| Title: | Document Version: |
|---|---|
| D7.2 WiseCOOP and WiseCORP Apps implementation and lab-testing | 0.5 |

| Project Number: | Project Acronym: | Project Title: |
|---|---|---|
| H2020-731205 | WiseGRID | Wide scale demonstration of Integrated Solutions for European Smart Grid |

| Contractual Delivery Date: | Actual Delivery Date: | Deliverable Type*-Security*: |
|---|---|---|
| M21 (July 2018) | M21 (July 2018) | RD-PU |

*Type: P: Prototype; R: Report; D: Demonstrator; O: Other.

**Security Class: PU: Public; PP: Restricted to other programme participants (including the Commission); RE: Restricted to a group defined by the consortium (including the Commission); CO: Confidential, only for members of the consortium (including the Commission).

| Responsible: | Organisation: | Contributing WP: |
|---|---|---|
| Alberto Zambrano | ETRA | WP7 |
| Álvaro Nofuentes | ETRA | WP7 |


**Authors (organization):**

Álvaro Nofuentes (ETRA), Alberto Zambrano (ETRA), Germán Martínez (ETRA), Julio César Díaz (ITE), Amparo Mocholí (ITE), Pascual Mullor (ITE), Antonis Papanikolaou (HYP), Benjamin Kraft (VS), Leandro Lombardo (ENG), Alexandre Lapedra (BYES), Vincent Dierickx (ECO-EID), Ine Swennen (ECO-EID), Foivos Palaiogiannis (ICCS)



**Abstract:**

This document reports the work performed within Task 7.2 “WiseCOOP and WiseCORP Apps Implementation” and Task 7.3 “WiseCOOP and WiseCORP Apps lab-testing and refinement”, following the specifications and architecture designed in Task 7.1 and reported in D7.1.



**Keywords:**

Energy efficiency, Building, Cooperative, Facility manager, ESCO, Retailer, DR, Flexibility; Aggregator, Implementation, Integration, Lab-Testing

## Revision History

| Revision | Date | Description | Author (Organisation) |
|---|---|---|---|
| 0.1 | 22.06.2018 | ToC | Alberto Zambrano (ETRA) |
| 0.2 | 09.07.2018 | Sections 3.2 and 4.2. Integration of test templates | Alberto Zambrano (ETRA), Antonis Papanikolaou (HYP), Pascual Mullor (ITE) |
| 0.3 | 18.07.2018 | Sections 3 and 4 | Alberto Zambrano (ETRA) |
| 0.4 | 27.07.2018 | Last Contributions and ready for peer review | Alberto Zambrano |
| 0.5 | 02.08.2018 | Deliverable ready for Submission | Alberto Zambrano (ETRA), Álvaro Nofuentes (ETRA) |

INDEX

## List of figures

## List of Tables

## EXECUTIVE SUMMARY

WiseGRID will facilitate and promote active participation of consumers and prosumers by means of new market structures and innovative ICT services through the implementation of the WiseCOOP and WiseCORP tools.

WiseCOOP is the WiseGRID technological solution targeting Smart Grid actors interfacing consumers and prosumers (particularly focused on domestic and small businesses), supporting them in their roles of energy retailers, third-party aggregators, local communities and cooperatives. The main goal of the solution is helping consumers and prosumers to work together in order to achieve better energy deals while relieving them from administrative procedures and cumbersome research.

WiseCORP is the WiseGRID technological solution targeting businesses, industries, ESCOs and public facility consumers and prosumers, with the objective of providing them the necessary mechanisms to become smarter energy players. By means of energy usage monitoring and analysis, proper information can be given to facility managers helping them to reduce energy costs and environmental impact.

Taking into consideration the aim of these tools, the respective architecture and modules were designed also having in mind the design of the other WiseGRID tools. Furthermore, having in mind the process of developing software, this document explains the implementation and lab-testing activities performed for assuring the quality of the tools previously designed.

For that purpose, it was first needed to align the terminology to be used during the lab-testing phase in order to assure that all the partners involved in this stage, work in the same direction and there are no misunderstandings. Then, the test plan was established which basically consists in the following 5 steps:

1) Review the project requirements and use cases
2) Define the features to be tested from those and classify them into test groups
3) Detail test cases for validation of named features
4) Execute the test cases
5) Document the test protocols

Moreover, for assuring the coherence and the easy understanding of each test, the following template was created, which summarizes the most important information to be shown and the features to be tested.

