The DELTA project has received funding from the EU's Horizon 2020 research and innovation programme under grant agreement No 773960

| | |
|---|---|
| Project Acronym: | **DELTA** |
| Project Full Title: | **Future tamper-proof Demand rEsponse framework through seLf-configured, self-opTimized and collAborative virtual distributed energy nodes** |
| Grant Agreement: | **773960** |
| Project Duration: | **36 months (01/05/2018 – 30/04/2021)** |

# DELIVERABLE D6.1

# DELTA Lab Testing, Evaluation and Test Suite Specification

| | |
|---|---|
| Work Package | **WP6 – Work package title DELTA Integration & Added-Value Services** |
| Tasks | **T6.1 – Planning and Integration of individual components and overall DELTA Framework**<br>**T6.2 - Lab Deployment, Configuration, Testing & Validation** |
| Document Status: | **Final** |
| File Name: | **DELTA_D6.1_Final** |
| Due Date: | **30 April 2020** |
| Submission Date: | **May 2020** |
| Lead Beneficiary: | **CERTH** |

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

# Authors List

| Leading Author | | | |
|---|---|---|---|
| **First Name** | **Last Name** | **Beneficiary** | **Contact e-mail** |
| Apostolos | Tsolakis | CERTH | tsolakis@iti.gr |

| Co-Author(s) | | | | |
|---|---|---|---|---|
| **#** | **First Name** | **Last Name** | **Beneficiary** | **Contact e-mail** |
| 1 | Christos | Patsonakis | CERTH | cpatsonakis@iti.gr |
| 2 | Kostas | Kostopoulos | CERTH | kostopoulos@iti.gr |
| 3 | Ioannis | Koskinas | CERTH | jkosk@iti.gr |
| 4 | Andrea | Cimmino | UPM | cimmino@fi.upm.es |
| 5 | Juan | Cano-Benitez | UPM | jcano@fi.upm.es |
| 6 | Raúl | García-Castro | UPM | rgarcia@fi.upm.es |
| 7 | Venizelos | Venizelou | UCY | venizelou.venizelos@ucy.ac.cy |
| 8 | Alexis | Frangoullides | UCY | frangoullides.alexis@ucy.ac.cy |
| 9 | Phivos | Theraponots | EAC | ptherapo@eac.com.cy |
| 10 | George | Spathoulas | NTNU | georgios.spathoulas@ntnu.no |
| 11 | Alessio | Baiocco | NTNU | alessio.baiocco@ntnu.no |
| 12 | Alexandre | Lucas | JRC | alexandre.lucas@ec.europa.eu |
| 13 | George | Karagiannopoulos | HIT | g.karagiannopoulos@hit-innovations.com |

# Reviewers List

| Reviewers | | | |
|---|---|---|---|
| **First Name** | **Last Name** | **Beneficiary** | **Contact e-mail** |
| | | HIT | |
| Venizelos | Venizelou | UCY | Venizelou.venizelos@ucy.ac.cy |

# Legal Disclaimer

The DELTA has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 773960. The sole responsibility for the content of this publication lies with the authors. It does not necessarily reflect the opinion of the Innovation and Networks Executive Agency (INEA) or the European Commission (EC). INEA or the EC are not responsible for any use that may be made of the information contained therein.

# Copyright

# Executive Summary

This report consists the Deliverable D6.1 “DELTA Lab Testing, Evaluation and Test Suite Specification” and documents the overall context for the evaluation and validation of the deployment of the DELTA individual and integrated components at lab environment to exemplify their usage at a pre-pilot level. The activities described in this deliverable are the results of both T6.1 “Planning and Integration of individual components and overall DELTA Framework” and T6.3 “T6.2 - Lab Deployment, Configuration, Testing & Validation”.

Although this report has only one version, activities within follow an iterative approach, and will be included in the second version of the DELTA integrated framework on M32. The current report, presents the overall evaluation methodology, the preliminary testing for all individual and integrated DELTA components (Sections 3 and 5 respectively) as well as future plans (Sections 4 and 6 respectively) as testing is an ongoing procedure that follows progressively development and deployment stages.

Following the requirement (D1.1/D1.5) and the architecture (D1.2/D1.6) this report establishes the testing methodology and delivers results, as these have been performed up to M24. Future testing activities will be included in D6.4 on M32 as part of the final integration report.

Furthermore, this report includes information regarding the deployment of the DELTA components, both individually and integrated versions, at the living lab infrastructure at the CERTH/ITI smart house. A description is also provided for the testing that will follow at the JRC testbed facilities.

This report signifies the importance of testing procedures as well as deployment and testing under real-life conditions before proceeding to the actual pilot cases. As demonstrated within its context most components are in a mature development status. There are components that require additional refinement before deployment to the pilots can be commenced, whereas others are already in a version that can adequately perform under real-life conditions.

Extended evaluation and validation of each component, as well as of the overall integrated DELTA framework, are expected in the following months, the plan of which is depicted in the respective sections.

# Table of Contents

1. Introduction ........ 8
1.1 Scope and objectives of the deliverable ........ 8
1.2 Structure of the deliverable ........ 8
1.3 Relation to Other Tasks and Deliverables ........ 9
2. Development of the Testing Methodology ........ 10
2.1 Hardware Testing ........ 10
2.2 Unit Testing ........ 11
2.3 Functional Testing ........ 11
2.4 Integration Testing ........ 11
2.5 System Testing ........ 12
2.6 DELTA Components for Experimental Evaluation & Validation ........ 12
3. Individual Component Testing – Preliminary Testing ........ 13
3.1 DELTA Customer ........ 13
3.1.1 Fog-enabled Intelligent Device ........ 13
3.2 DELTA Virtual Node ........ 16
3.2.1 Consumer/Prosumer Flexibility Data Monitoring and Profiling ........ 16
3.2.2 Generation/Consumption Optimal Dispatch ........ 17
3.2.3 Load Forecasting ........ 18
3.2.4 Inter/Intra Node Energy Matchmaking ........ 18
3.2.5 Consumer/Prosumer Energy/Social Clustering ........ 19
3.3 DELTA Aggregator ........ 20
3.3.1 Energy Market Price Forecast ........ 20
3.3.2 DR & Flexibility Forecasting ........ 21
3.3.3 Node Flexibility Data Monitoring and Profiling ........ 23
3.3.4 Asset Handling Optimization ........ 23
3.3.5 Self-Portfolio Energy Balancing ........ 24
3.3.6 Grid Stability Simulation Engine ........ 26
3.3.7 Energy Portfolio Segmentation & Classification ........ 37
3.4 Innovative Customer Engagement Tools ........ 38
3.4.1 DR Visualisation Kit ........ 38
3.4.2 Award –enabled Energy Behavioural Platform ........ 40
3.4.3 Social Interaction and Cooperation Platform ........ 41
3.5 Common Information Modelling ........ 41
3.6 Cybersecurity Services ........ 46
3.6.1 DELTA Blockchain ........ 46
3.6.2 Smart Contracts ........ 47
3.6.3 Threat Mitigation ........ 48
4. Individual Component Testing – Plan ........ 49
4.1 DELTA Customer ........ 49
4.1.1 Fog-enabled Intelligent Device ........ 49
4.2 DELTA Virtual Node ........ 50
4.2.1 Consumer/Prosumer Flexibility Data Monitoring and Profiling ........ 50
4.2.2 Generation/Consumption Optimal Dispatch ........ 51
4.2.3 Load Forecasting ........ 51

4.2.4 Inter/Intra Node Energy Matchmaking ........ 52
4.2.5 Consumer/Prosumer Energy/Social Clustering ........ 52
**4.3 DELTA Aggregator ........ 53**
4.3.1 Energy Market Price Forecast ........ 53
4.3.2 DR & Flexibility Forecasting ........ 54
4.3.3 Node Flexibility Data Monitoring and Profiling ........ 55
4.3.4 Asset Handling Optimization ........ 55
4.3.5 Self-Portfolio Energy Balancing ........ 56
4.3.6 Grid Stability Simulation Engine ........ 56
4.3.7 Energy Portfolio Segmentation & Classification ........ 56
**4.4 Innovative Customer Engagement Tools ........ 56**
4.4.1 DR Visualisation Kit ........ 56
4.4.2 Award –enabled Energy Behavioural Platform ........ 59
4.4.3 Social Interaction and Cooperation Platform ........ 59
**4.5 Common Information Modelling ........ 60**
**4.6 Cybersecurity Services ........ 61**
4.6.1 DELTA Blockchain ........ 61
4.6.2 Smart Contracts ........ 61
4.6.3 Threat Mitigation ........ 61

**5. Integration Testing – Preliminary Results ........ 63**

**5.1 DELTA Customer ........ 63**
**5.2 DELTA Virtual Node ........ 64**
**5.3 DELTA Aggregator ........ 67**
**5.4 Horizontal Services ........ 68**

**6. Integration Testing – Plan ........ 69**

**6.1 DELTA Customer ........ 69**
**6.2 DELTA Virtual Node ........ 69**
**6.3 DELTA Aggregator ........ 70**
**6.4 Horizontal Services ........ 71**

**7. System Testing at Lab environment ........ 73**

**7.1 Smart Home Testing Scenarios ........ 73**
**7.2 KIWI Testing Scenarios ........ 76**
**7.3 JRC TestBed Testing Scenarios ........ 77**

**8. Conclusions ........ 78**

**References ........ 79**

## List of Figures

**Figure 1: Agile methodology for iterative testing in DELTA ........ 10**

**Figure 2. Detailed model of the UCY campus power network. ........ 26**

**Figure 3. Illustration of the GSSE operation and inputs/outputs. ........ 27**

**Figure 4. Power Flow Analysis at the time of violation. ........ 28**

**Figure 5. Quasi-dynamic analysis – Feeders Loading. ........ 29**

**Figure 6. Quasi-Dynamic analysis – Feeder Active Power Flow. ........ 29**

**Figure 7. Power Flow Analysis at the time of violation. ........ 30**

**Figure 8. Quasi-dynamic analysis – Feeders Loading. ........ 31**

**Figure 9. Quasi-Dynamic analysis – Feeder Active Power Flow. ........ 32**

**Figure 10. Quasi-dynamic analysis – Overvoltage at Building/DVN busbars. ........ 33**

**Figure 11. Single Line Diagram of the violated LV Busbar. ........ 33**

**Figure 12. Quasi-dynamic analysis – Voltage restoration at Building/DVN busbars (overvoltage test) ........ 34**

**Figure 13. Quasi-dynamic analysis – Undervoltage at Building/DVN busbars. ........ 35**

**Figure 14. Quasi-dynamic analysis – Voltage restoration at Building/DVN busbars (undervoltage test). ........ 36**

**Figure 15 – CIM Semantic Interoperability overview ........ 42**

**Figure 16: CERTH/ITI Smart House Testbed ........ 73**

**Figure 17: Two FEID v1 (left) have been deployed to monitor lab “consumers” and one FEID v2 (right) has been deployed to monitor a lab “prosumer” ........ 73**

**Figure 19: FEID v2 at KIWI for testing the Frequency Support functionality and integration with the KIWI fruit. ........ 77**

## List of Acronyms and Abbreviations

| Term | Description |
|---|---|
| DER | Distributed Energy Resource |
| AHO | Asset Handling Optimization |
| SPEB | Self-portfolio Energy Balancing |
| GSSE | Grid Stability Simulation Engine |
| FEID | Fog-enabled Intelligent Device |
| CIM | Common Information Modelling |

# 1. Introduction

## 1.1 Scope and objectives of the deliverable

The purpose of this deliverable is to give a systematic methodology, results and a time schedule of the evaluation framework within DELTA. The overall activities are guided by the business scenarios and the technical use cases as analysed at D1.5 “DELTA Requirements, Business Scenarios and Use Cases v2” as well as the architectural interrelations and functional and non-functional requirements in D1.6 “DELTA Overall Framework Architecture v2”. This methodology and iterative procedure aims to ensure the compliance of the DELTA integrated framework with the DELTA vision.

The evaluation activities performed up to M24 in this deliverable reflects the work performed in Task T6.1 – “Planning and Integration of individual components and overall DELTA Framework” and T6.2 – “Lab Deployment, Configuration, Testing & Validation”.

Furthermore, beyond the individual and integrated testing performed up to M24 and planned for the remaining period, the lab deployment of the various DELTA components, as well as the DELTA framework at the Living Lab facilities in CERTH/ITI and JRC are elaborated.

As will be demonstrated by the methodology followed, evaluation and testing activities were continuously updated and refined through an iterative process that lead to the production of multiple software and hardware releases. As this process will continue, and actually intensify in the following moths, any further activities will be documented in D6.4 on M32.

## 1.2 Structure of the deliverable

The document is structured as follows:

- Section 2 provides an overview of the evaluation / testing methodology;
- Section 3 presents testing results per individual component as have been performed up to M24;
- Section 4 introduces the individual component testing plans for the next period;
- Section 5 presents the integrated DELTA framework testing results as have been performed up to M24;
- Section 6 introduces the integrated DELTA framework testing plans for the next period;
- Section 7 provides information regarding deployment, evaluation and validation on the project pre-pilot testing facilities, and
- Section 8 concludes the report

## 1.3 Relation to Other Tasks and Deliverables

This report is directly linked with all technical activities of WP3, WP4, WP5 and WP6 that undertake development and integration of DELTA components. Finally, the evaluation of both individual and integrated components is based on the architecture and requirements defined in WP1, aiming to deliver the business objectives of WP2.

# 2. Development of the Testing Methodology

Iterative testing is an important process in any software and hardware implementation process. After the development of a component it is imperative to test whether i) the initial implementation is robust, ii) the individual requirements for the specific component have been met, iii) how this component functions when integrated with other components, and finally iv) how the entire integrated system operates given the predefined business scenarios and technical use cases. Each of these tests is executed iteratively after a development process has reached a certain maturity level and there has been a stable version provided. In DELTA, for effectively providing viable solutions, the agile methodology has been followed for running the iterative process described. An indicative visual representation of the overall process is depicted in the following figure:

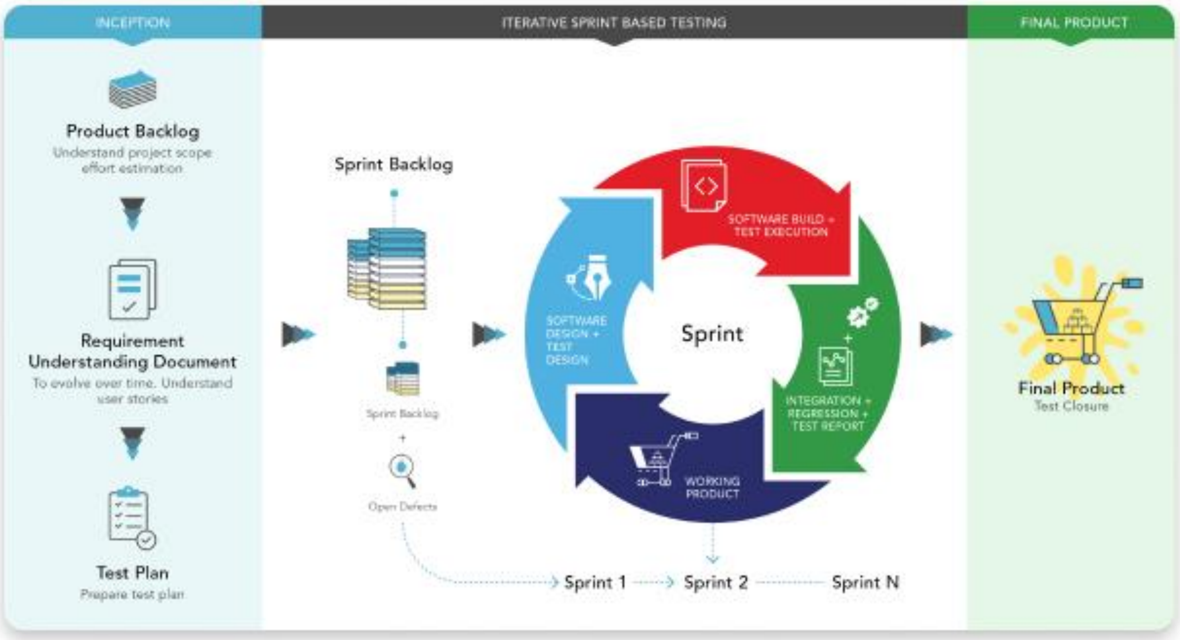


**Figure 1: Agile methodology for iterative testing in DELTA[1]**

Within DELTA, various types of tests have been foreseen (where and when applicable) to be executed to cover the above testing requirements, prior to the pilot execution. The following sections describe the testing levels used within DELTA. As both internal and external attributes of each component have been evaluated, the overall process follows the "grey-box" testing where in some cases the components are examined as completely transparent entities ("white-box testing"), whereas in other cases their overall functionality is tested as if nothing was known for the interior structure ("black-box" testing).

## 2.1 Hardware Testing

Within the core components of DELTA is the Fog-Enabled Intelligent Device (FEID). Besides software, DELTA delivers the hardware as well. As such, the various tests that have been performed during the manufacturing of this new hardware device are elaborated.

[1] http://www.qalab.co/agile-testing-process.html

## 2.2 Unit Testing

This level of testing aims to evaluate the core building blocks of a software application. This type of testing is typically executed by the developers, and involves the testing of individual classes, or small clusters of classes (a package). Its main purpose is to ensure high quality in the design and implementation of classes, checking that these behave as expected and identifying “bugs” prior to integrating these pieces of the code (packages) into the rest of the system. Early identification of “bugs” is significantly more cost-effective than in later stages, especially for commercial and industrial environments, while it also ensures that the delivered component will be stable and resource-wise efficient under normal operation. Some of the most common metrics examined during unit testing are: test/code coverage, cyclomatic complexity, code duplications, rules compliance, comment coverage, as well as other code related statistics.

Most languages have their own unit testing frameworks (i.e. pytest, junit, etc.), but there is also other third-party software that can provide such testing capabilities (e.g. Jenkins, SonarQube, Spock, etc.). The right tool will be chosen by the test team during the test plan preparation, based on testing needs per particular feature. For some DELTA components it may not be possible to apply unit testing (e.g. Grid Stability Simulation Engine), as their core development is based on other commercial software, which in some cases is a “black-box”. For these components, only functional tests are executed.

## 2.3 Functional Testing

The main objective of this test is to verify that the component behaves according to the related functional technical requirements that were created during the design process. The component under test is examined as an individual module, as if it was a “black-box”, towards evaluating its expected functionalities. A successful functional test enables the integration of the module in the system.

The functional tests are not based on a specific test suite, but rather on ad-hoc test cases focusing on the main functionalities and behaviour of the component under test. These are defined from the technical requirements (D1.5) and the architecture (D1.6) delivered earlier in the projects’ lifecycle (functional design specifications), and towards successfully delivering the business scenarios expected. As such, for each component a list of test cases has been identified and it is partially already executed towards assessing step by step the expected functionalities, along with limitations, performance issues, and other related metrics that can ensure the proper functional behaviour.

## 2.4 Integration Testing

This test level aims to ensure that the components can integrate among each other effectively and as designed within a proper environment. Communication and functional compatibility is expected and therefore tested. As the proper environment for each component is defined by the integration with other components, these test cases are limited per each of the DELTA layers, namely the DELTA customer (integration with devices, assets, building management systems, as well as internal software components, etc.) DELTA Virtual Node, and the DELTA aggregator as complete components. Beyond this, their in-between communication, and specifically their semantic interoperability, is tested as well.

