (REST/MQTT)	JSON	TBD

3.10.3 Data Flow

The only external communication of the STATCOM will occur with the DSO Toolset. The STATCOM will provide its working status to the Toolset, to centralize the information for the DSO and enable the network operation.

3.10.4 Integration Status

Integration Status	☐ **Final** ☒ **Under development**
Format for Integration	JSON format via API REST
Progress up to date	• Message format with the DSO Toolset is defined, yet to be implemented
Pending Integration Actions	• Development of communication with DSO Toolset • Test to ensure the appropriate integration

3.10.5 *Development Status*

Development Status	☐ **Final** ☒ **Under development**
Programming Language	.Net, Ada
Progress up to date	• The STATCOM is developed, and its communications tested in controlled environments, as CIRCE's laboratory, and integrated with other platforms for other projects.
Pending Development Actions	• Development of communication module. • Test in demo sites.

3.10.6 *Problems Encountered*

No major problems were encountered so far.

4. INTEGRATION ACTIVITIES

4.1 Pre-pilots (Hypertech & CERTH premises)

4.1.1 HYPERTECH lab trials

For the pre-validation activities at the Hypertech lab premises, a simple sidechain constituted by four different nodes is considered in order to test (under a controlled environment in this case) the PARITY market operation. More specifically, four different IoT gateways installed at the Hypertech Labs are used to represent four different blockchain nodes of a local energy community, namely the nodes of the DSO, the aggregator and those of two separate prosumers.

The trial focuses on the communication among the gateway, where the PARITY Oracle and the Blockchain Agent are deployed, the IML cloud and the LEM/LFM repository. The Oracle retrieves from the LEM/LFM repository, through the IML cloud, any KPIs relevant to the SLAs applicable to the specific business scenario (in this case and for the purposes of the pre-pilot trials at the Hypertech Labs, this scenario is the pre-paid one). The SLA checker of the Blockchain Agent uses this information for realizing any KPI deviations and determining prosumer payments or penalties.

The Blockchain Agent deployed on the IoT gateways also gets info from the Oracle on the post-optimisation, prosumer-level power forecasts, as well as actual power measurements to help decide on market prices.

Although the integration between the Blockchain Agent and the DSO and Aggregator Toolset is still under development, the scripts for the necessary data retrieval by the Blockchain Agent from the two aforementioned toolsets have been developed and tested under the PARITY market trials at the Hypertech lab premises. These necessary datasets include, in the case of the data retrieved from the DSO toolset, the grid state and constraints in the form of a traffic light signal, and in the case of the data retrieved by the Aggregator Toolset, the aggregated[1] power forecasts and actual measurements necessary for the calculation of certain KPIs and the market prices.

During the prevalidation activities mentioned above, the following problems were encountered:

- Power cuts affect the operation of the IoT gateways and hence communication between the gateway and the IML cloud and/or other PARITY components. Power cuts could (in a few cases) either permanently damage the SD card contained in the IoT gateways, in which case a replacement SD card is necessary, or prevent the safe automatic rebooting of the gateway, in which case a remote or physical manual reboot of the device is required.
- Potential changes in the IP of prosumer-level IoT gateways may hinder the ability of other PARITY components to communicate with them.

Solutions to these problems are being actively sought in order to mitigate the problems prior to the small-scale trials and certainly before the full roll-out of the PARITY system.

4.1.2 CERTH lab trials

The LEM/LFM Repository, including the off-chain services, were deployed on a dedicated server at CERTH premises, in order to test their functionalities and facilitate the integration with the other PARITY components. Initially, preliminary tests were made in order to ensure and confirm that the developed web services were showing the expected behaviour. To this end, a virtual environment was

[1] This is the aggregated power forecasts and measurements of all relevant prosumers under the aggregator's portfolio.

created, named ‘defaultmarket’, and example data were entered following the corresponding JSON formats according to the specified data models (users, assets, measurements, flexibility forecasts, etc.).