**Table 1 – Test case specification template**

| | | | |
|---|---|---|---|
| **Name** | *The test case code and name which is unique to the project.* | | |
| **Module under test** | *The devices or systems under test* | **Resp.** | *Main partner responsible for the test* |
| **Module requirement** | *The requirement, use case, or certification rule which is validated by the test case* | | |
| **Test environment** | *List of elements needed for the test execution* | | |
| **Features to be tested** | *List of features to be tested* | | |
| **Features not to be tested** | *Optional* | | |
| **Preparation** | *Short list of steps needed for preparing the test environment for test execution* | | |

| | |
|---|---|
| **Dependencies** | *(Optional) List of test case codes defining test cases which need to be passed before the test case at hand can be started* |
| **Steps** | *Testing procedures* |
| **Pass criteria** | *Expected (measurable) results, allowing to unambiguously judge if the test is passed or not passed (i.e. the product requirement was validated or not validated)* |
| **Suspension criteria** | *(Optional) Conditions under which continuation of the test is considered pointless because testing results would be invalid* |
| **Results** | *(Optional) Short list of results* |

Anyway, before starting the lab-testing phase, the implementation activities were performed in order to correctly integrate all the modules of the tools.

The following tables below list all the test cases and test results covered in the document at hand:

**Table 2 – WiseCOOP tests results**

| Test case code | Test case | Test result |
|---|---|---|
| **RTM001** | Read smart meter data from IOP | ✓ |
| **RTM002** | Store smart meter data to long-term DB | ✓ |
| **KPI001** | Individual energy delta calculation | ✓ |
| **KPI002** | Portfolio data aggregation | ✓ |
| **KPI003** | Portfolio profiling, demand-related behaviour | ✓ |
| **KPI004** | Portfolio profiling, production-related behaviour | ✓ |
| **KPI005** | Portfolio profiling, monthly $CO_2$ emissions | ✓ |
| **KPI006** | Portfolio profiling, monthly cost | ✓ |
| **KPI007** | Calculation of average profiles per clusterization and cluster | ✓ |
| **FOR001** | Demand/production forecasting training | ✓ |
| **FOR002** | Demand/Production forecasting | ✓ |
| **FOR003** | Request message parsing test of WiseCOOP forecast module | ✓ |
| **FOR004** | Forecast response message generation test of WiseCOOP forecast module | ✓ |
| **FOR005** | Forecast is periodically triggered | ✓ |
| **FOR006** | Forecast results are saved to operational DB | ✓ |
| **TDC001** | Create a tariff | ✓ |
| **TDC002** | Edit existing tariff | ✓ |
| **DRF001** | Obtain portfolio demand forecast | ✓ |
| **DRF002** | Obtain generation forecast | ✓ |
| **DRF003** | Price signal calculation | ✓ |
| **DRF004** | Dispatch price signal | ✓ |
| **DRF005** | Elicitation of demand flexibility per building | ✓ |
| **DRF006** | Estimate & dispatch device commands | ✓ |