Again there isn’t a specific framework to execute these tests, but certain methodologies have been followed based on the needs of each layer. For example the Smart Grid Architecture Model [1] has been followed for defining the semantic interoperability tests and following accordingly. The integration tests mainly cover test cases the aim to evaluate each integrated system’s behavior in terms of execution, stability and reliability.

## 2.5 System Testing

This testing level corresponds to the DELTA framework as a whole. Hence, the test cases are as close as possible to the business scenarios and their objective is the verification of correct integration and cooperation of all software components including the hardware interfaces. The overall system testing has been performed at the two lab environments provided within DELTA: a) the JRC test bed and b) the CERTH/ITI Smart home. In each lab environment, specific tests have been executed towards validating the up to date DELTA framework.

## 2.6 DELTA Components for Experimental Evaluation & Validation

The DELTA project includes an extended list of components that have been deployed at lab environment and has been tested extensively towards presenting the overall DELTA framework. As depicted both in architecture (D1.2/D1.5) and integration (D6.3) deliverables, these are:

| |
|---|
| DELTA Customer |
| Fog-Enabled Intelligent Device (Hardware/Software) |
| DELTA Virtual Node |
| Consumer/Prosumer Flexibility Data Monitoring and Profiling |
| Generation/Consumption Optimal Dispatch |
| Load Forecasting |
| Consumer/Prosumer Energy/Social Clustering |
| Inter/Intra Node Energy Matchmaking |
| DELTA Aggregator |
| Energy Market Price Forecasting |
| DR & Flexibility Forecasting |
| Node Flexibility Data Monitoring and Profiling |
| Asset Handling Optimisation |
| Self-Portfolio Energy Balancing |
| DELTA Grid State Simulation - Grid Stability Simulation Engine* |
| Energy Portfolio Segmentation & Classification |
| Common Information Model |
| Added Value Services |
| DR Visualisation Kit |
| Award-enabled Collaboration Platform |
| Cyber Security Services |
| DELTA blockchain |
| Smart Contracts & Gateway |
| Threat Mitigation Services |

*Although it has been highlighted as a separate component in the updated architecture in D1.6 it remains at the Aggregator layer level.

# 3. Individual Component Testing – Preliminary Testing

## 3.1 DELTA Customer

### *3.1.1 Fog-enabled Intelligent Device*

#### *3.1.1.1 Hardware Testing*

Several test points are manufactured at the PCB of the FEID. These test points allow the attachment of measurement equipment to monitor voltage and current at critical subsystems of the board as well as the main system power. In addition, there is a red LED to indicate that the board is powered on. Upon receiving the populated PCB from the manufacturer, the first test is to check if every subsystem is being supplied with the required voltage level.

After the initial setup and boot of the FEID in which a green LED is blinking, the peripherals of the device must be tested. Ethernet interface is plugged into a test local area network and the embedded LEDs at the connector are checked for connectivity and link budget. An online file is downloaded to check internet connectivity. Wi-Fi / BLE communication module has test point in which a debugger can directly connect and test. As with the Ethernet, the board connects to a Wi-Fi access point and downloads an online file to check internet connectivity.

The remaining interfaces SPI, UART, I2C, RS-232, RS-485, are tested by attaching a dummy device with embedded communication LEDs that blink on receiving a protocol packet, after running an automated test script. Lastly, a test load is connected to the two relays which are controlled by a script

#### *3.1.1.2 Unit Testing*

No unit testing has been performed yet.

#### *3.1.1.3 Functional Testing*

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Load Forecasting Execution Performance | Evaluate execution performance under various conditions | • Low execution time (under 3’) | Pass |
| | | | • Correct data results for the entire timeframe requested | Pass |
| 2 | Load Forecasting Accuracy Performance | Evaluate accuracy in regards to real-time measurements under various conditions | Accuracy under weekday, weekend, and other operational scenarios (errors less than 15%) | Partial Pass. There are still conditions where the error is above 15% without considered an outlier. |
| 3 | PV Forecasting Execution Performance | Evaluate execution performance under clear sky and cloud conditions | • Low execution time (under 3’)<br>• Correct data results for the entire timeframe requested | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 4 | PV Forecasting Accuracy Performance | Evaluate accuracy in regards to real-time measurements under clear sky and cloud conditions | Accuracy under clear sky and cloudy days (clear sky error less than 10%, cloudy days less than 15%) | Pass for clear sky days<br><br>Partial Pass for cloudy days. Further testing and refinement is required. |
| 5 | Flexibility Forecasting Execution Performance | Evaluate performance and accuracy of flexibility forecasting under various conditions | • Low execution time (under 3')<br>• Correct data results for the entire timeframe requested | Pass |
| 6 | Flexibility Forecasting Accuracy Performance | Evaluate accuracy in regards to real-time conditions under various scenarios | Accurate (in the context of same order of magnitude and relative value) estimation. +/-15% From actual available flexibility | Partial Pass. In certain conditions flexibility extracted was beyond accepted limits. Further testing is required. |
| 7 | Local Database | All the collect energy related measurements and predicted values should be stored locally in time-series database | • Data are stored in specific time intervals<br>• Data are stored in specific format<br>• Retention policy<br>• Only 3 months of data are kept | Pass |
| 8 | Customer User Interface Testing | FEID should support a user interface where customer can be informed about their infrastructure | • Friendly interface<br>• Multiple dashboards | Pass - The User interface provide a very friendly environment where the customer can have full access to it's infrastructure |
| | | | • Access to Historical Information (3 months) | Pass |
| | | | • Monitoring and control capabilities | Pass |
| | | | • Robust Communication | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 9 | Installer User Interface Testing | FEID should support a mobile user interface that facilitates the installation and configuration of the device at customers' premises. Testing of communication and functionalities | • Friendly interface<br>• Multiple and easy to operate dashboards | Pass - The Installer User interface provide a very friendly environment where the installer can easily navigate to install and configure the FEID |
| | | | • Communication with FEID through Mobile Device (Smart Phone / Tablet) | Pass |
| | | | • Addition of new assets | Pass |
| | | | • Update of customer's preferences | Pass |
| | | | • Registration of new FEID to the DELTA network/portfolio | Pass |
| 10 | Weather Forecasting Data Acquisition | Get from an online API the Weather forecast for the day ahead | Correct data collection especially for those that are required for the PV forecasting | Pass |
| 11 | Electricity Price Forecasting Data Acquisition | Get from an online API the Electricity Price predicted values for the day ahead | Correct data collection | Pass |
| 12 | Set up WiFi access point | At installation phase FEID must set up a WiFi access point in order other mobile devices could connect with it | WiFi access point with preferable Name and Security keys | Pass |

## 3.2 DELTA Virtual Node

### *3.2.1 Consumer/Prosumer Flexibility Data Monitoring and Profiling*

#### *3.2.1.1 Unit Testing*

No unit testing has been performed yet.

#### *3.2.1.2 Functional Testing*

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Store Historical Consumption | Store FEID Historical Consumption data | FEID Historical Consumption data are stored to DVN's database | Pass |
| 2 | Store Historical Generation | Store FEID Historical Generation data | FEID Historical Generation data are stored to DVN's database | Pass |
| 3 | Store Voltage & Frequency | Store FEID Voltage & Frequency | FEID Voltage & Frequency are stored to DVN's database | Pass |
| 4 | Store Flexibility Forecast | Store FEID's Forecasted Flexibility | FEID's Forecasted Flexibility data are stored to DVN's database | Pass |
| 5 | Provide Node Profiling | Node Profiling is exposed according to DELTA data model | Node Profiles are provided from DVN | Pass |
| 6 | Ensure that flexibility of distributed assets can be aggregated as a single unit to sell services | Constantly monitor the portfolio's composition and capabilities in terms of stability and flexibility | Single control requests communicate appropriately | Pass |
| 7 | Allow Aggregator to supervise each node's flexibility and contextual data | Provide real-time overview of the assets assigned to a specific DVN | Produce node profiling for each node that follows the DELTA data model specification | Pass |
| 8 | Provide real-time automated monitoring and control of buildings | Analyzes the FEIDs profiling of the underneath DELTA Fog Enabled Agent | Coordinated management of a building's assets in an energy efficient manner | Pass |

### 3.2.2 Generation/Consumption Optimal Dispatch

#### 3.2.2.1 Unit Testing

No unit testing has been performed yet.

#### 3.2.2.2 Functional Testing

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Compute optimal DR Signals | Calculate optimal DR signals to fulfil energy demands | Generate optimal DR signals | Pass |
| 2 | Generate Blockchain Transactions | After creating optimal DR Signals the relevant Transactions for DELTA Blockchain should be created | Generate Transactions to DELTA Blockchain | Pass |
| 3 | Handle unresolvable demands | When no solution can be found respond accordingly | Respond with inability to find optimal solution | Pass |
| 4 | Establish the optimal DR signals to be sent to the DELTA Fog Enabled Agent must fulfill | Compute the DR signals that should be sent to the DELTA Fog Enabled Agents | DR signals sent to the DELTA Fog Enabled Agent should be translated from the DR signal received form the DELTA aggregator | Pass |
| 5 | Faulty Input testing | The Optimal Dispatch Tool needs multiple input, a fact that creates dependencies with other DELTA modules. In case any of these is faulty, then the Optimal Dispatch Tool will not be able to calculate the optimal scheduling. | Potential faulty input timeseries should be successfully identified as such, proper logging should be executed and smooth termination of the Optimal Dispatch Tool. | Pass |
| 6 | DVN power balance | DVN FEIDs should at all timeslots of a DR signal satisfy the power constraint. | Power Balance is checked and verified for every optimal solution. | Pass |
| 7 | Test logging and return of Optimal Dispatch Tool | All possible output scenarios (optimal, infeasible, error in formation, error at input cases) should be foreseen and not cause a tool break | No tool collapse under any circumstances regarding tool configuration and DR signal. | Pass |

### 3.2.3 Load Forecasting

#### 3.2.3.1 Unit Testing

No unit testing has been performed yet.

#### 3.2.3.2 Functional Testing

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Handle lack of data | Detect if Node Profiling contains inadequate data to generate Forecast | Given an empty Node Profile or a Node Profile with inadequate data Load Forecasting returns an explanatory message | Pass |
| 2 | Load Forecasting Execution Performance | Evaluate execution performance under various conditions | • Low execution time (under 3') | Pass |
| | | | • Correct data results for the entire timeframe requested | Pass |
| 3 | Load Forecasting Accuracy Performance | Evaluate accuracy in regards to real-time measurements under various conditions | Accuracy under weekday, weekend, and other operational scenarios (errors less than 15%) | Partial Pass. There are still conditions where the error is above 15% without considered an outlier. |

### 3.2.4 Inter/Intra Node Energy Matchmaking

#### 3.2.4.1 Unit Testing

No unit testing has been performed yet.

#### 3.2.4.2 Functional Testing

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Dynamically update DVN | Automatically reassign a customer to another cluster/Node when one of the parameters changes | The DVN should have uniform characteristics among the customers | Pass |
| 2 | Control the balance of energy or stability inside the Node | Facilitate the self-balancing process, so as to prevent the loss of energy or stability within the portfolio | Ensure balance of energy or stability within the portfolio | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 3 | Provide accurate and close to real-time evaluation inside the Node | Accumulate and evaluate in close to real-time the excess or shortage of energy inside the Node's portfolio | Achieve close to real-time control inside the Node | Pass |
| 4 | Provide effective collaboration among the Nodes | Request/offer energy from adjacent Nodes when intra-Node energy matchmaking is not possible | Achieve coordination among the Nodes | Pass |
| 5 | Allow communication with Aggregator | Send an "insufficient resources" signal to the Aggregator in case of not sustained balance | Ensure information transmission for the state of the Node | Pass |

### *3.2.5 Consumer/Prosumer Energy/Social Clustering*

#### *3.2.5.1 Unit Testing*

Unit Testing Procedure applied over the Pytest module in order to evaluate the Consumer/Prosumer Energy/Social Clustering module. The basic test components focused on testing the eligibility of the following conditions: Clustering Results Format, Exploitation of all resources, Proper communication and connection with the DVN's assets. Testing Procedure applied over several random inputs in order to guarantee the proper functionality of Clustering Engine under any circumstances.

#### *3.2.5.2 Functional Testing*

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Results' Format | Test that the Clustering Module output structure has the appropriate format | Compare the structure of the output with the desired result. | Pass |
| 2 | Exploitation of all Resources | All DVN's assets have to take part in the Clustering Process | Examine the condition that all available assets participate in the clustering algorithm | Pass |
| 3 | Clustering Constraints Satisfaction | Evaluate the Constraints' Satisfaction of the Clustering process | Examine if all DELTA constraints are satisfied through the clustering results | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 4 | Communication with other DVN functionalities | Test the connection with other DVN's modules | Examine the condition that all the DVN's assets have access to Clustering Results | Pass |

## 3.3 DELTA Aggregator

### *3.3.1 Energy Market Price Forecast*

#### *3.3.1.1 Unit Testing*

The unit testing process was addressed using Pytest for Jypyter notebook and the NBextensions tools. Two stages of testing were performed. The first, tests that the scrapping of data is well performed by basically checking if the columns acquired match the desired ones. These are the parameters used in the model. The second, tests the algorithm, how the model for the price forecast performs. For this component the Elexon balancing energy market was used (www.bmreports.com)

**Parameters:**

- Scrapping LoLP and Derated Margin variables:

Assert all(df3_result.columns==['Date','Settlement Period','12h LoLP','12h DRM','8h LoLP','8h DRM','4h LoLP','4h DRM','2h LoLP','2h DRM','1h LoLP','1h DRM']) = Passed. Processing Time: 7.181s

- Scrapping Wind and Solar Generation

Assert all(df6.columns==['PSR Type', 'Settlement Date', 'Settlement Period', 'Day Ahead (MW)', 'Intraday (MW)', 'Current (MW)']) = Passed. Processing Time: 11.013s

- Scrapping System Demand and Base Generation (without Solar and Wind)

Assert all(df4.columns ['Settlement Date', 'SP', 'NDF Publish Time (GMT)', 'NDF (MW)', 'TSDF Publish Time (GMT)', 'TSDF (MW)', 'INDDEM Publish Time (GMT)', 'INDDEM (MW)', 'INDGEN Publish Time (GMT)', 'INDGEN (MW)']) = Passed. Processing Time: 2.252s

Assert all(df5.columns ['Time Series ID', 'Settlement Date', 'Settlement Period', 'Quantity (MW)']) = Passed. Processing Time: 1.747s

**Algorithm:**

Assert: Training of the model 80% of the data = Passed. Processing Time:2.717s
Assert : Testing the model with 20% of the data = Passed. Processing Time: 9.15s

Assert: Running the model with real time data: 3.236s (of which 154ms is the prediction)

#### 3.3.1.2 Functional Testing

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Check if the script runs with missing data | We have ran the script exposing it to missing values as is the case o LoLP of short time forecast (ex:1h) or zero. | Run/Does not run If missing than it replaces by zero | The script runs with zero values and missing data. Accuracy of the output will be lower |
| 1a | Wind and Solar Generation missing data and production and demand | No check for missing data was possible since the wind/solar generation forecasts are published all at once. The same happens for base generation and demand. In any case if missing data exists the model will assume as zero | Runs or does not run | Runs |
| 2 | Check model performance in terms of speed – Algorithm XGBoost | Model implementation with historic dataset. | Measures the time it takes to arrange data, train model and test. | Data formatting= 819ms Train=2.717s Test=9.15s Total=12,686s |
| 3 | Metrics of the model | Metrics used: R^2 score, Mean absolute error mean_squared_error explained_variance_score CrossValidation accuracy (CV=10) | These are the main metrics used for regression models. They take in test and predicted target variables | R^2=0.83 MAE=5.73 MSE=91.85 EVS=0.83 CV_Accuracy= 0.72 (+/- 0.12) |
| 4 | Real Data Prediction Cycle | The model predicts a full day balance energy market prices for each settlement period (48 outputs). It scrapes the data directly from the Market operator and runs the regression code | Measures the time it takes to retrieve the data from the web and predict the 48 settlement periods | 3.236s |

### 3.3.2 DR & Flexibility Forecasting

#### 3.3.2.1 Unit Testing

The core of the calculation is to apply a decision table to estimate the flexibility of appliances. A categorization was done dividing appliances between shiftable and variable, variable but not shiftable and shiftable but not variable. The load was forecasted using a non-intrusive load monitoring tool. If was observed that the accuracy was very low but this was because independent variables such as the weather/ temperature were missing for the regression. However the focus of the study is to apply a potential flexibility given a comfort limit of 95% for users. For the training of the load forecast model 2 datasets were used. The Refit dataset with 10 million observations was used, as it was recorded in a 1 second time step referring to four months. The split was performed at 75% and 25% between training and test. The training times for the Refit dataset were 39.20 and 9.22 seconds, corresponding to the Factorial Hidden Markov Model (FHMM) and Combinatorial Optimization (CO) algorithms.

Regarding the REDD dataset, all observations corresponded to 36 days for building 1 and were all taken into consideration to run the model. This corresponds to 3.1 million observations, since also a 1-second time step was used to record the data. The training times for the REDD dataset were 15.18 and 1.03 seconds, corresponding to the FHMM and CO algorithms.

#### *3.3.2.2 Functional Testing*

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Load forecast for a given appliance | A fridge was taken into consideration to estimate the flexibility of a fridge given a certain load profile forecasted by a non-intrusive load monitoring tool | Accuracy of the NILM approach F1 Score Metric | CO F1 score= 0.55<br>FHMM= 0.49<br><br>Very low. Other independent variables required to increase the accuracy, such as temperature, weather etc… |
| 2 | Flexibility Prediction with the load forecast for a given appliance | Assert: def application of the flexibility | Assuming a linear behavior in all appliances in terms of power and time decrease. Meaning that 50% of AC power reduction could be sustained for 16 minutes, 25% during for 32 minutes, and so on. Also that 95% of comfort of users would be maintained | Flexibility applied for 2h: Pool pump=100%; space heating=50%, Heat pump=50%; water heating=81.25%; AC=6.67%; Refrigerator=56.25%; Freezer=56.25%; Lighting=10%<br><br>Passed<br>Test Data: 2.02 s |
| 3 | Higher accuracy and higher processing speed were required. So only 8 appliances were used | K=8 higher contributors of power (appliances) | Speed and Power | Results show a flexibility maximum power of 200–245 W and 180–500 W for the REDD and Refit datasets respectively. |

### 3.3.3 Node Flexibility Data Monitoring and Profiling

#### 3.3.3.1 Unit Testing

This component is part of the DSS and has been developed in the same coding package as the Asset Handling Optimization. As such the unit testing has been performed in the combined version. Beyond that, functional testing has been performed during development as follows.

#### 3.3.3.2 Functional Testing

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Data integrity | Evaluate whether the data send by the FEIDs, and requested from the Aggregator/DVNs are correct and as expected | 100% Data Integrity | Pass |
| 2 | Update FEID profile in DVN repositories | Evaluate the correct profiling of customers based on data derived from FEIDs | Update key characteristics in regards to incoming data | Pass |

### 3.3.4 Asset Handling Optimization

#### 3.3.4.1 Unit Testing

Unit Testing Procedure applied over the Pytest module in order to evaluate the Asset Handling Optimization module functionalities. The basic test components focused on testing the eligibility of the following conditions: Results Format, Results Content, Time Processing Constraints and successful communication with other Components. Testing Procedure applied over several random inputs in order to guarantee that the AHO engine is not susceptible under any circumstances.