As a next step, the publicly accessible URL address of the server was shared to the partners who develop components that need to interact with the APIs that are provided by the LEM/LFM platform, in order to use during the development phase and implement the integration. Indicative lab trials that were performed with regard to the integration with the LEM/LFM Repository and off-chain services concern the VPP manager of the Aggregator Toolset, the Blockchain Agent which initiates the calls for pseudonym generation of a given entity’s name, and the EV profiling and Smart Charging component. Furthermore, first versions of the user interfaces for the Aggregator Toolset and LEM/LFM Management application were implemented and successfully utilised the LEM/LFM Repository API in order to retrieve and display information, as well as to store new data. For instance, the LEM/LFM Management application was employed to retrieve the latest grid network state event, the number of participants to the local market, and also to setup and edit the asset types, user roles, sensor types, building types, and add the descriptions of new users to the system.

The open platform of SteVe [3], was set up to implement connection with the EV charger available at SmartHouse Digital Innovation Hub at CERTH premises. SteVe is an open and free to use platform, which provides basic functions for the administration of charge points and EV user data. It allows to implement, test and evaluate authentication protocols, reservation mechanisms for charging points, as well as business models for electric mobility. SteVe supports various versions of the Open Charge Point Protocol (OCPP) protocol, while it has been tested and is able to operate with multiple models of EV charging stations. OCPP 1.6 is implemented in the CERTH pre-pilot trials.

It shall be noted that when there is an EV management application already installed at a pilot site, such as in BFS, the use of existing platform’s API for retrieving data and sending control signals will be preferred, in order not to disrupt already operating installations at the pilot.

Regarding the data coming from EV charging sessions, data models have already been defined. Data will be provided in JSON format and they are expected to contain all the necessary measurements coming from the charging session and moreover the EV driver preferences that will affect the charging strategy. All the relevant information will be stored at the LEM/LFM repository.

4.2 Pilot sites

Integration activities that have been performed so far at the four pilot sites in Spain, Greece, Switzerland, and Sweden, refer mostly to the integration of the installed equipment with the IoT Gateway, which is well-documented and pre-tested. Therefore, a detailed description of all the activities to be performed will be provided in the updated version of this deliverable.

5. INTEGRATION TESTING AND RESULTS

This section describes the System Integration Testing (SIT) procedure and results. SIT is defined as a type of software testing that is performed in an integrated hardware and software environment to verify the interactions between the modules of a system.

The selected SIT approach for PARITY is the incremental integration testing. In this type of SIT, each module is initially tested individually and then testing is continued by gradually appending other modules to it. The system is constructed and tested in small segments, allowing detection and fixing of found errors easier. There are two approaches that can be followed for incremental testing: the Top down approach and the Bottom up approach [4].

Top down approach: According to this approach, the testing begins at the topmost module, and the functionality of the modules at a lower level is simulated using stubs. The top module is integrated with the stubs of the modules at the lower levels. Once each test is completed, the respective stub is replaced by the real module. The process is repeated until the whole system is tested.

Bottom up approach: According to this approach, the lowest level modules are united to form clusters. Then a driver is made to get the input and output of the test case, and then the cluster is tested. Subsequently, the driver is removed, so that the cluster can be combined with the module or modules of the upper level. The process is repeated until the whole system is tested.

Within PARITY, even though the Bottom up is the main selected approach, a hybrid model has been followed, where both approaches are applied. In this way, integration tests of initial versions of the UIs can be made during the early stages.

Testing in the production environment itself can create various problems to the operation of already deployed software/equipment, disruption in databases, etc. Therefore, initial integration testing has been performed at the pre-pilots and the small-scale pilot deployments that are used as test environments, allowing to have full control and make any modifications without affecting the state of actual systems.

5.1 Pre-pilots

Successful integration tests that have been performed so far at the pre-pilots (CERTH & Hypertech premises), are described in the following tables.

Table 2. LEM/LFM Repository testing.

ID	Objective	Description	Expected result	Status
1	To check that the LEM/LFM Repository returns appropriate responses	LEM/LFM Repository returns the responses in JSON format. All formats must be valid, and response must indicate if the call succeeded or failed.	All responses are valid JSONs, and ‘failed’ status is returned if a check has failed (e.g. missing required input parameter)	PASS
2	To check that the LEM/LFM Repository returns correctly historical data	LEM/LFM Repository implements certain web services that return data found within a given time period that is passed as input (start-end datetime)	All stored events within the given time period are returned in the correct order	PASS

Table 3. LEM/LFM Management Application UI testing.