Table 3 – WiseCORP tests results

| Test case code | Test case | Test result |
|---|---|---|
| **RTM001** | Read smart meter data from IOP | ✓ |
| **RTM002** | Read sensor data from IOP | ✓ |
| **RTM003** | Read battery data from IOP | ✓ |
| **RTM004** | Read HVAC data from IOP | ✓ |
| **RTM005** | Store smart meter data to Long-term DB | ✓ |
| **RTM006** | Store sensor data to Long-term DB | ✓ |
| **RTM007** | Store battery data to long-term DB | ✓ |
| **RTM008** | Store HVAC data to Long-term DB | ✓ |
| **KPI001** | Associated $CO_2$ emissions | ✓ |
| **KPI002** | Associated economic costs | ✓ |
| **KPI003** | Distribution of demand per tariff period | ✓ |
| **KPI004** | Calculation of indicators per building. Monthly economic costs | ✓ |
| **KPI005** | Calculation of indicators per building. $CO_2$ emissions | ✓ |
| **KPI006** | Calculation of indicators per building. Total demand | ✓ |
| **KPI007** | Calculation of indicators per building. Total production | ✓ |
| **KPI008** | Calculation of indicators per building. Self-consumption ratio | ✓ |
| **FOR001** | Demand/production forecasting training | ✓ |
| **FOR002** | Demand/Production forecasting | ✓ |
| **FOR003** | Request message parsing test of WiseCORP forecast module | ✓ |
| **FOR004** | Forecast response message generation test of WiseCORP forecast module | ✓ |
| **FOR005** | Forecast is periodically triggered | ✓ |
| **FOR006** | Forecast results are saved to operational DB | ✓ |
| **TC001** | Create simulated bill for building | ✓ |
| **EUO001** | Unit testing | ✓ |
| **EUO002** | Produce day-ahead optimum schedule for assets | ✓ |
| **DRF001** | Estimate occupant thermal/visual comfort profile | ✓ |
| **DRF002** | Calculate human-centric demand flexibility of building | ✓ |
| **DRF003** | Receive request to activate demand flexibility | ✓ |
| **DRF004** | Asset schedule modification | ✓ |
| **AD001** | Load schedule from operational database | ✓ |
| **AD002** | Read current asset status from operational database | ✓ |
| **AD003** | Detect deviation from schedule | ✓ |
| **AD004** | Trigger asset setpoint | ✓ |

All these activities have been supported by the setup of a virtual environment which replicates to the extent possible the conditions that will be found in the deployment of the applications in the different pilot sites. The lab-testing platform consists of a couple of virtual machines running in the VMWare vSphere infrastructure of ETRA I+D.

# 1 INTRODUCTION

## 1.1 PURPOSE OF THE DOCUMENT

The purpose of this document is to summarise the results from Tasks 7.2 “WiseCOOP and WiseCORP implementation” and 7.3 “WiseCOOP and WiseCORP lab-testing and refinement”. In these tasks, the WiseCOOP and WiseCORP designs and developments within Task 7.1 “WiseCOOP and WiseCORP Apps design” is verified in a controlled environment before deployment at the pilot sites.

## 1.2 SCOPE OF THE DOCUMENT

This deliverable covers the development of the WiseCOOP and WiseCORP tools during their implementation and lab-testing phases, including an overview of the designed architectures in order to make the reader aware about the previous work performed. In this way, this document describes the test cases that were performed to validate the WiseCOOP and WiseCORP frameworks before deploying them at the pilot sites.

## 1.3 STRUCTURE OF THE DOCUMENT

The document starts with the establishment of the lab-testing basis that will be used for the evaluation of the test cases. Then, the document continues with the explanation of the implementation of the different WiseCOOP modules. After this implementation starts the lab-testing phase of WiseCOOP, which describes the different tests done to evaluate the performance of the tool and their results. After these sections there are similar ones for WiseCORP. Finally, a section for extracting the conclusions of these tasks closes the documents and settles the next steps to be followed.

# 2 LAB-TESTING APPROACH

The lab-testing approach that has been followed for these tools is the same one that was followed in the NOBEL GRID project. This methodology has demonstrated to be successful for this kind of projects so it has been properly studied and used taking into account the particularities of the WiseGRID project.

## 2.1 DEFINITION OF TERMS

In order to provide a common methodology for testing the WiseCOOP and WiseCORP tools, a common definition of terms was used. The following definitions were developed considering the state of the art in software, smart grid and system integration testing, especially with respect to the IEEE 829 Standard for software test documentation [1] [2] [3].