#### 3.3.4.2 Functional Testing

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Test Results Format | Evaluate the condition that the responses from AHO have the appropriate format. | Examine if the AHO responses' structure suits with the Delta result format. | Pass |
| 2 | Test Results Content | Evaluate the condition that the responses from AHO have all the information needed for a functional DR | Examine if the AHO responses' content contains all the demanded information. | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 3 | Test Communication with other components | Evaluate the condition that the AHO module communicates with other Components without any faults. | Examine the interaction with all cooperative components and the communication responses | Pass |
| 4 | Test time Limits | Evaluate the condition that the AHO module can process all the information in reasonable time limits | Examine the that the Processing time of AHO module does not overpass specific time limits | Pass |

### *3.3.5 Self-Portfolio Energy Balancing*

#### *3.3.5.1 Unit Testing*

Self – Portfolio Energy Balancing (SPEB) component as part of the DELTA Aggregator/Energy Retailer layer, evaluates the DVNs' portfolios based on several criteria to optimize the bidding strategies of the Aggregator. The component is developed in Python and it is divided into two functions:

- Identification of the optimal combination of DVNs based on the criteria of availability, profitability, reliability, flexibility and fairness
- Update of the Reliability and Fairness Indices

The two functions exchange data with the "DR & Flexibility Forecasting" and "Asset Handling Optimization" components through the common Aggregator/Energy Retailer layer as well as with the DELTA Repository through the DELTA CIM.

#### *3.3.5.2 Functional Testing*

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Optimal Combination of DVNs | Combinations of all available DVNs that can participate in the upcoming DR request (flexibility and market) are prioritized | Identify and prioritize all derived combinations based on the most profitable, fair and reliable combination of DVNs that are available and can meet the total requested flexibility | 08/04/2020 |

Each DVN is represented by the available flexibility (either static or range) that can serve specific energy markets and the compensation price of those services with the respective penalty prices, as derived from the smart contracts.

Based on historical participations, each DVN is characterized by both a Reliability and Fairness Index. The following table summarizes all buildings/DVNs located within the UCY campus with the respective values of price, reliability and fairness indices.

| DVNs | Markets | Flexibility [kWh] | Flexibility Prices [€/kWh] | Penalty Prices [€/kWh] | Reliability Index [%] | Fairness Index [%] |
|---|---|---|---|---|---|---|
| 121 | Day-Ahead, Imbalance | [1-5], 7 | 0.065, 0.07 | 0.01083, 0.0116 | 0.81 | 0.1818181818, 0.0819672131 |
| 122 | Imbalance | [2,4,6] | 0.08 | 0.0133 | 0.52 | 0.081967213 |
| 123 | Day-Ahead | 8 | 0.095 | 0.01583 | 0.68 | 0.109090909 |
| 124 | Day-Ahead, Imbalance | [2-4], 9 | 0.074, 0.080 | 0.0123, 0.0133 | 0.68 | 0.0545454545, 0.1147540984 |
| 125 | Imbalance | 3 | 0.102 | 0.017 | 0.66 | 0.06557377 |
| 126 | Day-Ahead | 3 | 0.0735 | 0.01225 | 0.77 | 0.072727273 |
| 127 | Day-Ahead, Imbalance | 9, [2,4] | 0.100, 0.150 | 0.0166, 0.025 | 0.7 | 0.3454545455, 0.262295082 |
| 111 | Imbalance | [2,4] | 0.070 | 0.0116 | 0.85 | 0.180327869 |
| 112 | Day-Ahead | 1 | 0.085 | 0.01416 | 0.3 | 0.036363636 |
| 113 | Imbalance | 4 | 0.101 | 0.01683 | 0.9 | 0.016393443 |
| 114 | Day-Ahead, Imbalance | 1, 1 | 0.075, 0.085 | 0.0125, 0.0142 | 0.4 | 0.0727272727, 0.0655737705 |
| 115 | Day-Ahead | 12 | 0.0852 | 0.0142 | 0.55 | 0.127272727 |
| 116 | Imbalance | 11 | 0.1050 | 0.0175 | 0.6 | 0.131147541 |

The upcoming DR signal, received from the "DR & Flexibility Forecasting" component, provisions a flexibility volume equal to 6 kW for the period of one 1 hour assigned for the Day-Ahead market.

The following table shows all the possible combinations of available DVNs, that can meet the requested flexibility, along with the total revenue and their fairness metrics. The table also indicates which combinations are eligible to participate in the upcoming DR signal (Fair or Unfair).

| ID | Combination of DVNs | Flexibility per DVN | Total Revenue (including reliability) | Combination Fairness Index | Combination Fairness Weight | Fair? |
|---|---|---|---|---|---|---|
| 1 | '121', '126', '112', '114' | [1, 3, 1, 1] | 26.4214 | 0.36363 | 0.30769 | UNFAIR |
| 2 | '121', '124', '112', '114' | [1, 3, 1, 1] | 26.0253 | 0.34545 | 0.30769 | UNFAIR |
| 3 | '121', '124', '126' | [1, 2, 3] | 21.3529 | 0.30909 | 0.23076 | UNFAIR |
| 4 | '121', '124', '114' | [1, 4, 1] | 21.2212 | 0.30909 | 0.23076 | UNFAIR |
| 5 | '121', '124', '112' | [1, 4, 1] | 19.2906 | 0.27272 | 0.23076 | UNFAIR |
| 6 | '121', '126', '114' | [2, 3, 1] | 15.9917 | 0.32727 | 0.23076 | UNFAIR |
| 7 | '124', '126', '114' | [2, 3, 1] | 14.4099 | 0.20001 | 0.23076 | FAIR |
| 8 | '121', '126', '112' | [2, 3, 1] | 14.061 | 0.29090 | 0.23076 | UNFAIR |
| 9 | '121', '112', '114' | [4, 1, 1] | 13.1385 | 0.29091 | 0.23076 | UNFAIR |
| 10 | '124', '126', '112' | [2, 3, 1] | 12.4793 | 0.16363 | 0.23076 | FAIR |
| 11 | '124', '112', '114' | [4, 1, 1] | 12.3476 | 0.16363 | 0.23076 | FAIR |
| 12 | '121', '124' | [2, 4] | 8.8608 | 0.23636 | 0.15384 | UNFAIR |
| 13 | '121', '114' | [5, 1] | 8.4997 | 0.25454 | 0.15384 | UNFAIR |
| 14 | '121', '126' | [3, 3] | 7.9209 | 0.25454 | 0.15384 | UNFAIR |
| 15 | '124', '126' | [3, 3] | 6.8664 | 0.12727 | 0.15384 | FAIR |
| 16 | '121', '112' | [5, 1] | 6.5691 | 0.21818 | 0.15384 | UNFAIR |

Although, the first combination yields the highest revenue for the Aggregator, the results of the SPEB component reject the option based on the "Fairness" criterion. Instead, SPEB identifies the 7$^{th}$ combination ('124', '126', '114') as the most profitable solution where both reliability and fairness criteria are met.

Assuming that all DVNs, which were selected in the participating combination, fulfilled their contracted flexibility obligations, then the SPEB component updates the Reliability and Fairness Indices as follows:

| DVNs | Reliability Index [%] Before | Reliability Index [%] After | Fairness Index [%] Before | Fairness Index [%] After |
|---|---|---|---|---|
| 124 | 0.68 | 0.713 | 0.0545454545 | 0.06557377 |
| 126 | 0.77 | 0.82 | 0.072727273 | 0.08653455 |
| 114 | 0.4 | 0.417 | 0.0727272727 | 0.08642354 |

### *3.3.6 DELTA Grid State Simulation - Grid Stability Simulation Engine*

#### *3.3.6.1 Unit Testing*

The development of the Grid Stability Simulation Engine (GSSE) component involves integration between Python and DIgSILENT PowerFactory. As an input the engine will receive the forecasted and real time power data, through JSON format, which will be fed to DIgSILENT and assigned to the respective grid components. Through DIgSILENT, the GSEE performs a Quasi Dynamic Analysis on the developed electrical/geographical (accurate representation of electrical and geographical parameters of lines and loads) model of the investigated power network to identify grid violations.

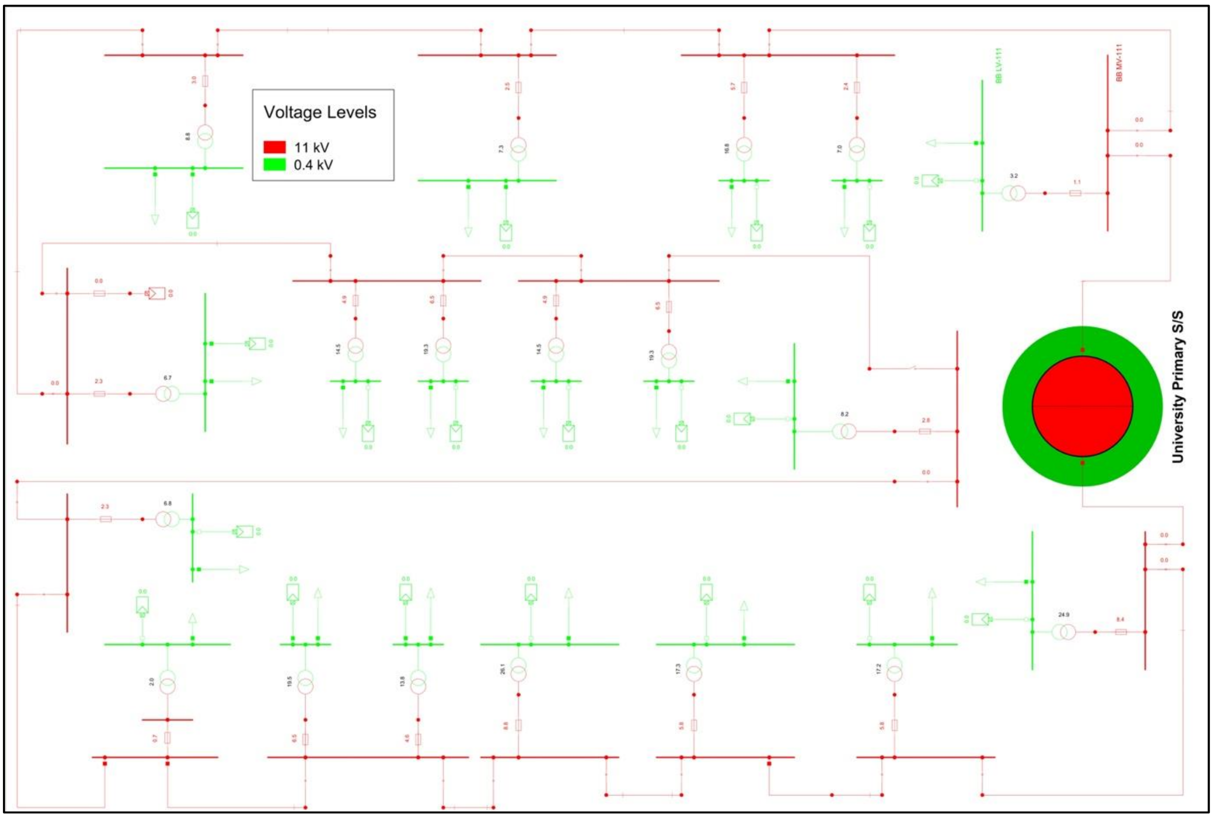


**Figure 2. Detailed model of the UCY campus power network.**

The Python script was developed to establish real time and automatic control capabilities over DIgSILENT, which is a third-party licensed software. To this end, target areas in the investigated power network can be simulated, thus enabling identification of potential grid violations (type, location, and time) and restoring conditions based on predefined constraints set by national Grid rules.

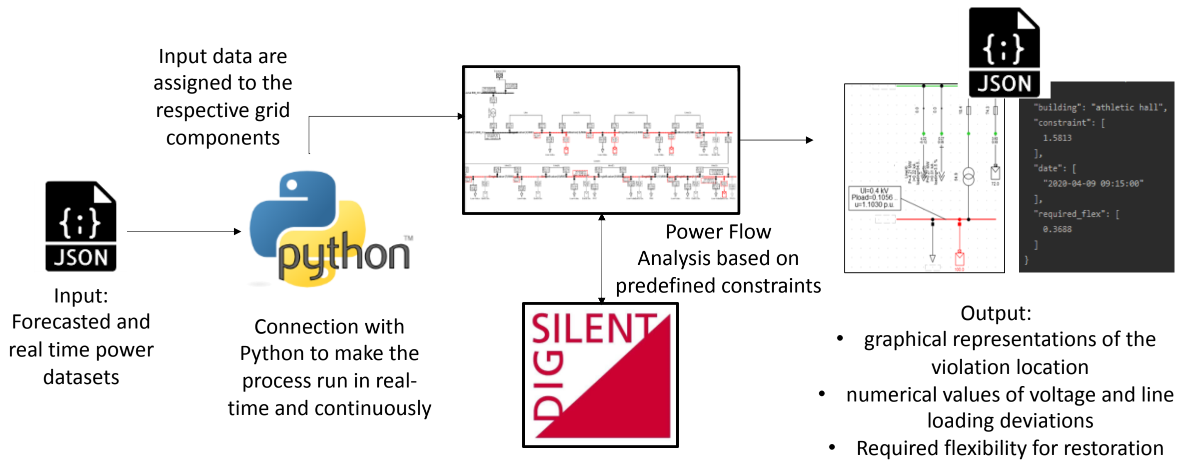


**Figure 3. Illustration of the GSSE operation and inputs/outputs.**

### *3.3.6.2 Functional Testing*

The GSSE component is able to identify any voltage or line loading issues, including time and specific location, occurring within the investigated power network along with the required flexibility for restoring the voltage and line loading levels back to nominal. The following table summarizes the tests performed for verifying the component's functionalities.

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Feeder Overload 1 | Prediction of Feeder Overload due to high loading Conditions | Feeder Loading Exciting 100%<br>Active Power Flow > 0 | 08/04/2020 |
| 2 | Feeder Overload 2 | Prediction of Feeder Overload due to excess generation (RES) | Feeder Loading Exciting 100%<br>Active Power Flow < 0 | 08/04/2020 |
| 3 | Overvoltage | Prediction of Busbar Overvoltage | Busbar Voltage < 1.1p.u | 04/04/2020 |
| 4 | Undervoltage | Prediction of Busbar Undervoltage | Busbar Voltage <0.95p.u | 09/04/2020 |

**Test Results**

**Test 1: Feeder Overload 1**

GGSE identifies an overload violation at Feeder 2 (101.07%) that will occur at 19:15:00 as shown in Figure 5. Active power flow of Feeder 2 at the violation time is positive, thus the expected overload will be caused due to high loading conditions (Figure 3). GSSE calculates the amount of flexibility needed (MW) to decrease in order to avoid overload as it can be seen in Figure 6. It should be mentioned that, for the overload cases GSSE estimates the required flexibility that can be provided by any flexibility service provider connected to the violated Feeder.