ID	Objective	Description	Expected result	Status
1	To check that the information on which are the active markets is displayed correctly	The LEM/LFM Management application UI retrieves grid status events from the LEM/LFM Repository. Based on the last event and the pre-defined rules, it updates the view about the active/suspended markets	The grid status events are read successfully and view about the active/suspended markets is updated accordingly	PASS
2	To check that information is stored correctly when registering users	The LEM/LFM Management application UI allows the operator to enter the details of a new user, such as a prosumer. This information is then stored to the LEM/LFM Repository	The resulting user JSON is valid, is correctly stored to the database, and can be retrieved on demand successfully	PASS

Table 4. Aggregator Toolset testing.

ID	Objective	Description	Expected result	Status
1	To check that the Aggregator Toolset UI presents successfully the information about prosumers and their assets	The Aggregator Toolset retrieves data about prosumers, assets, buildings, and measurements (among other) from the LEM/LFM Repository. This is performed through a set of call to the related web services	The Aggregator Toolset UI retrieves and displays all the data correctly	PASS
2	To check that the Aggregator Toolset UI is able to retrieve the properties of created clusters (dynamic VPPs) from VPP Manager	The VPP Manager of Aggregator Toolset implements a web service that provides on demand various properties of the dynamically created VPPs in JSON format. This web service is utilised by the UI for visualisation purpose	The JSON response is returned and is parsed correctly by the Aggregator Toolset UI	PASS
3	To check that the Aggregator Toolset UI is able to retrieve correctly the prosumer metrics from VPP Manager	The VPP Manager of Aggregator Toolset implements a web service that provides prosumers' metrics on demand. These metrics are calculated by the VPP Manager. The web service is utilised by the UI for visualisation purpose	The JSON response is returned and is parsed correctly by the Aggregator Toolset UI	PASS

5.2 Small-scale pilot deployments

Integration tests at small-scale pilot deployments are planned to be made. These tests will primarily focus on the LEM/LFM market operation and the main functionalities of the DSO Toolset and Aggregator Toolset. The objective is to discover and solve any flaws that may encountered in actual installations.

5.3 Full-scale pilot deployments

Full-scale integration tests at the pilot sites will be made before the start of the piloting phase and they will be reported in the second version of the deliverable.

5.4 End-user applications

Final integration tests of all user interface applications will be performed and reported in the second version of the deliverable.

6. CONCLUSIONS

This document provided a description on activities for system integration within the PARITY project. The PARITY components are under development at the time of writing this document; in most of the cases, the main functionalities have been implemented, and initial integration tests have been performed successfully at the pre-pilots in controlled environment.

An integration plan has been defined, and the integration status per each main component was presented. Furthermore, the integration activities that took place at the pre-pilot sites, which are located at the premises of Hypertech and CERTH, were described. Regarding system integration testing, the incremental integration testing approach has been selected, following a hybrid model that utilizes both the Bottom up and the Top down methods. Up to this point no major issues have been observed, however, still various interfaces have to be defined, implemented and tested.

System integration testing in a controlled environment will be continued in the next months in order to evaluate the integration for remaining functionalities that are under development, and to cover more complex cases such as the coordination of participation in the different types of markets at varying grid network states according to the traffic light approach.

The second version of the deliverable which is planned to be submitted by the end of the project, will present the final integration status and integration activities, as well as the results of all integration tests, showcasing the behaviour of the whole system at the different use cases.

7. REFERENCES

[1]. Rajabalinejad, M., van Dongen, L., & Ramtahalsing, M. (2020, January). Systems integration theory and fundamentals. In *Safety and Reliability* (Vol. 39, No. 1, pp. 83-113). Taylor & Francis.

[2]. ACN-Data dataset, https://ev.caltech.edu/dataset

[3]. SteVe - https://github.com/RWTH-i5-IDSG/steve

[4]. Introduction to System Integration Testing (SIT), https://www.educba.com/system-integration-testing/