**Table 4 – Definition of terms**

| Term | Definition |
|---|---|
| **Device under test** | a product or software which is verified by a certain test case. It is part of the test environment |
| **Expected results** | a description of the status of the test environment after a test case was carried out and pass criteria have been met |
| **Features (not) to be tested** | a list of product requirements or specifications which are (not) covered by a certain test case |
| **Pass/fail criteria** | a definition of how to judge or measure if a product under test conforms to specifications and requirements that shall be validated by a certain test case |
| **Retesting** | re-execution of a test case that previously returned a "fail" result, to evaluate the effectiveness of intervening correction actions |
| **Subsystem acceptance criteria** | conditions to be fulfilled by a subsystem for including it into the system integration test. Conditions should include the availability of testing protocols for standalone subsystem tests. Also, subsystems should have similar level of maturity |
| **Suspension criteria** | a description of conditions which indicate that the test was carried out incorrectly or that any situation was produced which renders the testing results unusable, making test continuation pointless and requiring the test to be halted and restarted |
| **System integration test** | a test designed to verify that a system made up of two or more interacting products (subsystems) conforms to system-wide specifications and requirements. The device under test is the system itself. It is specially designed for finding inconsistencies which emerge only through the subsystem interaction. The system integration test plan may define partial system integration tests which allow for adding subsystems subsequently |
| **(System integration test) Level** | The number of system layers which are included in a system integration test case minus one |
| **System layer** | a group of one or more subsystems which is defined prior to the system integration test. According to group definition for a given system should be used for all system integration test cases |
| **Testing** | set of activities conducted to facilitate discovery, validation and/or evaluation of properties of one or more test WiseGRID components |
| **Test analysis** | elaboration about why a test result emerged. It may also include a conclusion about what the test result implies for the future work |
| **Test case** | a collection of features (not) to be tested, testing procedures, and pass/fail criteria used for testing a system or device under test. Test cases may refer to specific types of product requirements, e.g. the function, reliability, stability, safety, or vulnerability. Test cases may be applied to different test environments, e.g. the same test case may be applied to different pilot sites |
| **Test case code** | an identifier for a test case which is unique throughout the project, e.g. KPI001 |

| | |
|---|---|
| **Test case specification** | documentation of one or more test cases |
| **Test coverage** | a list of product requirements or specifications which are verified by a test plan |
| **Test data** | data created or selected which is needed for executing one or more test cases. It may be defined in the test case specification |
| **Test environment** | a list of all elements (software, hardware, information, external conditions) needed to carry out a test case, including the device or system under test and all elements needed to judge the test outcome |
| **Test environment set-up process** | a list of actions needed for establishing and maintaining a required test environment |
| **Test execution** | the actions needed to carry out the testing procedures for a given test case |
| **Test group** | a collection of test cases which share at least one defined criterion. E.g. all test cases which relate to cybersecurity testing might be defined to make up a test group |
| **Test method** | a general definition of testing procedures and test environment for a test plan |
| **Test plan** | a strategy or list of tasks used to verify that a product conforms to design specifications and product requirements |
| **Test preparation** | a definition of steps which are needed to prepare a test environment for test execution |
| **Testing procedures** | a specific list of steps which are needed to carry out a test case |
| **Test protocol** | a summary of the test results of all test cases defined in a test plan. It may also contain the test analysis for said test cases |
| **Test requirements** | a definition stating the status of the test environment which is needed for carrying out a specific test case or a test group. Ideally, it is also stated how it can be checked if the test environment is ready for test execution |
| **Test responsibilities** | a definition stating which persons or organizations are needed for the test. It may also include an assignment of tasks to those people or organizations |
| **Test result** | an indication of whether a specific test case has passed or failed. May also include any data that has been obtained through execution of the test case |

## 2.2 TEST PLAN

The test plan used for testing both applications consists of the following steps:

1) Review the project requirements and use cases
2) Define the features to be tested from those and classify them into test groups
3) Detail test cases for validation of named features
4) Execute the test cases
5) Document the test protocols

## 2.3 FEATURES TO BE TESTED

The partners have defined a number of features to be tested. Those features are based on the project requirements defined by the consortium and use cases for the WiseGRID project.

The features to be tested were classified into different test groups which are defined in Table 5. The table also defines which criteria are shared by the test cases within the test groups.

**Table 5 – Test groups**

| Test group | Common Criteria |
|---|---|
| **Visualisation and analysis** | The feature under test provides visualization and/or analysis of the data collected |
| **Control** | The feature under test provides control of assets. |

| | |
|---|---|
| **Compliance** | The feature under test relates to compliance of the tool with the USEF standard or other standards. |
| **Functionality** | The feature under test is a complex function provided by a combination of software and communication between multiple WiseGRID subsystems. |
| **Communication** | The feature under test is basic data transmission between two communication endpoints, one being the tool. |
| **Robustness and stability** | The feature under test is related to fault tolerant and stability. |
| **Cyber Security** | The feature under test mitigates vulnerabilities of the software or malicious attacks aimed at it. |

Table 6 and Table 7 show the features tested as defined for testing the WiseCOOP and WiseCORP. Each feature shall define one test case.