**Figure 4. Power Flow Analysis at the time of violation.**

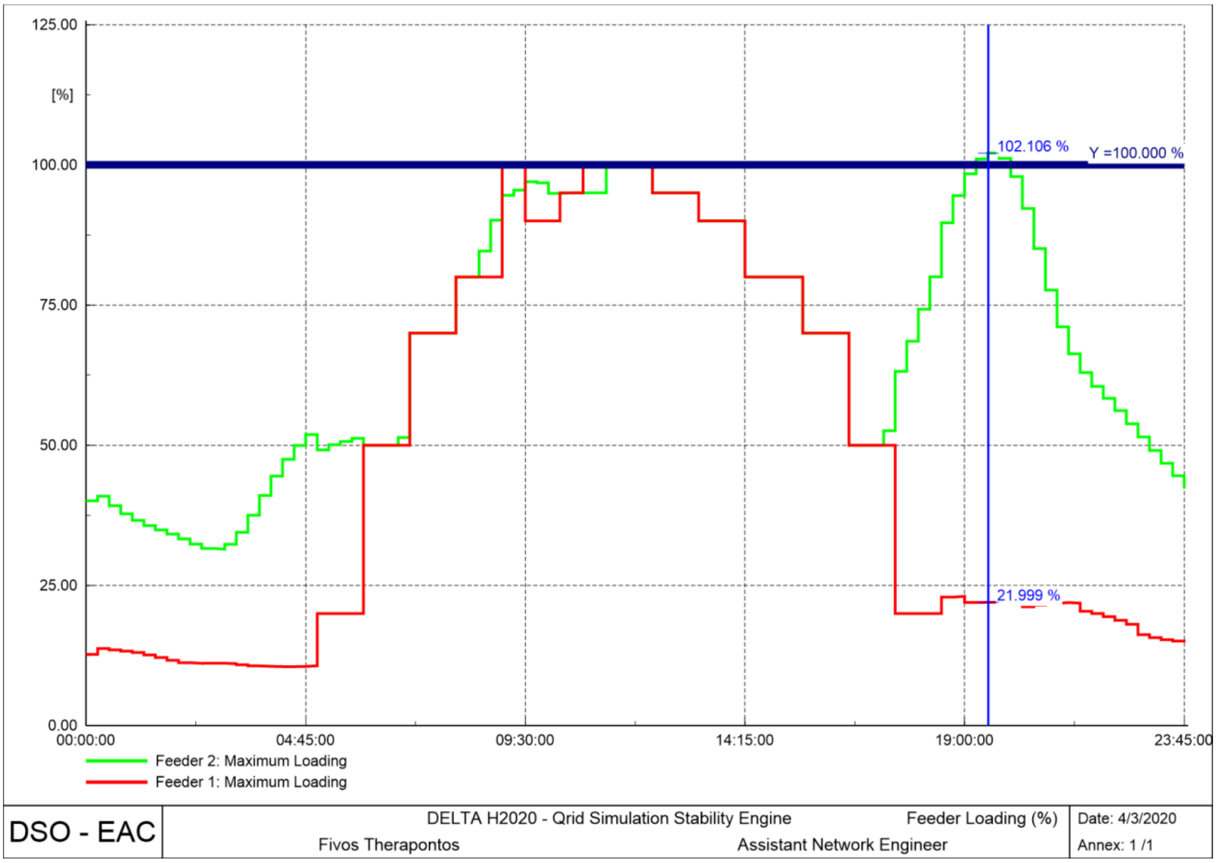

**Figure 5. Quasi-dynamic analysis – Feeders Loading.**

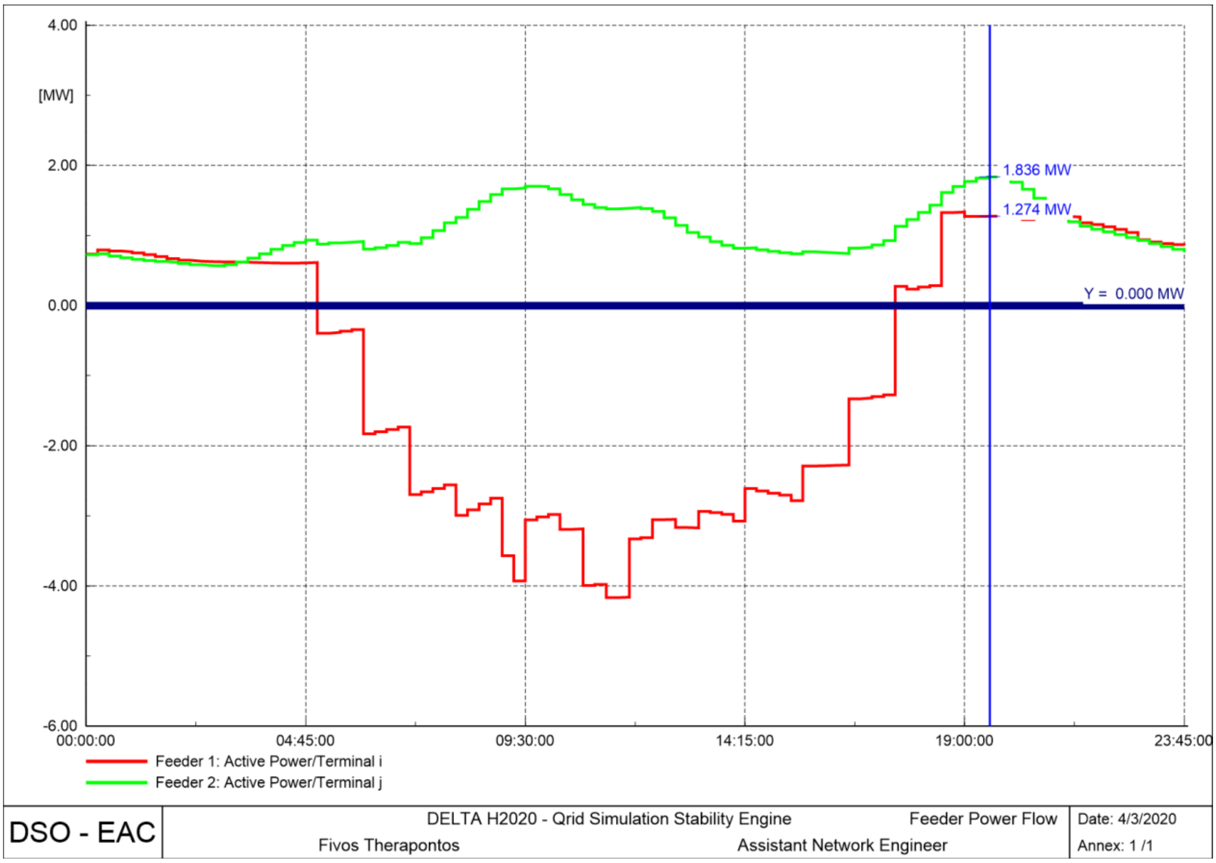


**Figure 6. Quasi-Dynamic analysis – Feeder Active Power Flow.**

Python output of GSSE that shows the violation data and the estimated flexibility request.

```
FEEDER OVERLOAD IDENTIFIED

Feeder 2 101.07 % at 19:15:00
Violation due to: High Loading Condition
Mitigation:
Reduce  Feeder 2  load at  19:15:00  by  0.019  MVA


Feeder 2 102.11 % at 19:30:00
Violation due to: High Loading Condition
Mitigation:
Reduce  Feeder 2  load at  19:30:00  by  0.038  MVA


Feeder 2 101.16 % at 19:45:00
Violation due to: High Loading Condition
Mitigation:
Reduce  Feeder 2  load at  19:45:00  by  0.021  MVA
```

### Test 2: Feeder Overload 2

GGSE predicts that an overload at Feeder 1 of 102.0% will occur from 11:15 until 11:30 as shown in Figure 8. Active power flow of Feeder 1 at the time of violation is negative, thus the expected overload will be caused due to excess RES generation (Figure 7). GSSE calculates the amount of flexibility needed (MW) to be increased in order to avoid overload as it can be seen in Figure 9.

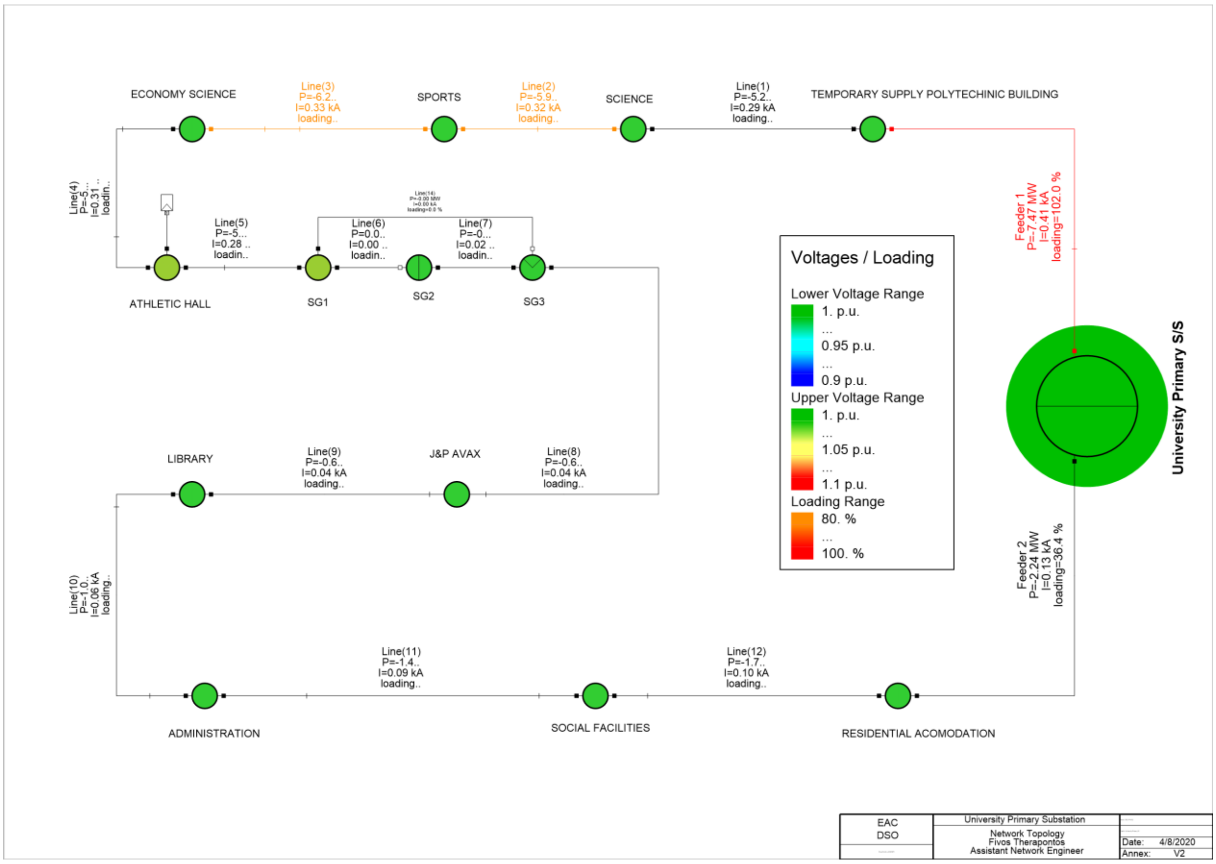


**Figure 7. Power Flow Analysis at the time of violation.**

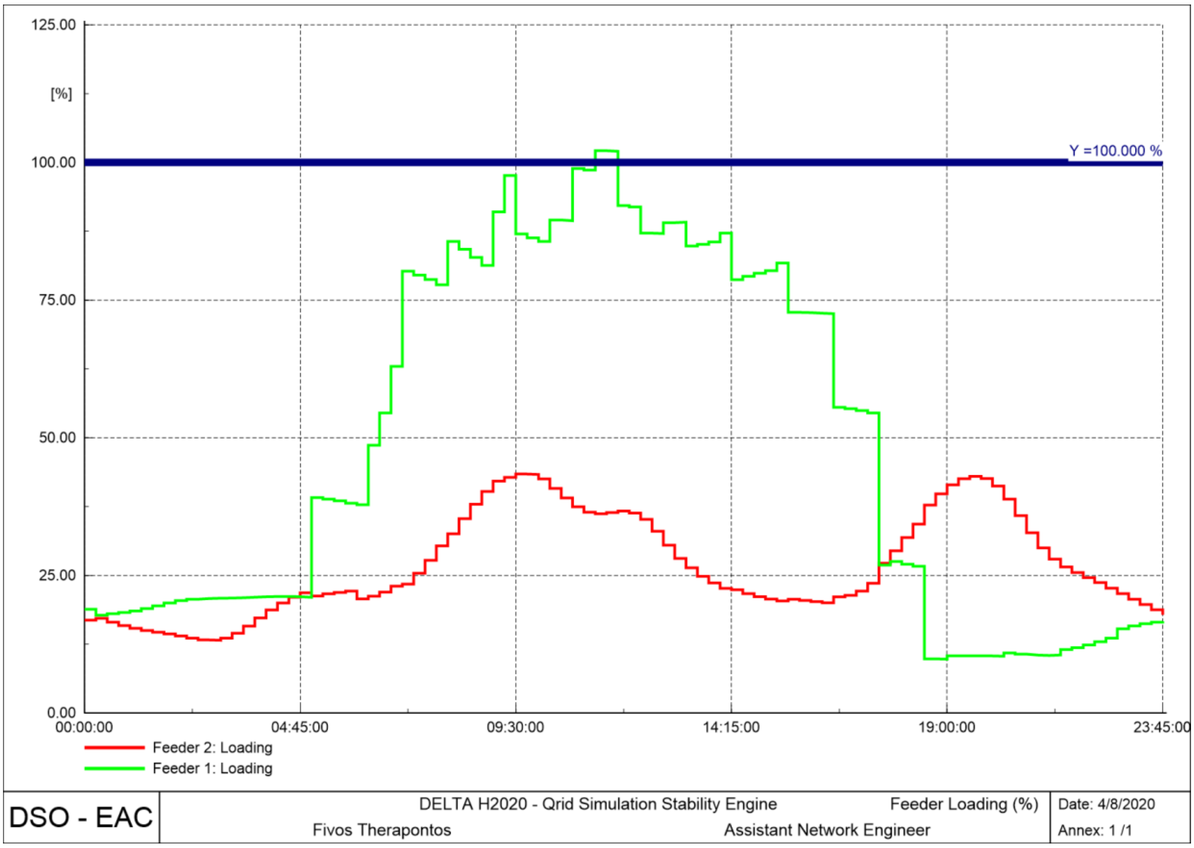


**Figure 8. Quasi-dynamic analysis – Feeders Loading.**

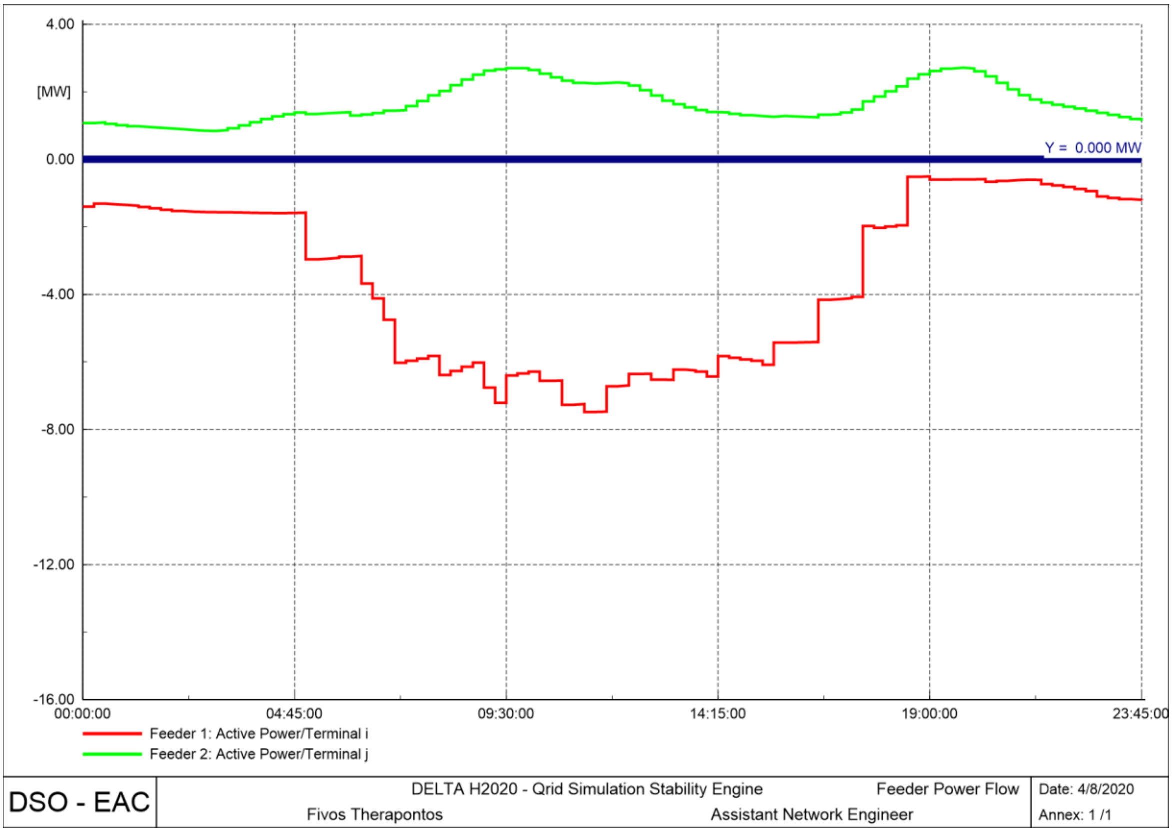

**Figure 9. Quasi-Dynamic analysis – Feeder Active Power Flow.**

Python output of GSSE that shows the violation data and the estimated flexibility request.

```
FEEDER OVERLOAD IDENTIFIED

Feeder 1 102.14 % at 11:15:00
Violation due to: Excess generation
Mitigation:
Increase  Feeder 1  load at  11:15:00  by  0.16  MVA


Feeder 1 102.04 % at 11:30:00
Violation due to: Excess generation
Mitigation:
Increase  Feeder 1  load at  11:30:00  by  0.152  MVA
```

**Test 3: Overvoltage**

GGSE identifies that the voltage of LV Busbar of Substation 115 (ATHLETIC HALL) will be above the nominal limits (Voltage > 1.1p.u). Specifically, as shown in the following figure, the 115 LV Busbar voltage is estimated to be 1.1016p.u at 09:00 and 1.103p.u at 09:15. GSSE calculates the amount of flexibility needed to avoid overvoltage by either increasing active power consumption or increasing reactive power consumption at LV Busbar 115.

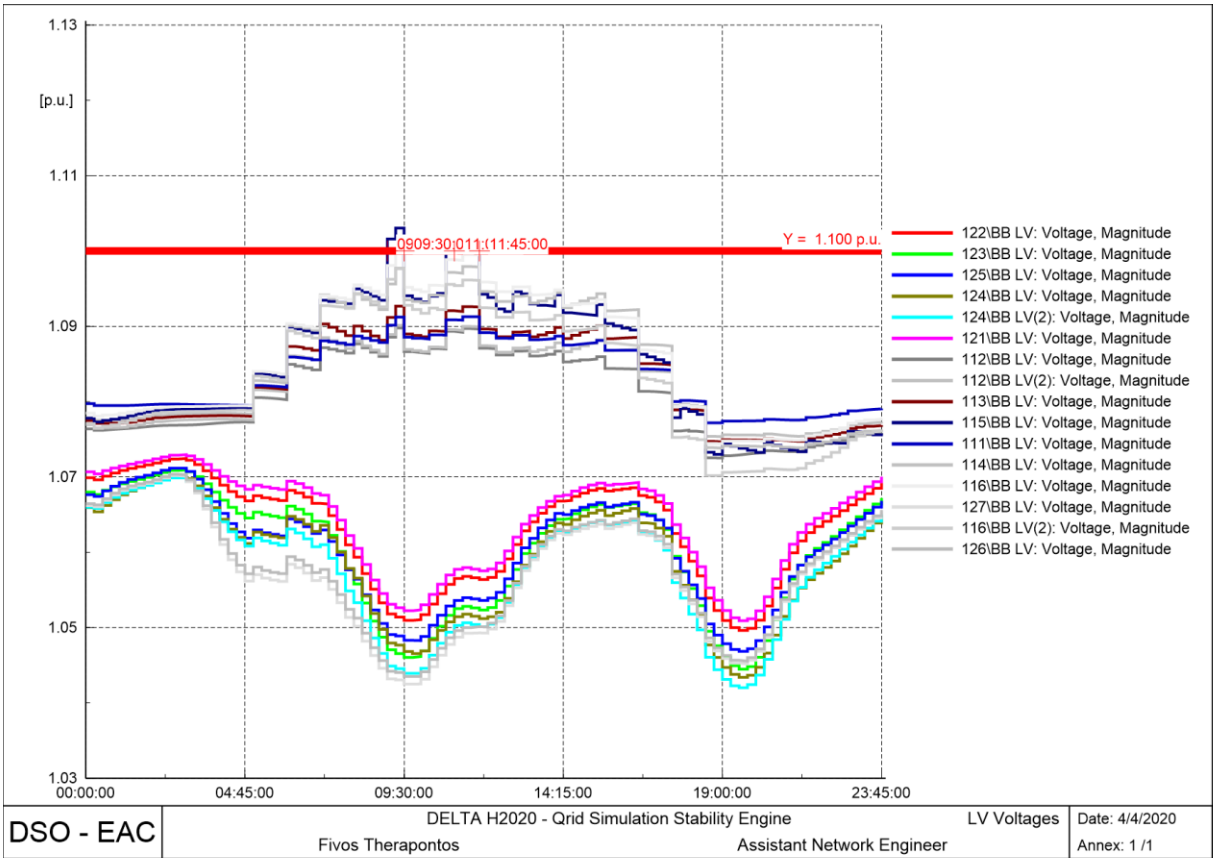

**Figure 10. Quasi-dynamic analysis – Overvoltage at Building/DVN busbars.**

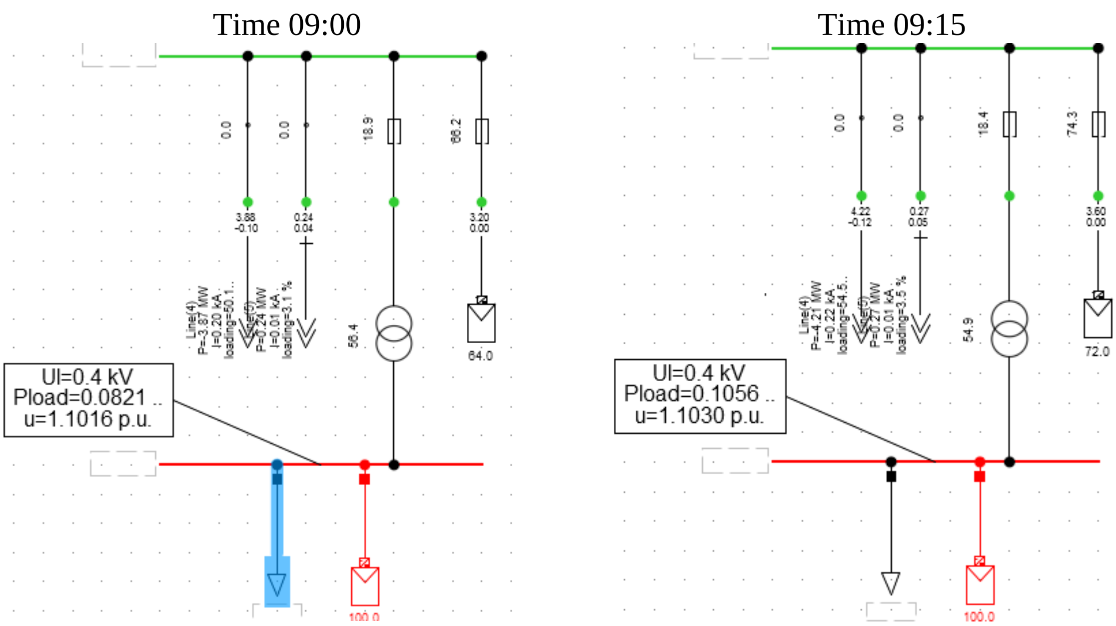


**Figure 11. Single Line Diagram of the violated LV Busbar.**

Python output of GSSE that shows the violation data and the estimated flexibility request.

```
OVERVOLTAGE VIOLATION IDENTIFIED
Location: ATHLETIC HALL
Busbar: BB LV
Voltage(p.u):  1.1016 p.u
Time: 09:00:00


Mitigation
1) Increase  active power consumption at substation  ATHLETIC HALL at  09:00:00 by  0.2018  MW
OR
2) Increase reactive power consumption at substation  ATHLETIC HALL at  09:00:00 by  0.0368  MVAr

 OVERVOLTAGE VIOLATION IDENTIFIED
 Location: ATHLETIC HALL
 Busbar: BB LV
 Voltage(p.u):  1.103 p.u
 Time: 09:15:00


 Mitigation
 1) Increase  active power consumption at substation  ATHLETIC HALL at  09:15:00 by  0.3688  MW
 OR
 2) Increase reactive power consumption at substation  ATHLETIC HALL at  09:15:00 by  0.0696  MVAr
```

The following figure shows the expected voltages at all LV Busbars (normal conditions) if the requested flexibility has been procured.

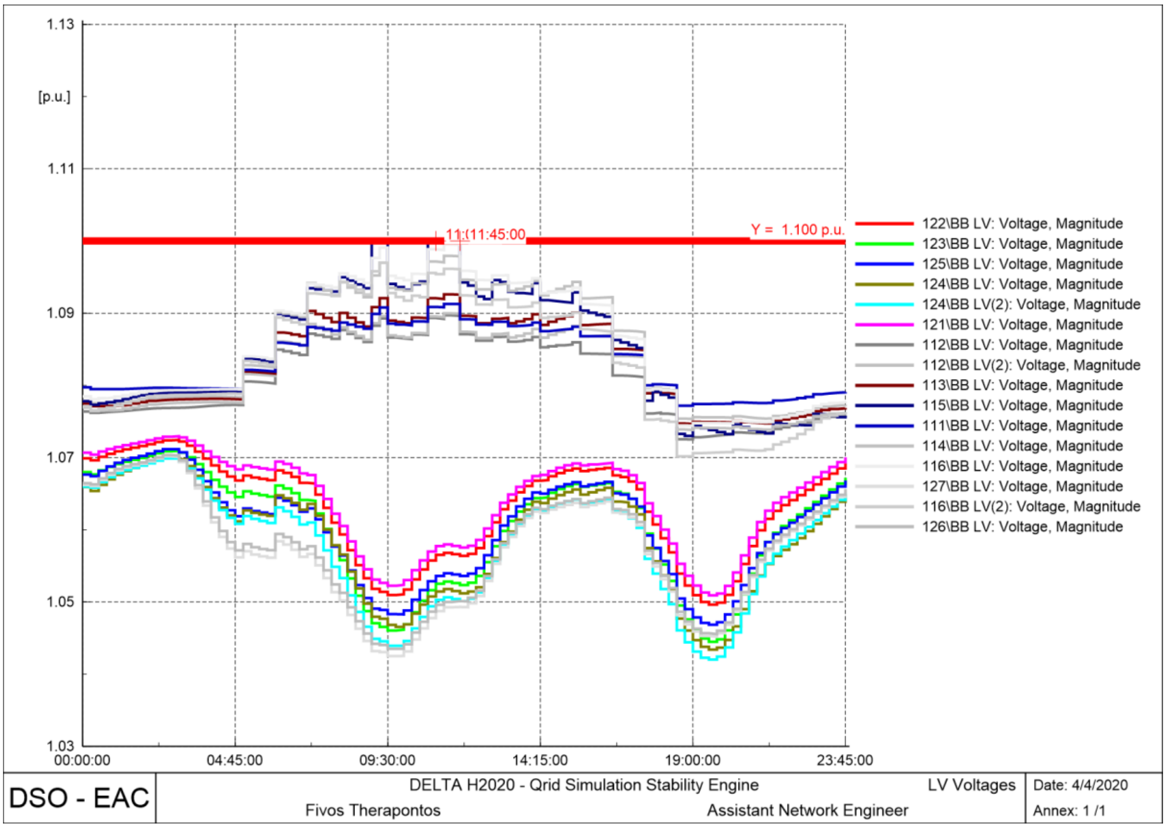


**Figure 12. Quasi-dynamic analysis – Voltage restoration at Building/DVN busbars (overvoltage test).**

The needed required flexibility for each case of overvoltage violation is send to AHO in a JSON format as shown below.


```
{
  "building": "athletic hall",
  "constraint": [
    1.5813
  ],
  "date": [
    "2020-04-09 09:15:00"
  ],
  "required_flex": [
    0.3688
  ]
}
```

```
{
  "building": "athletic hall",
  "constraint": [
    0.9464
  ],
  "date": [
    "2020-04-09 09:00:00"
  ],
  "required_flex": [
    0.2018
  ]
}
```


| Label | Description |
|---|---|
| Building | The name of the building that will have overvoltage violation |
| Constraint | The least predicted active power that the building must consume at the time when the violation was predicted |
| Date | The predicted date and time when an overvoltage violation will occur |

| Label | Description |
|---|---|
| Required_flex | The predicted required flexibility that the building will need to avoid violation (**upwards flexibility**) |

**Test 4: Undervoltage**

GGSE predicts that voltage of MV Busbar of Substation 126 (SG3) will be below the nominal limits (Voltage < 0.95 p.u). Specifically, 126 MV Busbar voltage is estimated to be lower than 0.95p.u from 09:00 until 10:15. As shown in Figure 13, GSSE calculates the amount of flexibility needed to avoid undervoltage.

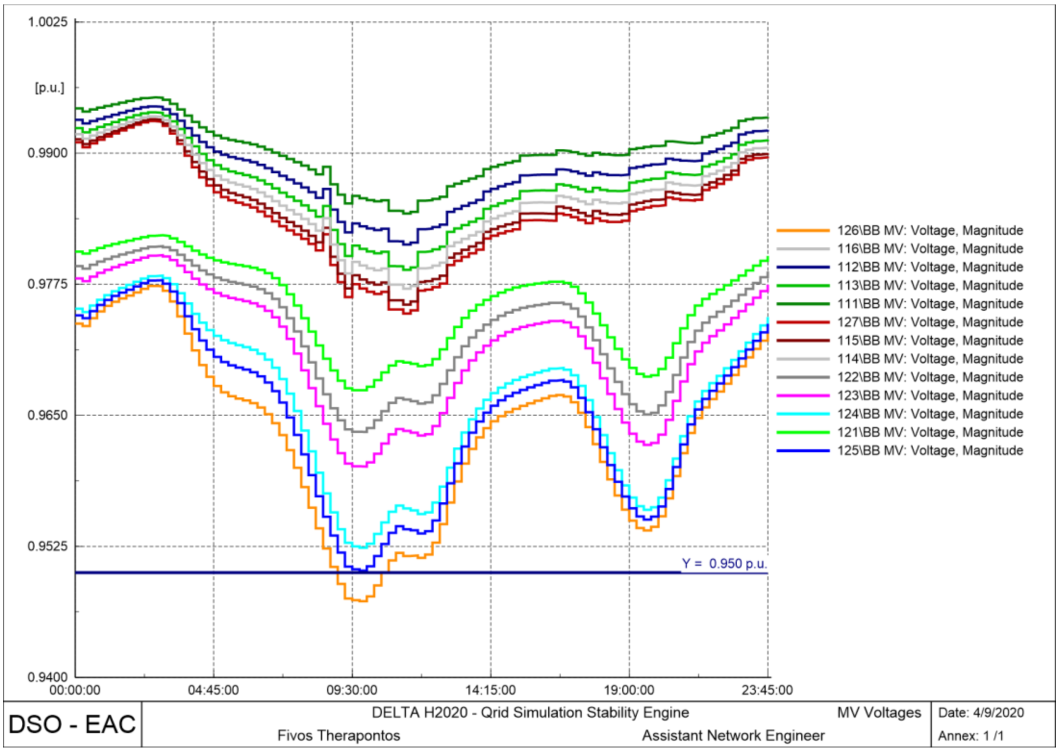


**Figure 13. Quasi-dynamic analysis – Undervoltage at Building/DVN busbars.**

Python output of GSSE that shows the violation data and the estimated flexibility request.

```
UNDERVOLTAGE VIOLATION IDENTIFIED
Location: SG3
Busbar: BB MV
Voltage(p.u):  0.9489 p.u
Time: 09:00:00


Mitigation
1) Decrease  active power consumption at substation  SG3 at  09:00:00 by  0.1569  MW
OR
2) Decrease reactive power consumption at substation  SG3 at  09:00:00 by  0.1698  MVAr
```

```
UNDERVOLTAGE VIOLATION IDENTIFIED
Location: SG3
Busbar: BB MV
Voltage(p.u):  0.9488 p.u
Time: 10:15:00


Mitigation
1) Decrease  active power consumption at substation  SG3 at  10:15:00 by  0.1692  MW
OR
2) Decrease reactive power consumption at substation  SG3 at  10:15:00 by  0.1936  MVAr
```

The expected voltages at all LV Busbars if the requested flexibility has been procured are illustrated in Figure 14.

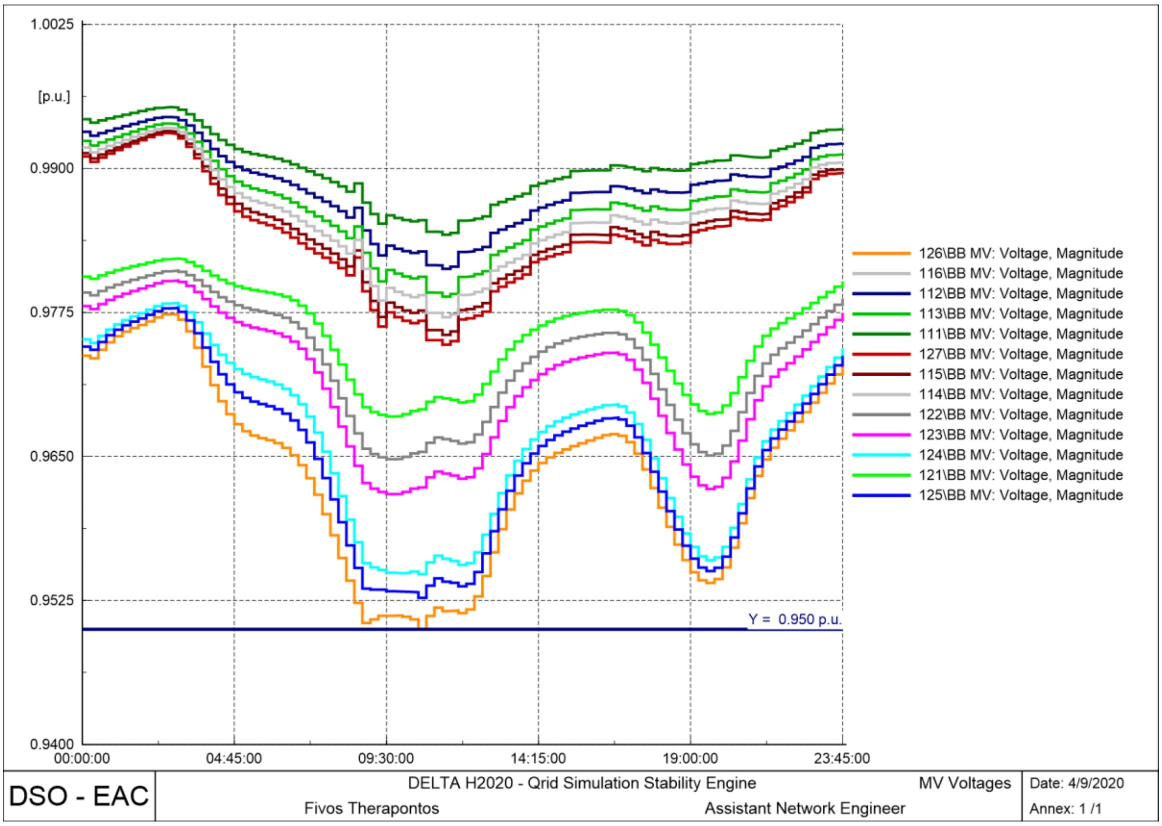


**Figure 14. Quasi-dynamic analysis – Voltage restoration at Building/DVN busbars (undervoltage test).**

Furthermore, GSSE script will send a JSON format file to AHO similar to overvoltage scenario as it is shown below. The difference at this scenario is that the required flexibility will be the amount of power that a building must decrease to avoid the predicted undervoltage violation.

```
{
  "building": "sg3",
  "constraint": [
    0.7279
  ],
  "date": [
    "2020-04-09 09:00:00"
  ],
  "required_flex": [
    0.1569
  ]
}
```

```
{
  "building": "sg3",
  "constraint": [
    0.667
  ],
  "date": [
    "2020-04-09 10:15:00"
  ],
  "required_flex": [
    0.1692
  ]
}
```


| Label | Description |
|---|---|
| Building | The name of the building that will have undervoltage violation |
| Constraint | The least predicted active power that the building must consume at the time when the violation was predicted |
| Date | The predicted date and time when an undervoltage violation will occur |
| Required_flex | The predicted reduction of power that the building will have to make at the predicted violation time to avoid undervoltage violation (**downwards flexibility**) |

### *3.3.7 Energy Portfolio Segmentation & Classification*

#### *3.3.7.1 Unit Testing*

Unit Testing Procedure applied over the Pytest module in order to evaluate the EPS&C module's functionalities. The basic test components focused on testing the eligibility of the following conditions: Results content, Results Format and successful communication with other components. Testing Procedure applied over several random inputs in order to guarantee that the EPS&C engine is not susceptible under any circumstances.

#### *3.3.7.2 Functional Testing*

| No | Test | Description | Evaluation criteria |
|---|---|---|---|
| 1 | Results Content | Evaluate the content of the result | Pass |
| 2 | Results Format | Examine that the basic structure of the results have the appropriate format | Pass |
| 3 | Communication | Examine the proper Communication with other Components like GSSE, FEIDs, DVNs | Pass |

## 3.4 Innovative Customer Engagement Tools

### *3.4.1 DR Visualisation Kit*

The DR Visualisation Kit provides two visualisation levels one for the Aggregator and one for the Customer. Therefore tests have been added to cover both levels.

#### *3.4.1.1 Unit Testing*

No unit testing has been performed yet.

#### *3.4.1.2 Functional Testing*

Aggregator Level:

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Handle missing Customers information | Handle failures while retrieving Customer information | Inform user for lack of Customers data | Pass |
| 2 | Display Customers information | Retrieve and display Customers information | Display all Customers and their information | Pass |
| 3 | Handle missing Historical Consumption information | Handle failures while retrieving Historical Consumption information | Inform user for lack of Historical Consumption data | Pass |
| 4 | Display Historical Consumption information | Retrieve and display Historical Consumption information | Display Historical Consumption data | Pass |
| 5 | Handle missing Historical Generation information | Handle failures while retrieving Historical Generation information | Inform user for lack of Historical Generation data | Pass |
| 6 | Display Historical Generation information | Retrieve and display Historical Generation information | Display Historical Generation data | Pass |
| 7 | Handle missing Forecasted Flexibility information | Handle failures while retrieving Forecasted Flexibility information | Inform user for lack of Forecasted Flexibility data | Pass |
| 8 | Display Forecasted Flexibility information | Retrieve and display Forecasted Flexibility information | Display Forecasted Flexibility data | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 9 | Handle missing DR Signals information | Handle failures while retrieving DR Signals information | Inform user for lack of DR Signals data | Pass |
| 10 | Display DR Signals | Display DR signals and information such as their status, time period, participating FEIDs etc. | Display DR Signals data | Pass |
| 11 | Handle missing Bids information | Handle failures while retrieving Bids information | Inform user for lack of Bids data | Pass |
| 12 | Display Bids information | Display Bids and information such as time period, responses, status etc. | Display Bids data | Pass |
| 13 | Handle missing Rewards information | Handle failures while retrieving Rewards information | Inform user for lack of Rewards data | Pass |
| 14 | Display Rewards information | Display available Rewards | Display Rewards data | Pass |
| 15 | Handle missing Energy price Profiling information | Handle failures while retrieving Energy price Profiling information | Inform user for lack of Energy price Profiling data | Pass |
| 16 | Display Energy price Profiling information | Display Energy price Profiling | Display Energy price Profiling data | Pass |
| 17 | Handle missing DVN Clusters information | Handle failures while retrieving DVN Clusters information | Inform user for lack of DVN Clusters data | Pass |
| 18 | Display DVN Clusters information | Display current DVN Clusters and their features | Display DVN Clusters data | Pass |
| 19 | Handle missing Node Profiling information | Handle failures while retrieving Node Profiling information | Inform user for lack of Node Profiling data | Pass |
| 20 | Display Node Profiling information | Display Node Profiles | Display Node Profiling data | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 21 | Handle missing Aggregated Profiling information | Handle failures while retrieving Aggregated Profiling information | Inform user for lack of Aggregated Profiling data | Pass |
| 22 | Display Aggregated Profiling information | Display Aggregated Profiles | Display Aggregated Profiling data | Pass |

Customer Level:

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Handle missing Rewards information | Handle failures while retrieving Customer's Rewards | Inform user for lack of Customer's Rewards data | Pass |
| 2 | Display Rewards information | Display Customer's current Rewards | Display Customer's data for Rewards up to date | Pass |
| 3 | Handle missing DR Signals information | Handle failures while retrieving DR Signals that Customer participated/declined | Inform user for lack of DR Signals data | Pass |
| 4 | Display DR Signals information | Display DR Signals that Customer participated/declined | Display data for DR Signals where Customer participated or declined | Pass |
| 5 | Handle missing FEID Energy Profile information | Handle failures while retrieving Customer's FEID Energy Profile | Inform user for lack of FEID Energy Profile data | Pass |
| 6 | Display FEID Energy Profile information | Display Customer's FEID Energy Profile | Display Customer's FEID Energy Profile data | Pass |

### *3.4.2 Award –enabled Energy Behavioural Platform*

#### *3.4.2.1 Unit Testing*

No unit testing has been performed yet.