**Table 6 – WiseCOOP's features tested**

| Test case code | Feature to be tested | Test group |
|---|---|---|
| **RTM001** | Data from SMX is properly collected in the operational database of WiseCOOP | Communication |
| **RTM002** | Data from SMX is properly collected in the long-term database of WiseCOOP (big data) | Communication |
| **KPI001** | - Smart meters provide information of the total accumulated energy demand/production.<br>- The system therefore needs to calculate the energy deltas across consecutive readings in order to properly monitor the energy demand/production profiles.<br>- Three different aggregations of the deltas are considered: quarterly, hourly and daily. | Visualisation and analysis |
| **KPI002** | An overview of the demand and production metrics for the whole portfolio is required | Visualisation and analysis |
| **KPI003** | All portfolio members get classified based on the distribution of the energy demand in the following time slots:<br>- Working days, 08h-18h<br>- Working days, 18h-08h<br>- Non-working days, 08h-18h<br>Non-working days, 18h-08h | Visualisation and analysis |
| **KPI004** | All portfolio members get classified based on their daily production | Visualisation and analysis |
| **KPI005** | All portfolio members get classified based on the monthly equivalent CO2 emissions | Visualisation and analysis |
| **KPI006** | All portfolio members get classified based on the economic cost associated to their demand | Visualisation and analysis |
| **KPI007** | For each group of each clusterization, the *average member* is computed, in order to enable comparison of individual behaviour with the group of similar individuals | Visualisation and analysis |
| **FOR001** | WiseCOOP forecast module is trained | Functionality |
| **FOR002** | WiseCOOP forecast module performs demand/production forecasting training | Functionality |
| **FOR003** | Performance of WiseCOOP forecast module, at parsing forecast queries. | Functionality |
| **FOR004** | Performance of WiseCOOP forecast module, at generating and submitting | Functionality |

| | | |
|---|---|---|
| | the demand forecast response. | |
| **FOR005** | WiseCOOP periodically posts a demand and a production forecast request per bus to the corresponding queue of the internal ESB | Functionality |
| **FOR006** | WiseCOOP receives the results of the forecast module, formats them following the same format used to store real-time data, and stores the in the operational database | Functionality |
| **TDC001** | WiseCOOP facilitates to retailers the ability to define tariffs from the UI | Functionality |
| **TDC002** | WiseCOOP facilitates to retailers the ability to define tariffs from the UI | Functionality |
| **DRF001** | Retrieval of the aggregated day-ahead demand of the retailer portfolio from the long-term DB of the WiseCOOP tool | Communication |
| **DRF002** | Retrieval of the day-ahead generation forecast of the retailer portfolio from the long-term DB of the WiseCOOP tool | Communication |
| **DRF003** | Estimation of the day-ahead 24-hour retail electricity price forecast based on the availability of renewable generation | Visualisation and analysis |
| **DRF004** | Dispatching of the price signal (encoded in the format specified in D10.2) to a queue in the IOP so that other WiseGRID products can receive it. | Communication |
| **DRF005** | Elicitation and collection of the demand flexibility potential from the all the buildings that belong to the portfolio of the WiseCOOP operator | Visualisation and analysis |
| **DRF006** | Breakdown of demand flexibility requested from the DSO into the optimal flexibility per building device, based on the information retrieved about the demand flexibility potential. | Control |

**Table 7 – WiseCORP's features tested**

| Test case code | Feature to be tested | Test group |
|---|---|---|
| **RTM001** | Data from SMX is properly collected in the operational database of WiseCORP | Communication |
| **RTM002** | Data from sensor Wrapper is properly collected in the operational database of WiseCORP | Communication |
| **RTM003** | Data from battery is properly collected in the operational database of WiseCORP | Communication |
| **RTM004** | Data from HVAC is properly collected in the operational database of WiseCORP | Communication |
| **RTM005** | Data from SMX is properly collected in the long-term database of WiseCORP (big data) | Communication |
| **RTM006** | Data from sensor is properly collected in the long-term database of WiseCORP (big data) | Communication |
| **RTM007** | Data from battery is properly collected in the long-term database of WiseCORP (big data) | Communication |
| **RTM008** | Data from HVAC is properly collected in the long-term database of WiseCORP (big data) | Communication |
| **KPI001** | This module must crosscheck the energy mix information with the individual energy demand readouts in order to compute the equivalent $CO_2$ emissions | Visualisation and analysis |
| **KPI002** | This module must cross-check the energy price for the contracted tariff with the individual energy demand readouts in order to compute the associated economic costs | Visualisation and analysis |
| **KPI003** | This module must crosscheck the energy demand of the building with the periods of the contracted tariff, giving an overview of the distribution of | Visualisation and analysis |