#### *3.4.2.2 Functional Testing*

| No | Test | Description | Evaluation criteria | Results |
| --- | --- | --- | --- | --- |
| 1 | Track rewards | Track users' rewards history | Rewards are stored per user and per game | Pass |
| 2 | Store rewards | Store rewards earned by the end-user in the Award-Enable Energy behavioural platform | The awards are received from Award-Enable Energy behavioural platform | Pass |
| 3 | Provide to the end-users an overview of the real time data related to their physical devices | Produce a web based tool with demand response visualizations along with other visual analytics information | Provide correct and easy access to monitoring and control of assets as well as DR-related information | Pass |

### *3.4.3 Social Interaction and Cooperation Platform*

#### *3.4.3.1 Unit Testing*

No unit testing has been performed yet.

#### *3.4.3.2 Functional Testing*

| No | Test | Description | Evaluation criteria | Results |
| --- | --- | --- | --- | --- |
| 1 | Store Q&A | Store users discussions | Q&A are stored to database | Pass |
| 2 | Store user contacts | Store user contacts | Use can connect to other users and save them as contacts | Pass |
| 3 | Allow end-users to interact among them and the platform | Provide a collaboration platform that offers a large portfolio of useful activities, data and features | The platform should support discussion and knowledge diffusion, Q&A, chatting content posting, timeline of customer activities, social connections etc. | Pass |
| 4 | Gain access to data users will be interested in | Search database for previously asked questions and inserted data from other users | The Innovative customer engagement tools must provide information | Pass |

## 3.5 Common Information Modelling

The DELTA project implements a novel Semantic Interoperability architecture. The current approach is built upon two main pillars: an ontology and a software component called DELTA Common Information Model (CIM).

Semantic interoperability is the property that allows systems to exchange data, and more importantly, consume such data transparently [2]. Therefore, it is critical for Demand Response scenarios, in which different systems that take decisions must control third-party systems by sending them the data, the latter of which must use these data correctly to perform some local actions.

In general, Semantic Interoperability is built upon three layers [3], namely: technical, syntactic, and semantic interoperability. Technical interoperability refers to heterogeneous protocols and mechanisms that can be used to exchange data [4] Syntactic interoperability refers to the heterogeneity of formats that data may adopt [5] Finally, semantic interoperability refers to how data is modelled and its incurred meaning [6].

The approach implemented in DELTA consists of establishing a common data model, format and mechanism to exchange data, i.e., homogenising data in the three semantic interoperability layers. To achieve such goal, this approach relies on semantic web technologies. As a result, the syntactic layer is achieved by requiring systems to use any serialisation of RDF, e.g., Turtle, JSON-LD, or N3.

The semantic layer is achieved by establishing an ontology to be used by the involved systems, which must cover Demand Response concepts. However, to our knowledge, there is no ontology for DR. To address this issue, in DELTA, we have developed and published[2], a semantic and enriched version of the OpenADR standard, to which we refer as the OpenADR ontology.

The OpenADR standard already establishes the mechanisms that can be employed to exchange data [7], e.g., REST APIs, which can be invoked by agents participating in a P2P, network with specific features. In order to meet these requirements, we developed the CIM.

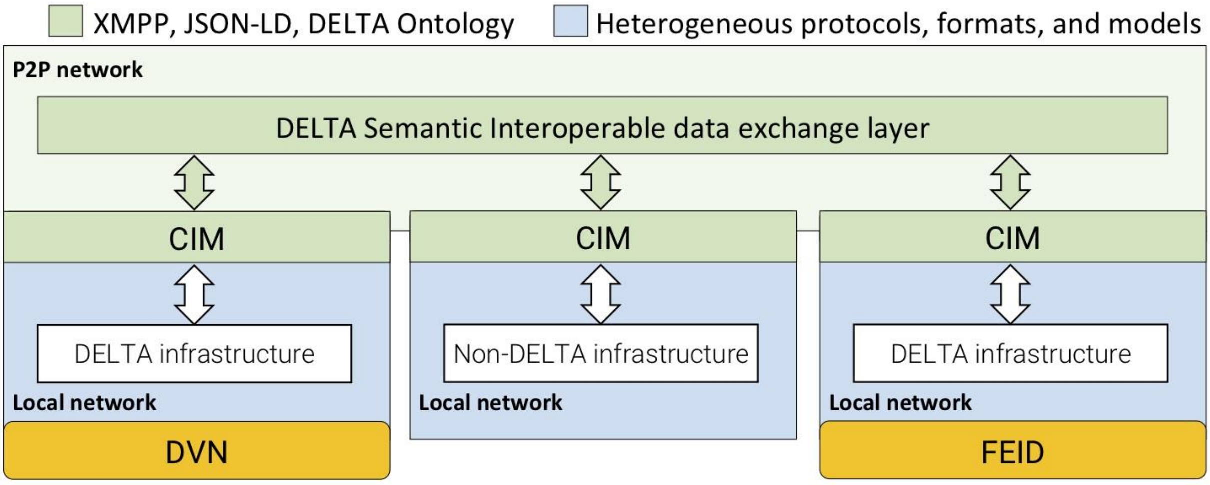


**Figure 15 – CIM Semantic Interoperability overview**

[2]https://albaizq.github.io/OpenADRontology/OnToology/ontology/openADRontology.owl/documentation/index-en.html

The CIM is the DELTA component that interconnects the rest of components in the DELTA platform and allows them to transparently exchange data, as depicted in Figure 1 In addition, we are currently working for the CIM to offer to those components that do not meet the DELTA interoperability requirements, either technical, syntactic, or semantic, a mechanism to be DELTA compliant. As a result, the CIM will be able to interconnect a DELTA component with a non-DELTA compliant component.

Figure 1 depicts how the CIM is deployed as a sub-component of both the DELTA Virtual Node (DVN) and the Fog-Enabled Intelligent Device (FEID). These are two distinct components of the DELTA architecture that employ the CIM to communicate DR signals by employing the DELTA (OpenADR-compliant) ontology between its technical layers (i.e., aggregator, virtual nodes, and customers).

In following sub-sections the tests performed to validate the Semantic Interoperability will be reported.

#### *3.5.1.1 Unit Testing*

For the CIM no Unit has been carried out, since testing Semantic Interoperability requires some tests more complex tests than just unitary tests; as reported in the next sub-section.

#### *3.5.1.2 Functional Testing*

In the following table the tests performed to validate the Semantic Interoperability implemented in DELTA are presented. Notice that usually the tests cover the technical interoperability by sending data and checking that data has been received, and then, checking that even if some data sent had a format that is not JSON-LD and a model that is not DELTA ontology, the data received must have those two requirements.

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Technical Interoperability using JSON-LD and SAREF | Test the correct interaction between the FEID and the DVN by sending and correctly receiving packages of data, payloads are expressed in JSON-LD with SAREF | Step 1: All other links are considered fully operational.<br>Step 2: Send 100 messages.<br>Step 3: Evaluate receipt of 100 messages.<br>Step 4: Validate integrity of received messages.<br>Step 5: Output test verdict. | Pass |
| 2 | Syntactic and Semantic Interoperability using JSON-LD and SAREF | Test that the messages transmitted in Test 1 have the proper format (JSON-LD) and use the SAREF Ontology. | Step 1: A PASS in the communication layer is verified.<br>Step 2: Validate that the data received have the proper format (Syntactic Interoperability).<br>Step 3: Validate that the data received have the proper model by means of the DELTA SHACL Shapes (Semantic Interoperability).<br>Step 4: Output test verdict. | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 3 | Technical Interoperability using JSON-LD and SAREF4ENER | Test the correct interaction between the FEID and the DVN by sending and correctly receiving packages of data, payloads are expressed in JSON-LD with SAREF4ENER | Step 1: All other links are considered fully operational.<br>Step 2: Send 100 messages.<br>Step 3: Evaluate receipt of 100 messages.<br>Step 4: Validate integrity of received messages.<br>Step 5: Output test verdict. | Pass |
| 4 | Syntactic and Semantic Interoperability using JSON-LD and SAREF4ENER | Test that the messages transmitted in Test 1 have the proper format (JSON-LD) and use the SAREF4ENER Ontology. | Step 1: A PASS in the communication layer is verified.<br>Step 2: Validate that the data received have the proper format (Syntactic Interoperability).<br>Step 3: Validate that the data received have the proper model by means of the DELTA SHACL Shapes (Semantic Interoperability).<br>Step 4: Output test verdict. | Fail |
| 5 | Technical Interoperability using XML and the model of OpenADR standard | Test the correct interaction between the FEID and the DVN by sending and correctly receiving packages of data, payloads are expressed in XML with OpenADR | Step 1: All other links are considered fully operational.<br>Step 2: Send 100 messages.<br>Step 3: Evaluate receipt of 100 messages.<br>Step 4: Validate integrity of received messages.<br>Step 5: Output test verdict. | Pass |
| 6 | Syntactic and Semantic Interoperability using XML and the model of OpenADR standard | Test that the messages transmitted in Test 1 have the proper format (JSON-LD) and use the DELTA Ontology. | Step 1: A PASS in the communication layer is verified.<br>Step 2: Validate that the data received have the proper format (Syntactic Interoperability).<br>Step 3: Validate that the data received have the proper model by means of the DELTA SHACL Shapes (Semantic Interoperability).<br>Step 4: Output test verdict. | Fail |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 7 | Technical Interoperability using JSON-LD and OpenADR ontology | Test the correct interaction between the FEID and the DVN by sending and correctly receiving packages of data, payloads are expressed in JSON-LD with OpenADR ontology | Step 1: All other links are considered fully operational.<br>Step 2: Send 100 messages.<br>Step 3: Evaluate receipt of 100 messages.<br>Step 4: Validate integrity of received messages.<br>Step 5: Output test verdict. | Pass |
| 8 | Syntactic and Semantic Interoperability using JSON-LD and OpenADR ontology | Test that the messages transmitted in Test 1 have the proper format (JSON-LD) and use the OpenADR Ontology. | Step 1: A PASS in the communication layer is verified.<br>Step 2: Validate that the data received have the proper format (Syntactic Interoperability).<br>Step 3: Validate that the data received have the proper model by means of the DELTA SHACL Shapes (Semantic Interoperability).<br>Step 4: Output test verdict. | Pass |
| 9 | Technical Interoperability using JSON-LD and DELTA Ontology | Test the correct interaction between the FEID and the DVN by sending and correctly receiving packages of data, payloads are expressed in JSON-LD with DELTA Ontology | Step 1: All other links are considered fully operational.<br>Step 2: Send 100 messages.<br>Step 3: Evaluate receipt of 100 messages.<br>Step 4: Validate integrity of received messages.<br>Step 5: Output test verdict. | Pass |
| 10 | Syntactic and Semantic Interoperability using JSON-LD and DELTA Ontology | Test that the messages transmitted in Test 1 have the proper format (JSON-LD) and use the DELTA Ontology. | Step 1: A PASS in the communication layer is verified.<br>Step 2: Validate that the data received have the proper format (Syntactic Interoperability).<br>Step 3: Validate that the data received have the proper model by means of the DELTA SHACL Shapes (Semantic Interoperability).<br>Step 4: Output test verdict. | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 11 | Technical Interoperability using TURTLE and DELTA Ontology | Test the correct interaction between the FEID and the DVN by sending and correctly receiving packages of data, payloads are expressed in JSON-LD with DELTA Ontology | Step 1: All other links are considered fully operational.<br>Step 2: Send 100 messages.<br>Step 3: Evaluate receipt of 100 messages.<br>Step 4: Validate integrity of received messages.<br>Step 5: Output test verdict. | Pass |
| 12 | Syntactic and Semantic Interoperability using JSON-LD and DELTA Ontology | Test that the messages transmitted in Test 1 have the proper format (JSON-LD) and use the DELTA Ontology. | Step 1: A PASS in the communication layer is verified.<br>Step 2: Validate that the data received have the proper format (Syntactic Interoperability).<br>Step 3: Validate that the data received have the proper model by means of the DELTA SHACL Shapes (Semantic Interoperability).<br>Step 4: Output test verdict. | Pass |

Notice that some tests are not PASS, this is due to the capability of the CIM to integrate external non-DELTA components in which we are still working.

## 3.6 Cybersecurity Services

### *3.6.1 DELTA Blockchain*

#### *3.6.1.1 Unit Testing*

The DELTA blockchain network is a critical component of the DELTA platform, that enables the collaboration between the aggregator and the prosumers with regards to demand and response schemes.

The main components of the blockchain network that shall be tested are the Aggregator's CA, the peers of the network and the ordering service. These components are going to be tested in order to validate that each one functions properly. Specifically, the unit tests are:

Aggregator CA:

- An identity is registered properly
- An identity is revoked properly
- The CA responses with an updated CRL when asked.

Network peers:

- A query transaction is served properly
- An update transaction is served properly
- An identity that has not the required arguments cannot make any transactions against the peer

Ordering service:

- A transaction that is endorsed by peers is added to the ledger of the channel

- A transaction that is not properly endorsed by the peers is rejected

#### *3.6.1.2 Functional Testing*

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Identity enrollment | A new identity is registered and a component issues a CSR request | Certificates are generated and are communicated to the client along with private keys | True / False |
| 2 | Identity revocation | The certificates for an identity are revoked | The client with the revoked certificates cannot communicate to peers | True / False |
| 3 | Certificates distribution | The signing certificate for a new identity is issued | This certificate is communicated to all existing clients (endpoints) and the new client gets all certificates issued up to this point | True / False |
| 4 | CRL update | A certificate is revoked | The CRL of all other existing clients is updated | True / False |

### *3.6.2 Smart Contracts*

#### *3.6.2.1 Unit Testing*

As DELTA's smart contracts are developed in Go, we employed the standard testing tool that the language provides, i.e., "go test".

| No | Test | Description | Results |
|---|---|---|---|
| 1 | Payload Parsing | Validate the correctness of the parsers that the DR Management smart contract employs to convert OadrDistributeEvent JSON payloads to Go structures. | Pass |
| 2 | Payload Validation | Validate the checks that the DR Management smart contract performs to judge whether OadrDistributeEvent JSON payloads are malicious or not. | Pass |
| 3 | DR Event Queries | Validate that the smart contract functions that expose data return the expected results. | Pass |
| 4 | DR Issue | Validate that the appropriate actors can issue OadrDistributeEvents and that their status is marked appropriate. | Pass |
| 5 | DR Honest Lifecycle | Validate the entire state transition of an event's lifecycle under honest interactions. | Pass |

| | | | |
|---|---|---|---|
| 6 | DR Invalid Lifecycle | Validate that the smart contract prohibits state transitions in the event's lifecycle that do not comply to its state machine. | Pass |
| 7 | DR Honest Complete Report | Validate that the smart contract successfully marks the completion of an event on input a report that corresponds to what was requested by the event. | Pass |
| 8 | DR Invalid Complete Report | Validate that the smart contract does not mark as completed an event on input an invalid report. | Pass |
| 9 | DR Honest Fail Report | Validate that the smart contract successfully marks an event as failed on input a report that proves a deviation compared to what was requested by the event. | Pass |
| 10 | DR Invalid Fail Report | Validate that the smart contract does not mark as failed an event on input an invalid report. | Pass |
| 11 | DR Point Allocation | Validate that on successful completion of a DR event, the smart contract correctly allocates points to the target VENs. | Pass |
| 12 | DR Penalty Allocation | Validate that on failure of a DR event, the smart contract correctly distributes the penalty to the VEN. | Pass |

More information about the Smart contracts unit testing has been documented in D5.2.

#### *3.6.2.2 Functional Testing*

For the smart contracts, functional testing has been integrated into the unit testing as shown above.

### *3.6.3 Threat Mitigation*

As this component is still on the early stages of development no unit or functional testing has been performed yet.

#### *3.6.3.1 Unit Testing*

-

#### *3.6.3.2 Functional Testing*

-

# 4. Individual Component Testing – Plan

## 4.1 DELTA Customer

### *4.1.1 Fog-enabled Intelligent Device*

#### *4.1.1.1 Hardware Testing*

FEID add-ons will be tested in a similar way with the main board. Initially, the voltage level of the input pins will be checked if it in compliance with the value specified in the requirements. An automated test script will be then used for the validation of the communication between the FEID's main board and the connected add-on.

#### *4.1.1.2 Unit Testing*

Extensive Unit testing will be performed using pytest or SonarQube upon completion of individual sub-component implementation. Results will be included in D6.4 on M32.

#### *4.1.1.3 Functional Testing*

Most of iterative tests performed so far (presented in Section 3.1.1) will be repeated when the final FEID component is delivered. On top of that the following tests will also be included in the process.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Assess embedded hardware Security | Validate security measures that have been established through the Trusted Platform Module | FEID information is stored in a secure way in the TPM and no unauthorized access is allowed | M26-M28 |
| 2 | Data Integrity for Add-on Protocols | Validate robust communication with each protocol supported by the each FEID add-on | Data integrity for EnOcean Protocol Add-on | M26-M28 |
| | | | Data integrity for LoRa Protocol Add-on | M26-M28 |
| | | | Data integrity for NB-IoT Protocol Add-on | M26-M28 |
| | | | Data integrity for combined Add-ons | M27-M29 |
| 3 | Automatic Restart upon | Install a supervising sub-component for restarting partially or completely FEID algorithms upon encountering measurement or connectivity malfunction | Successful restart and 100 % uptime operation given power supply available | M25-M26 |

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 4 | Validate system calls' provision for TPM to applications | Successful access to all required software components to store and retrieve in a secure manner information from the TPM (could be tested during unit testing) | Data integrity and robust communication with the TPM from various FEID functions | M27-M29 |
| 5 | Check for data security against different attack techniques | Deploy specific cyber/physical attacks to validate security aspects offered by the TPM | Successful prevention of all attacks (details will be elaborated in D5.3 and D6.4) | M27-M30 |

## 4.2 DELTA Virtual Node

### *4.2.1 Consumer/Prosumer Flexibility Data Monitoring and Profiling*

#### *4.2.1.1 Unit Testing*

Extensive Unit testing will be performed using pytest or SonarQube upon completion of individual sub-component implementation. Results will be included in D6.4 on M32.

#### *4.2.1.2 Functional Testing*

Upon new releases, all previously performed tests (see Section 3.2) will be repeated to ensure sustainable performance. On top of that the following tests will also be included in the process.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Error Notification system performance | Missing data and lost communication with FEIDs will be reported and logged. This test will ensure the system works as expected | Missing Data, Lost Communication, Corrupted data, and other errors successfully being documented and reported to the aggregator | M25-M26 |

### *4.2.2 Generation/Consumption Optimal Dispatch*

#### *4.2.2.1 Unit Testing*

Extensive Unit testing will be performed using pytest or SonarQube upon completion of individual sub-component implementation. Results will be included in D6.4 on M32.

#### *4.2.2.2 Functional Testing*

Upon new releases, all previously performed tests (see Section 3.2) will be repeated to ensure sustainable performance. On top of that the following tests will also be included in the process.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Sensitivity analysis for DR setpoints | The available solution space highly depends of the setpoint of explicit DR signals. Thus, the setpoint must be within reasonable ranges in order to ensure calculation of an optimal solution. | All the variables that have an impact on the DR setpoints have been correctly defined | M26-M30 |
| 2 | Scale Up Testing | Perform all previous tests for 100 and 1000 customers | Each test’s criteria | M27-M30 |

### *4.2.3 Load Forecasting*

#### *4.2.3.1 Unit Testing*

Unit testing will be performed using pytest or SonarQube. Results will be included in D6.4 on M32.

#### *4.2.3.2 Functional Testing*

Upon new releases, all previously performed tests (see Section 3.2) will be repeated to ensure sustainable performance. On top of that the following tests will also be included in the process.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Forecasting Aggregation Performance from the assigned FEIDs | One of the functionalities of the forecasting engine at DVN level is to aggregate the forecasted results from each FEID to provide the DVN forecasts. This test will validate this functionality | Validate the aggregation results | M25-M26 |

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 2 | Weighted Combination Performance | The combination of the individual FEID forecasting aggregation and the forecasting of aggregated measurements | Validate improved behaviour under different operational scenarios (e.g. missing FEID data, unexpected consumption patterns, etc.) | M25-M26 |
| 3 | Validate adaptive re-training | All forecasting module will adaptively re-train after a given time (e.g. once per month) There will be tests to ensure that the performance and accuracy of the modules remains the same or even improves over time | Upon each re-train, performance and accuracy tests on previous data will be executed | M26-M28 |

### *4.2.4 Inter/Intra Node Energy Matchmaking*

#### *4.2.4.1 Unit Testing*

Unit testing will be performed using pytest or SonarQube. Results will be included in D6.4 on M32.

#### *4.2.4.2 Functional Testing*

Upon new releases, all previously performed tests (see Section 3.2) will be repeated to ensure sustainable performance. On top of that the following tests will also be included in the process.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Scale Up Testing | Perform all previous tests for 100 and 1000 customers with multiple failure scenarios | Each test's criteria | M27-M30 |

### *4.2.5 Consumer/Prosumer Energy/Social Clustering*

#### *4.2.5.1 Unit Testing*

Pytest tests will be repeated and extended upon new releases. Results will be included in D6.4 on M32.

#### *4.2.5.2 Functional Testing*

Upon new releases, all previously performed tests (see Section 3.2) will be repeated to ensure sustainable performance. On top of that the following tests will also be included in the process.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Social Engagement Clustering Performance | Validate clustering results based on engagement strategies and gamification results | Silhouette score > 0.75 | M26-M27 |
| 2 | Assess additional feature extraction | Multiple tests will be executed to assess additional energy and social clustering features | Correlation metrics<br>Clustering metrics | M27-M30 |
| 3 | Validate adaptive re-clustering | The spatial and temporal clustering will be updated after a given time (e.g. once per week)<br>There will be tests to ensure that the performance and clustering results of the module remain the same or even improve over time | Upon each re-clustering, performance and accuracy tests on previous data will be executed | M26-M28 |

## 4.3 DELTA Aggregator

### *4.3.1 Energy Market Price Forecast*

#### *4.3.1.1 Unit Testing*

Necessary unit testing has already been concluded. If needed additional tests will be executed.

#### *4.3.1.2 Functional Testing*

Upon new releases, all previously performed tests (see Section 3.3) will be repeated to ensure sustainable performance. On top of that the following tests will also be included in the process.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Net Imbalance Forecasting Performance | In an effort to improve the performance of the energy price forecasting tools, the net volume imbalance will be forecasted. As with other forecasting tools, various performance aspects and metrics will be assessed. | Execution time and Accuracy | M25-M26 |
| 2 | Multi-step forecasting on real-life performance | The various forecasting price schemes for 1, 2, 4, 8, 12, and 24 hours ahead will be assessed in real-time operation | Execution time and Accuracy | M25-M26 |

### *4.3.2 DR & Flexibility Forecasting*

#### *4.3.2.1 Unit Testing*

Pytest tests will be repeated and extended upon new releases. Results will be included in D6.4 on M32.

#### *4.3.2.2 Functional Testing*

Upon new releases, all previously performed tests (see Section 3.3) will be repeated to ensure sustainable performance. On top of that the following tests will also be included in the process.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Forecasting Aggregation Performance from the assigned DVNs | One of the functionalities of the forecasting engine at Aggregator level is to aggregate the forecasted results from each DVN to provide the Aggregator forecasts. This test will validate this functionality | Validate the aggregation results | M25-M26 |

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 2 | Weighted Combination Performance | The combination of the individual DVN forecasting aggregation and the forecasting of aggregated measurements | Validate improved behaviour under different operational scenarios | M27-M29 |
| 3 | Validate adaptive re-training | All forecasting module will adaptively re-train after a given time (e.g. once per month) There will be tests to ensure that the performance and accuracy of the modules remains the same or even improves over time | Upon each re-train, performance and accuracy tests on previous data will be executed | M27-M29 |

### *4.3.3 Node Flexibility Data Monitoring and Profiling*

#### *4.3.3.1 Unit Testing*

This component as part of the DSS has been developed in the same coding package as the Asset Handling Optimization. As such the unit testing has been performed in the combined version. Beyond that, functional testing has been performed during development as follows.

#### *4.3.3.2 Functional Testing*

All previous tests that have been deployed so far will be repeated until reaching the final version of this component. In addition, the following tests will be executed.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Error Notification system performance | Missing data and lost communication with DVNs will be reported and logged. This test will ensure the system works as expected | Missing Data, Lost Communication, Corrupted data, and other errors successfully being documented and reported to the aggregator | M25-M26 |

### *4.3.4 Asset Handling Optimization*

#### *4.3.4.1 Unit Testing*

Pytest tests will be repeated and extended upon new releases. Results will be included in D6.4 on M32.

#### *4.3.4.2 Functional Testing*

Upon new releases, all previously performed tests (see Section 3.3) will be repeated to ensure sustainable performance. On top of that the following tests will also be included in the process.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Evaluation of the Optimization's efficiency in Responsiveness | Evaluate the improvements as far as the Responsiveness of the DVNs to DRs concerned | Compare the Responsiveness of DVNs through an optimized DR with a standard DR. | M24-25 |
| 2 | Evaluation of the Optimization efficiency in Profitability | Evaluate the Profit of the aggregator as far as the DR yield. | Compare the Profits of the Aggregator through the Optimized DR | M24-25 |

### *4.3.5 Self-Portfolio Energy Balancing*

The component is considered final and not further testing is required. Nevertheless, as the integration is still ongoing, potential testing may be required if further development occur.

### *4.3.6 DELTA Grid State Simulation - Grid Stability Simulation Engine*

The component is considered final and not further testing is required. Nevertheless, as the integration is still ongoing, potential testing may be required if further development occur.

### *4.3.7 Energy Portfolio Segmentation & Classification*

#### *4.3.7.1 Unit Testing*

Extensive Unit testing will be performed using pytest or SonarQube upon completion of individual sub-component implementation. Results will be included in D6.4 on M32.

#### *4.3.7.2 Functional Testing*

Upon new releases, all previously performed tests (see Section 3.3) will be repeated to ensure sustainable performance.

## 4.4 Innovative Customer Engagement Tools

### *4.4.1 DR Visualisation Kit*

As already mentioned DR Visualisation Kit contains two levels Aggregator and Customer, hence relative tests are grouped by level.

#### *4.4.1.1 Unit Testing*

The component will be evaluated using when completed using the jestjs tool[3].

#### *4.4.1.2 Functional Testing*

Aggregator: Level

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Handle missing Customers information | Handle failures while retrieving Customer information | Inform user for lack of Customers data | M24 –M25 |
| 2 | Display Customers information | Retrieve and display Customers information | Display all Customers and their information | M24 –M25 |
| 3 | Handle missing Historical Consumption information | Handle failures while retrieving Historical Consumption information | Inform user for lack of Historical Consumption data | M24 –M25 |
| 4 | Display Historical Consumption information | Retrieve and display Historical Consumption information | Display Historical Consumption data | M24 –M25 |
| 5 | Handle missing Historical Generation information | Handle failures while retrieving Historical Generation information | Inform user for lack of Historical Generation data | M24 –M25 |
| 6 | Display Historical Generation information | Retrieve and display Historical Generation information | Display Historical Generation data | M24 –M25 |
| 7 | Handle missing Forecasted Flexibility information | Handle failures while retrieving Forecasted Flexibility information | Inform user for lack of Forecasted Flexibility data | M24 –M25 |
| 8 | Display Forecasted Flexibility information | Retrieve and display Forecasted Flexibility information | Display Forecasted Flexibility data | M24 –M25 |
| 9 | Handle missing DR Signals information | Handle failures while retrieving DR Signals information | Inform user for lack of DR Signals data | M24 –M25 |

[3] https://jestjs.io/

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 10 | Display DR Signals | Display DR signals and information such as their status, time period, participating FEIDs etc. | Display DR Signals data | M24 –M25 |
| 11 | Handle missing Bids information | Handle failures while retrieving Bids information | Inform user for lack of Bids data | M24 –M25 |
| 12 | Display Bids information | Display Bids and information such as time period, responses, status etc. | Display Bids data | M24 –M25 |
| 13 | Handle missing Rewards information | Handle failures while retrieving Rewards information | Inform user for lack of Rewards data | M24 –M25 |
| 14 | Display Rewards information | Display available Rewards | Display Rewards data | M24 –M25 |
| 15 | Handle missing Energy price Profiling information | Handle failures while retrieving Energy price Profiling information | Inform user for lack of Energy price Profiling data | M24 –M25 |
| 16 | Display Energy price Profiling information | Display Energy price Profiling | Display Energy price Profiling data | M24 –M25 |
| 17 | Handle missing DVN Clusters information | Handle failures while retrieving DVN Clusters information | Inform user for lack of DVN Clusters data | M24 –M25 |
| 18 | Display DVN Clusters information | Display current DVN Clusters and their features | Display DVN Clusters data | M24 –M25 |
| 19 | Handle missing Node Profiling information | Handle failures while retrieving Node Profiling information | Inform user for lack of Node Profiling data | M24 –M25 |
| 20 | Display Node Profiling information | Display Node Profiles | Display Node Profiling data | M24 –M25 |
| 21 | Handle missing Aggregated Profiling information | Handle failures while retrieving Aggregated Profiling information | Inform user for lack of Aggregated Profiling data | M24 –M25 |

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 22 | Display Aggregated Profiling information | Display Aggregated Profiles | Display Aggregated Profiling data | M24 –M25 |

Customer Level:

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Handle missing Rewards information | Handle failures while retrieving Customer's Rewards | Inform user for lack of Customer's Rewards data | M24 –M25 |
| 2 | Display Rewards information | Display Customer's current Rewards | Display Customer's data for Rewards up to date | M24 –M25 |
| 3 | Handle missing DR Signals information | Handle failures while retrieving DR Signals that Customer participated/declined | Inform user for lack of DR Signals data | M24 –M25 |
| 4 | Display DR Signals information | Display DR Signals that Customer participated/declined | Display data for DR Signals where Customer participated or declined | M24 –M25 |
| 5 | Handle missing FEID Energy Profile information | Handle failures while retrieving Customer's FEID Energy Profile | Inform user for lack of FEID Energy Profile data | M24 –M25 |
| 6 | Display FEID Energy Profile information | Display Customer's FEID Energy Profile | Display Customer's FEID Energy Profile data | M24 –M25 |

### *4.4.2 Award –enabled Energy Behavioural Platform*

#### *4.4.2.1 Unit Testing*

The component will be evaluated using when completed using the jestjs tool.

#### *4.4.2.2 Functional Testing*

Upon new releases, all previously performed tests (see Section 3.4) will be repeated to ensure sustainable performance. Additional tests will also be defined upon completion of the gamified strategies design. All tests will be included in D6.4 which due M32.

### *4.4.3 Social Interaction and Cooperation Platform*

#### *4.4.3.1 Unit Testing*

The component will be evaluated when completed using the jestjs tool.

#### *4.4.3.2 Functional Testing*

Upon new releases, all previously performed tests (see Section 3.4) will be repeated to ensure sustainable performance. Additional tests will also be defined upon completion of the gamified strategies design. All tests will be included in D6.4 which due M32.

## 4.5 Common Information Modelling

#### *4.5.1.1 Unit Testing*

As previously stated in section 3, Unit tests do not provide added value for testing Semantic Interoperability. Due to this reason no Unit tests will be performed in future

#### *4.5.1.2 Functional Testing*

In the following period, we aim at testing the CIM to pass all the previous tests that the CIM failed, reported below. Additionally, since currently DELTA is in touch with the OpenADR Alliance Group other functional tests will be derived to fully cover such standard in DELTA.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Syntactic and Semantic Interoperability using JSON-LD and SAREF4ENER | Test that the messages transmitted in Test 1 have the proper format (JSON-LD) and use the SAREF4ENER Ontology. | Step 1: A PASS in the communication layer is verified.<br>Step 2: Validate that the data received have the proper format (Syntactic Interoperability).<br>Step 3: Validate that the data received have the proper model by means of the DELTA SHACL Shapes (Semantic Interoperability).<br>Step 4: Output test verdict. | June, 2020 (M26) |
| 2 | Technical Interoperability using XML and the model of OpenADR standard | Test the correct interaction between the FEID and the DVN by sending and correctly receiving packages of data, payloads are expressed in XML with OpenADR | Step 1: All other links are considered fully operational.<br>Step 2: Send 100 messages.<br>Step 3: Evaluate receipt of 100 messages.<br>Step 4: Validate integrity of received messages.<br>Step 5: Output test verdict. | June, 2020 (M26) |

## 4.6 Cybersecurity Services

### *4.6.1 DELTA Blockchain*

#### *4.6.1.1 Unit Testing*

Further unit testing will be performed if needed given the updates in the component's development.

#### *4.6.1.2 Functional Testing*

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Identity enrollment | A new identity is registered for a FEID and the FEID issues a CSR request | Certificates are generated and are communicated to the FEID along with private keys | M25-M28 |
| 2 | Identity revocation | The certificates for a FEID, peer, installer, etc. identity are revoked | The node with the revoked certificates cannot communicate to peers | M25-M28 |
| 3 | Certificates distribution | The signing certificate for a new node is issued | This certificate is communicated to all existing endpoints including peers, ordering services and Aggregator's components. The new node gets all certificates issued up to this point. | M25-M28 |
| 4 | CRL update | A certificate is revoked | The CRL of all existing nodes, like peers, ordering services and Aggregator's components is updated | M25-M28 |

### *4.6.2 Smart Contracts*

#### *4.6.2.1 Unit/Functional Testing*

Following the same approach as performed up to this point unit testing will also cover functional testing for this component. Previous tests will be repeated if needed for new contracts whereas testing will follow in regards to the Smart Contract Gateway.

### *4.6.3 Threat Mitigation*

#### *4.6.3.1 Unit Testing*

Unit testing are performed once the threat mitigation mechanisms will be developed and integrated with the other HW-SW DELTA components

#### *4.6.3.2 Functional Testing*

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Communication s | Attacks to the communication channels | VPN channels instauration/destruction | M28-M32 |
| 2 | Certification Authority | Disruption/attack to the Certification Authority infrastructure(s) | DoS against C.A, certificates forgery/destruction of local DELTA entity storage DB | M28-M32 |
| 3 | Protocols | Attacks/Disruptio n targeted to the communication protocols used by DELTA entities (real and virtual owners) | TCP/IP/UDP spoofing, DoS, reply attack, impersonation | M28-M32 |
| 4 | Databases | Disruption/alterati on of the local DB | DoS, spoofing/tampering | M28-M32 |

# 5. Integration Testing – Preliminary Results

## 5.1 DELTA Customer

In the following table the integration tests in general for components within the FEID are presented.

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Communication with Installer App | The technician that installs FEID in the infrastructure uses the Installer App in order to connect with FEID and pass the initial parameters | • The Installer App connects to FEID<br>• The initial parameters are inserted in FEID | The installation procedure was completed successfully |
| 2 | Smart meters data acquisition | FEID must gather real-time measurements from all the smart meters of the infrastructure | • Connection with the smart meter<br>• Collect the measurements in specific time intervals | FEID successfully connects to smart meters and acquires data in predefined time intervals |
| 3 | Device direct control | FEID must be able to control the devices that are directly connected with them | Apply an action to device | FEID successfully controls the devices that are directly connected with it |
| 4 | Registration to DELTA aggregator | As the installation of FEID has been completed, the registration information should be sent to the aggregator | Aggregator receives the appropriate information related to FEID registration | Aggregator successfully receives the registration information BUT the data are not transferred with the proper format (plain JSON instead of JSON-LD) |
| 5 | Forward real-time measurements to DELTA DVN | FEID must send real-time measurements gathered from smart meters to DVN | Broadcast the real-time measurements to DVN:<br>• In specific time interval<br>• With specific format | FEID sends successfully real-time measurements to DVN in 1 minute time interval and with JSON-LD format |
| 6 | Forward predicted values to DELTA DVN | FEID must perform forecasting algorithms to predict the day ahead operation and send these data to DVN | Broadcast the predicted values to DVN:<br>• In specific time interval<br>• With specific format | FEID successfully produces forecasts for day ahead at midnight and sends them with the proper format (JSON-LD) to DVN |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 7 | Receive Demand Response message from DELTA DVN | FEID should be able to receive any type of Demand Response messages from DVN | Receive DR message formatted in JSON-LD | FEID successfully receives DR messages |
| 8 | Inform Blockchain for Demand Response participation | FEID must inform in any case the Blockchain system for its upcoming DR participation | Blockchain system receives acceptance or rejection messages from FEID | FEID successfully informs the Blockchain system |
| 9 | Respond to Demand Response message to DELTA DVN | FEID must inform in any case the DVN system for its upcoming DR participation | DVN receives acceptances or rejection messages from FEID | FEID successfully informs the DVN |
| 10 | Registration to DELTA aggregator | As the installation of FEID has been completed, the registration information should be sent to the aggregator | Aggregator receives registration information with proper format (JSON-LD) | FEID Successfully registers to the Aggregator |

## 5.2 DELTA Virtual Node

In the following table the integration tests in general for components within the DVN are presented.