| | | |
|---|---|---|
| | the demand over a period of time (e.g. monthly) | |
| **KPI004** | This module analyses historic information available in the long-term database to provide insights on the economic costs faced by the measured facilities | Visualisation and analysis |
| **KPI005** | This module analyses historic information available in the long-term database to provide insights on the equivalent $CO_2$ emissions produced by the measured facilities. | Visualisation and analysis |
| **KPI006** | This module analyses historic information available in the long-term database to provide insights on the total energy demand of the measured facilities. | Visualisation and analysis |
| **KPI007** | This module analyses historic information available in the long-term database to provide insights on the total energy production of the measured facilities. | Visualisation and analysis |
| **KPI008** | This module analyses historic information available in the long-term database to provide insights on the total self-consumption and the energy production surplus of the measured facilities | Visualisation and analysis |
| **FOR001** | WiseCORP forecast module is trained | Functionality |
| **FOR002** | WiseCORP forecast module performs demand/production forecasting training | Functionality |
| **FOR003** | Performance of WiseCORP forecast module, at parsing forecast queries. | Functionality |
| **FOR004** | Performance of WiseCORP forecast module, at generating and submitting the demand forecast response. | Functionality |
| **FOR005** | WiseCORP periodically posts a demand and a production forecast request per bus to the corresponding queue of the internal ESB | Functionality |
| **FOR006** | WiseCORP receives the results of the forecast module, formats them following the same format used to store real-time data, and stores the in the operational database | Functionality |
| **TC001** | WiseCORP facilitates to ESCOs the ability to simulate bills according to historical demand data and tariff definitions | Visualisation and analysis |
| **EUO001** | The energy usage optimizer must calculate the optimum 24-hours long schedule for the given assets, considering usage calendar and energy price. | Functionality |
| **EUO002** | Upon completion of the execution of the energy usage optimizer module, results are stored in the operational database of WiseCORP | Functionality |
| **DRF001** | Generation of the comfort profile for the individual occupants regarding thermal and visual comfort | Visualisation and analysis |
| **DRF002** | Estimation of demand flexibility time series | Functionality |
| **DRF003** | Reception of the appropriate message from WiseCOOP specifying the detailed break-down of demand flexibility per device to be activated | Communication |
| **DRF004** | Estimation of optimal setpoint per device and dispatch to the “asset dispatcher” component that sends the setpoints to the device wrappers. | Visualisation and analysis |
| **AD001** | The asset dispatcher module can read from operational database all necessary information about the schedule of the controllable assets | Communication |
| **AD002** | The asset dispatcher module can read from operational database all necessary information about the current setpoint executed by the controllable assets | Communication |
| **AD003** | Given a point in time when current setpoint and scheduled setpoint differs for a controllable asset, the asset dispatcher must be able to detect the incoherence | Control |
| **AD004** | Given a point in time when current setpoint and scheduled setpoint differs for a controllable asset, the detection of this incoherence must result in the publication of a command to the controllable asset to set the appropriate setpoint | Control |

## 2.4 TEST CASES SPECIFICATION

**Table 8 – Test case specification template**

| | | | |
|---|---|---|---|
| **Name** | *The test case code and name which is unique to the project.* | | |
| **Module under test** | *The devices or systems under test* | **Resp.** | *Main partner responsible for the test* |
| **Module requirement** | *The requirement, use case, or certification rule which is validated by the test case* | | |
| **Test environment** | *List of elements needed for the test execution* | | |
| **Features to be tested** | *List of features to be tested* | | |
| **Features not to be tested** | *Optional* | | |
| **Preparation** | *Short list of steps needed for preparing the test environment for test execution* | | |
| **Dependencies** | *(Optional) List of test case codes defining test cases which need to be passed before the test case at hand can be started* | | |
| **Steps** | *Testing procedures* | | |
| **Pass criteria** | *Expected (measurable) results, allowing to unambiguously judge if the test is passed or not passed (i.e. the product requirement was validated or not validated)* | | |
| **Suspension criteria** | *(Optional) Conditions under which continuation of the test is considered pointless because testing results would be invalid* | | |
| **Results** | *(Optional) Short list of results* | | |

## 2.5 LAB-TESTING PLATFORM DETAILS

The implementation and lab-testing phases of the development of WiseCOOP and WiseCORP applications has been supported by the setup of a virtual environment which replicates up to the extent possible the conditions that will be found in the deployment of the applications in the different pilot sites.