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Retrieve and store FEID profile data and measurements among DVN components | Through the Consumer/Prosumer Flexibility Data Monitoring and Profiling data originating from the FEIDs is circulated in real-time in components that require so through dedicated endpoints while also storing them in the DVN local repository | Data integrity on all endpoints in adequate execution times | Pass |
| | | | Data integrity on stored data on local DVN repository – fast and reliable data exchange with the PostgreSQL instance | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 2 | Execute on time forecasting sub-components | Either for day-ahead (long-term) or 1 hour-ahead (short-term) the forecasting tools should be executed on time and fast enough to produce results that can be used by other components | On-time execution<br>Fast execution in terms of time | Pass |
| 3 | Monitor effectively incoming and outgoing OpenADR compliant DR signals | Effective monitoring and logging of incoming and outgoing DR requests with proper data parsing and handling with the JSON-LD format and a payload compliant with the DELTA ontology and the OpenADR ontology | No packets loss in incoming / outgoing DR requests | Partial Fail – additional health check e.g. SHA/md5 checksum need to be added |
| | | | Compliant Payload handling (technical and semantically) | Pass |
| 4 | Execute on time optimal dispatch with correct FEID information | Upon incoming DR request from the Aggregator, the DVN deploys the Optimal Dispatch to identify how to break down the DR to each FEID. At this point, all available FEIDs are provided to the Optimal Dispatch component to select where to assign the DRs | On-time execution<br>Fast execution time | Pass |
| | | | Assess all provided FEIDs | Pass |
| | | | Successfully select a list of FEID based on their flexibility to assign DR requests | Fail – the OptiDVN selects all FEIDs to execute even a part of the DR, which may not be the optimal solution business wise. |
| | | | Maintain the initial target and reward requested by the Aggregator to the DVN in full while breaking down the initial DR | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 5 | Execute on time intra matchmaking based on DR failure information | When a FEID fails to deliver the requested amount of power for any given reason, the intra matchmaking is the one that is executed first to cover that failure. The matchmaking process needs to be quite fast in order to provide a solution that will ensure the overall DR to be successful on time. | On-time execution Fast execution time | Pass |
| 6 | Feed properly non-used FEIDs to the matchmaking process | When the Optimal Dispatch selects to which FEIDs the DR will be divided to, some FEIDs are not used. These (As well as others that become available in time) are eligible for the matchmaking process. | Provide correctly the available FEID information to the Matchmaking process | Pass |
| 7 | Provide robust endpoints for Aggregator and FEIDs to post/get data from the DVN | Assess all endpoints created for accessing information from / to the DVN | Robust Communication – No package lost under various scenarios | Pass |
| | | | Data Integrity under various scenarios | Pass |
| 8 | Ensure efficient and secure data storage for all local components | All local components should have secure access following basic authorization through a local endpoint with the DVN local repository | 24/7/365 PostgreSQL uptime | Pass |
| | | | Automated backups to avoid data loss | Pass |
| | | | Access only through Basic Authorization through local endpoint | Fail – needs for external development and testing has prevented adequately testing this feature, even though supported |

## 5.3 DELTA Aggregator

In the following table the integration tests in general for components within the Aggregator (including also the communication with the DELTA Grid State Simulation component) are presented.

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | External Load Dispatch and ToU DR requests serviced | Examine the potential to Service Successful DRs. | Statistical Measurements about the efficiency of DRs with regard to DR completion. | Pass |
| 2 | Retrieve and store DVN profile data and measurements | Through the Node Flexibility Monitoring and Profiling data component Aggregator collects data about the DVNs | Data integrity on all endpoints in adequate execution times | Pass |
| 3 | Monitor effectively incoming and outgoing OpenADR compliant DR signals | Effective monitoring and logging of incoming and outgoing DR requests with proper data parsing and handling with the JSON-LD format and a payload compliant with the DELTA ontology and the OpenADR ontology | Compliant Payload handling (technical and semantically) | Pass |
| 4 | Execute on time Segmentation task in order to redistribute DVNs and assign a new FEID to a DVN | Segmentation task needs to distribute all available FEIDs and formulate fairly shared DVNs | Evaluate Segmentation task processing time and efficient distribution | Pass |
| 5 | Provide robust endpoints for DVNs and FEIDs to communicate with the aggregator | Assess all endpoints created for accessing information from / to the Aggregator | Robust Communication and Data Integrity under various scenarios | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 6 | Generate optimized DR signals in accepted time intervals | DR Signals need to be generated on time in order to be serviced from DVNs | Assess the Processing time needed to generate Complex DR Signals | Pass |

## 5.4 Horizontal Services

In the following table the integration tests in general for horizontal components, such as CIM, Engagement tools, and cybersecurity services are presented.

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | Store on blockchain all DR related DVN transactions | DR Transactions must be stored to Blockchain | Successfully completed transactions are stored to Blockchain | Pass |
| | | | Failed Transactions are handled e.g. by retrying or marked as failed | Fail – Failed transactions or neither marked as failed nor specially handled |
| 2 | Visualise DR Signals and their status | Display DR Signals and their outcome e.g. completed, failed, pending | DR Signals and participating FEIDs as also outcome should be displayed | Pass |
| 3 | As a Customer I should be able to handle my FEID Devices from the UI | Handle FEID Dev ices | Customer can view FEID Devices and interact with them via the UI e.g. turn them on/off | Pass |
| 4 | As a Customer I should be able to respond to upcoming DR Requests | Customer can view upcoming DR Requests and decide whether to participate or not | Customer can approve/reject DR Requests and FEID/DVN/Aggregator respect the decision | Pass |
| 5 | Reward Customers based on their participation to DR Requests | Customers that accept and participate in DR Requests should be rewarded according to relative Game Rules | Relative game rules rewards are appointed to participating Customers | Pass |

# 6. Integration Testing – Plan

## 6.1 DELTA Customer

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Complete Integration with the DELTA CIM | An instance of the DELTA CIM will be installed on the FEID (FEID CIM) towards supporting complete communication through the CIM and OpenADR / DELTA Ontology | Complete communication with other DELTA layers through the CIM. Technical and Semantic interoperability Achieved | M24 – M25 |
| 2 | Connection to BMS | FEID should be able to connect to multiple type BMS | Connection with BMS Collect energy related measurements Control devices connected to BMS | M25-M27 |

## 6.2 DELTA Virtual Node

Upon new integrated releases, all previously performed tests (see Section 5.2) will be repeated to ensure sustainable performance. Additional tests will also be executed as described below. All tests will be included in D6.4 which due M32.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Complete Integration with the DELTA CIM | An instance of the DELTA CIM will be installed on the DVN (DVN CIM) towards supporting complete communication through the CIM and OpenADR / DELTA Ontology | Complete communication with other DELTA layers through the CIM. Technical and Semantic interoperability Achieved | M24 – M25 |
| 2 | Execute on time FEID clustering | Intra clustering should be executed on time and fast enough to produce results | On-time execution Fast execution in terms of time | M25 – M26 |
| 3 | Assess Optimal Dispatch and Inter/Intra Matchmaking with FEID clusters | Optimal Dispatch should take in account FEID clusters pre-computed by Clustering | Optimal Dispatch generates DR Signals for FEIDs in specific cluster(s) | M25 – M27 |

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 4 | Assess transition from intra to inter matchmaking | Matchmaking should be able to transition from intra to inter when intra can not provide a solution | Whenever intra matchmaking fails to provide a solution inter matchmaking should be executed | M25 – M27 |
| 5 | Execute on time inter matchmaking based on DR failure information | DR failures should trigger inter matchmaking execution | On-event execution | M25 – M26 |
| 6 | Assess stored information from other DVNs to be used within matchmaking | Share information among DVNs needed for inter matchmaking | DVNs information used by inter matchmaking is stored in all DVNs | M25 – M26 |
| 7 | Assess Optimal Dispatch results for both energy related and social related clusters | Optimal Dispatch results should be targeted to energy related or social related clusters | Optimal Dispatch results should aim energy related or social related clusters depending on business objective | M25 – M28 |

## 6.3 DELTA Aggregator

Upon new integrated releases, all previously performed tests (see Section 5.3) will be repeated to ensure sustainable performance. Additional tests will also be executed as described below. All tests will be included in D6.4 which due M32.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Complete Integration with the DELTA CIM | An instance of the DELTA CIM will be installed on the DVN (DVN CIM) towards supporting complete communication through the CIM and OpenADR / DELTA Ontology | Complete communication with other DELTA layers through the CIM. Technical and Semantic interoperability Achieved | M24-M25 |

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 2 | External Load Dispatch and ToU DR requests serviced | Examine the potential to Service Successful DRs. | Statistical Measurements about the efficiency of DRs with regard to DR completion. | M24-M26 |
| 3 | Aggregator Flexibility Aggregation Performance | Measure the Performance of the Aggregator. | Execution Time and Accuracy | M25-M27 |
| 4 | Ensure Fairness and Reliability for DVN DR requests | Measure the Fairness and Reliability acceptance. | Check that Fairness and Reliability indicators have accepted values | M25-M27 |
| 5 | Imbalance Market clearance executed | The Imbalance market’s clearance completion ability. | Evaluate the efficiency of the Aggregator to achieve a Market Clearance. | M25-M26 |
| 6 | Imbalance market bidding times met | Examine the capability of Aggregator to react in accepted time period | Check that the Aggregator’s reaction meets the bidding time requirements. | M25-M26 |
| 7 | Day ahead Market Prices Retrieval | Examine the Capability to parse Day ahead Market Prices | Ensure that the Day ahead Market Prices have been retrieved successfully. | M25-M26 |
| 8 | Self-Portfolio Day ahead Scheduling | Examine the Capability of the aggregator to schedule Day ahead Self-Portfolio efficiently | Validate that all the resources from the Self-Portfolio have been exploited. | M25-M26 |

## 6.4 Horizontal Services

Upon new integrated releases, all previously performed tests (see Section 5.4) will be repeated to ensure sustainable performance. Additional tests will also be executed as described below. All tests will be included in D6.4 which due M32.

| No | Test | Description | Evaluation criteria | Execution Dates |
|---|---|---|---|---|
| 1 | Performance of extra smart contracts (if any) | In case more smart contracts are needed then their performance should be evaluated in regards to transactions between DELTA layers | See sections 3.6 and 4.6 | M26-M30 |
| 2 | Large Scale UI Testing in terms of Customers | All the UIs developed in the context of DELTA need to be evaluated in real-life scaling for an Aggregator's Portfolio | All tests performed so far for a significantly larger portfolio | M28-M30 |
| 3 | Large Scale Testing for Gamified Services | All the gamified services developed in the context of DELTA need to be evaluated in real-life scaling for an Aggregator's Portfolio | All tests performed so far for a significantly larger portfolio | M28-M30 |
| 4 | Large Scale Testing for Collaboration Services | All the Collaboration Services developed in the context of DELTA need to be evaluated in real-life scaling for an Aggregator's Portfolio | All tests performed so far for a significantly larger portfolio | M28-M30 |
| 5 | Visual Analytics Effectiveness for segments /clusters | The radial tree visual representation should be linked with more information from multiple components and act as the key, easy to operate, navigation tool for the aggregator | Evaluation criteria will be concluded upon completion of all related functionalities | M28-M30 |

# 7. System Testing at Lab environment

## 7.1 Smart Home Testing Scenarios

One of the core testing facilities of the integrated DELTA framework, especially before the real-life demonstration at the pilot premises, is the CERTH/ITI Smart House.

**Figure 16: CERTH/ITI Smart House Testbed**

In terms of equipment three FEIDs have been deployed in the premises, two of which responsible for dedicated offices (each), and connected a smart meter and having control access to the lights through direct relays. The third FEID had access to two smart meters and multiple devices either through the embedded relay or through the Smart House BMS, or even with direct Modbus TCP/IP for generation and storage assets.

**Figure 17: Two FEID v1 (left) have been deployed to monitor lab “consumers” and one FEID v2 (right) has been deployed to monitor a lab “prosumer”**

To support the efficient evaluation and validation of the DELTA framework, multiple end-to end tests have been deployed and are still ongoing at the CERTH/ITI Smart House premises. The list of all the tests that have been executed so far is presented in the following table.

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 1 | End-to-End Load Dispatch DR with Specific Power Setpoint (Explicit Customers) | Incoming DR in the form of a Load Dispatch Power Setpoint from a higher entity than the Aggregator was send to the framework including all three basic layers, cybersecurity services and functional UIs, gamification and collaborative services – only explicit DR applied | Successful service of the DR request by applying Direct Load Control to available FEIDs | Partial Pass – A lot of failures were observed mainly due to the accuracy of the flexibility engine. Further testing is needed. The end-to-end procedure was successful. |
| 2 | End-to-End Load Dispatch DR with Specific Power Setpoint (Implicit Customers) | Incoming DR in the form of a Load Dispatch Power Setpoint from a higher entity than the Aggregator was send to the framework including all three basic layers, cybersecurity services and functional UIs, gamification and collaborative services – only implicit DR applied | Successful service of the DR request by informing end users, and letting them make changes to their systems. | Partial Pass – A lot of failures were observed mainly due to the accuracy of the flexibility engine. Further testing is needed. The end-to-end procedure was successful. |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 3 | End-to-End Load Dispatch DR with Specific Power Setpoint (Both Explicit and Implicit Customers) | Incoming DR in the form of a Load Dispatch Power Setpoint from a higher entity than the Aggregator was send to the framework including all three basic layers, cybersecurity services and functional UIs, gamification and collaborative services – both implicit and explicit DR applied | Successful service of the DR request by applying Direct Load Control to available explicit customers (FEIDs) and informing implicit customers (FEIDs), and letting them make changes to their systems. | Partial Pass – A lot of failures were observed mainly due to the accuracy of the flexibility engine. Further testing is needed. The end-to-end procedure was successful. |
| 4 | End-to-End ToU DR with Specific Pricing slots (Implicit Customers) | Incoming DR in the form of a Time of Use pricing scheme from a higher entity than the Aggregator was send to the framework including all three basic layers, cybersecurity services and functional UIs, gamification and collaborative services | All FEIDs/ customers informed about the change in their pricing scheme. For explicit DR customers the OptiDVN transformed the ToU to Load Dispatch and applied the respective DR | Pass (absent specific target set) |
| 4 | Addition of a new Customer/FEID | Based on the respective UC | Successful addition of a new customer/FEID | Pass |
| 5 | Deletion of a Customer | Based on the respective UC | Successful deletion of a new customer/FEID | Pass – remains unclear of whether should be kept anonymized or not. |
| 6 | Addition of a device / asset to the FEID | Following the procedure that needs to be handled by the installer | Successful addition of a device / asset through the installer UI and robust communication for monitoring and control from both the FEID and the Customer | Pass |

| No | Test | Description | Evaluation criteria | Results |
|---|---|---|---|---|
| 7 | Update of User Preferences (Comfort, DR availability, etc.) | Customer can change their preferences either in terms of comfort or DR availability, etc. | Successful change of user preferences from the customer UI and update on user / FEID profile on all required layers | Pass |

Testing at the CERTH/ITI Smart House premises is ongoing and will continue for several months and in parallel with the actual pilot deployment to ensure that all envisioned capabilities of the DELTA framework are delivered in full. A full testing suite will be included in D6.4, explaining in detail both currently on going and future end-to-end evaluation scenarios.

## 7.2 KIWI Testing Scenarios

In order to ensure that everything operates as should, an extended lab testing was agreed to occur prior to the pilot deployment at KIWI premises. Towards that direction, a FEID v2 was sent as soon as it was ready to KIWI headquarters to be integrated with the FRUIT and to be tested by KIWI experts.

So far, testing at KIWI premises has covered the Static Frequency support scenario, under which – and based on UK regulations, in any deviation below 49.7Hz or above 50.3 Hz should trigger a relay and change the operational status of a device linked to that relay. Due to the critical operation of such assets from KIWI perspective, a FRUIT will be also deployed as intermediate device to ensure that in case of a FEID failure the expected action will be ensured within the expected timeframe (i.e. 30 minutes).

All performed tests, through Modbus RTU, have led to a successful outcome in terms for FEID – FRUIT integration and Static Frequency Support services from the FEID.

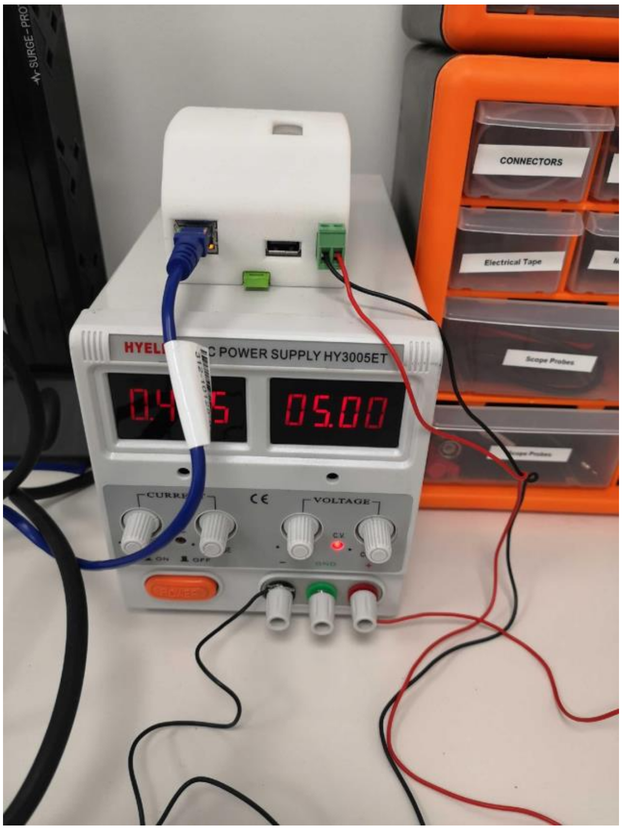


**Figure 18: FEID v2 at KIWI for testing the Frequency Support functionality and integration with the KIWI fruit.**

## 7.3 JRC TestBed Testing Scenarios

There haven't been any tested performed until the examined period at the JRC Testbed. Mainly due to the corona virus outbreak, as the testing facilities are located in Italy. When available, at least one FEID will be deployed to integrate that JRC testbed as an additional customer to the testing DELTA network for evaluating more complex scenarios.

# 8. Conclusions

Following the implementation phase of all DELTA components, it is necessary to evaluate their performance both individually and in an integrated system. One of the core objectives of the present deliverable was to establish a common methodology under which the evaluation of each individual component will be thoroughly tested and its envisioned functionalities will be validated, followed by the respective integrated scenarios per layer (Customer, DVN, Aggregator, Cybersecurity, CIM, and Customer Engagement Tools) and for the entire DELTA framework.

Building upon the methodology presented, all the evaluation steps followed and concluded up to M24 are presented in detail, elaborating more on the unit and functional testing per component, followed by the respective evaluation plan for the remaining period. Subsequently, the integration tests per layer as executed up to M24 are also delivered, revealing also future evaluation activities that will ensure the sustainability and the envisioned functionalities of the DELTA framework.

Beyond evaluating the architectural components at lab environment, it is imperative to demonstrate their effectiveness under real-life conditions, to assess their actual performance through unexpected situations and mitigate accordingly any introduced challenged. To that end, all integrated components, both hardware and software, have been deployed at the CERTH/ITI Smart House and have been tested under real-life conditions. Testing results so far demonstrated promising results with a very mature development progress of most individual components, however further refinement is needed to reach the project objectives.

To facilitate further evaluation documentation all future tests, evaluation and validation activities will be documented in D6.4 which is expected on M32.

# References

[1] https://ec.europa.eu/energy/sites/ener/files/documents/xpert_group1_reference_architecture.pdf

[2] ISC Committee. (1990). IEEE Standard Glossary of Software Engineering Terminology (IEEE Std 610.12-1990). Los Alamitos. CA IEEE Comput. Soc.

[3] Gyrard, A., Datta, S. K., & Bonnet, C. (2018, February). A survey and analysis of ontology-based software tools for semantic interoperability in IoT and WoT landscapes. In 2018 IEEE 4th World Forum on Internet of Things (WF-IoT) (pp. 86-91). IEEE.

[4] van der Veer, H., & Wiles, A. (2008). Achieving technical interoperability. European telecommunications standards institute.

[5] Tolk, A. (2013, October). Interoperability, composability, and their implications for distributed simulation: Towards mathematical foundations of simulation interoperability. In 2013 IEEE/ACM 17th International Symposium on Distributed Simulation and Real Time Applications (pp. 3-9). IEEE

[6] Ouksel, A. M., & Sheth, A. (1999). Semantic interoperability in global information systems. ACM Sigmod Record, 28(1), 5-12.

[7] Alliance, O. (2013). OpenADR 2.0 b profile specification. Raportti. OpenADR Alliance.