The lab-testing platform consists of a couple of virtual machines running in the VMWare vSphere infrastructure of ETRA I+D.

**Table 9 – Characteristics of the lab-testing platform servers**

| Characteristics | wisegridpre.lab.id | wintest.lab.id |
|---|---|---|
| **OS** | Ubuntu Server 16.04 | Microsoft Windows Server 2012 |
| **CPU** | 2 CPU | 1 CPU |
| **Memory** | 8GB | 5GB |
| **Hard disk** | 50GB | 35GB |

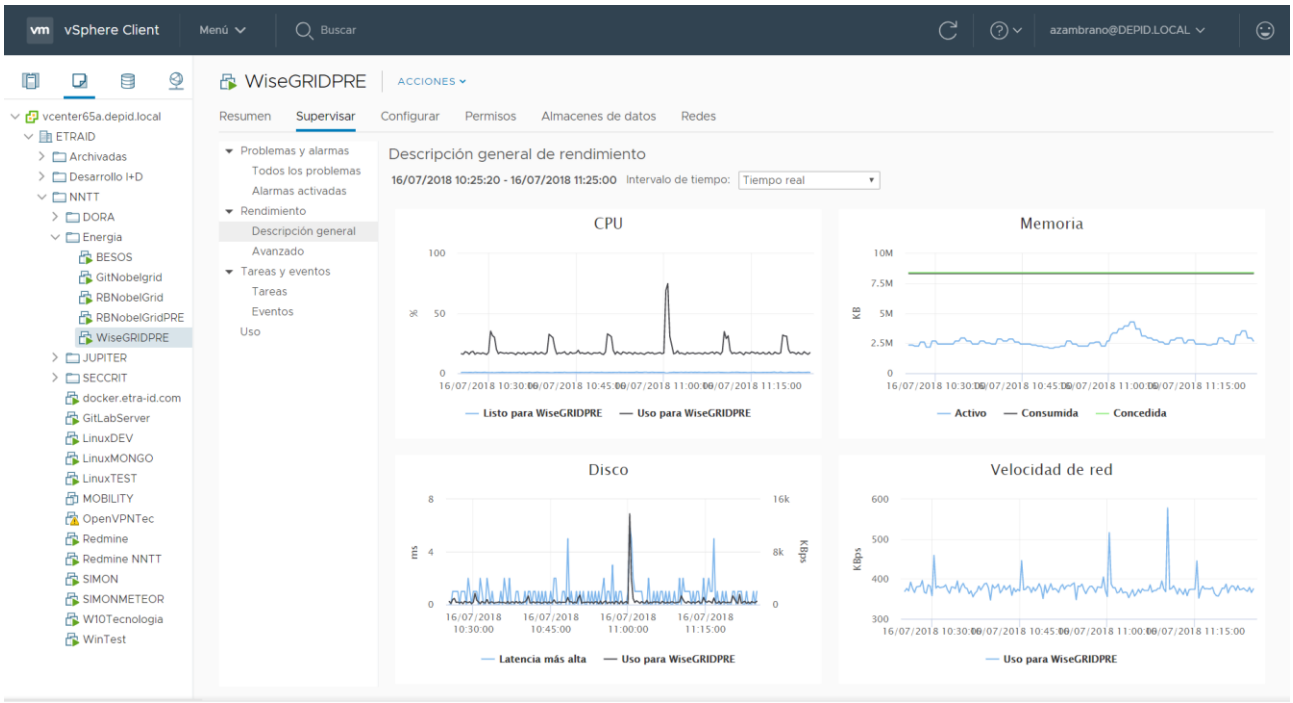



**Figure 1 – Screenshot of wisegridpre.lab.id server on vSphere platform**

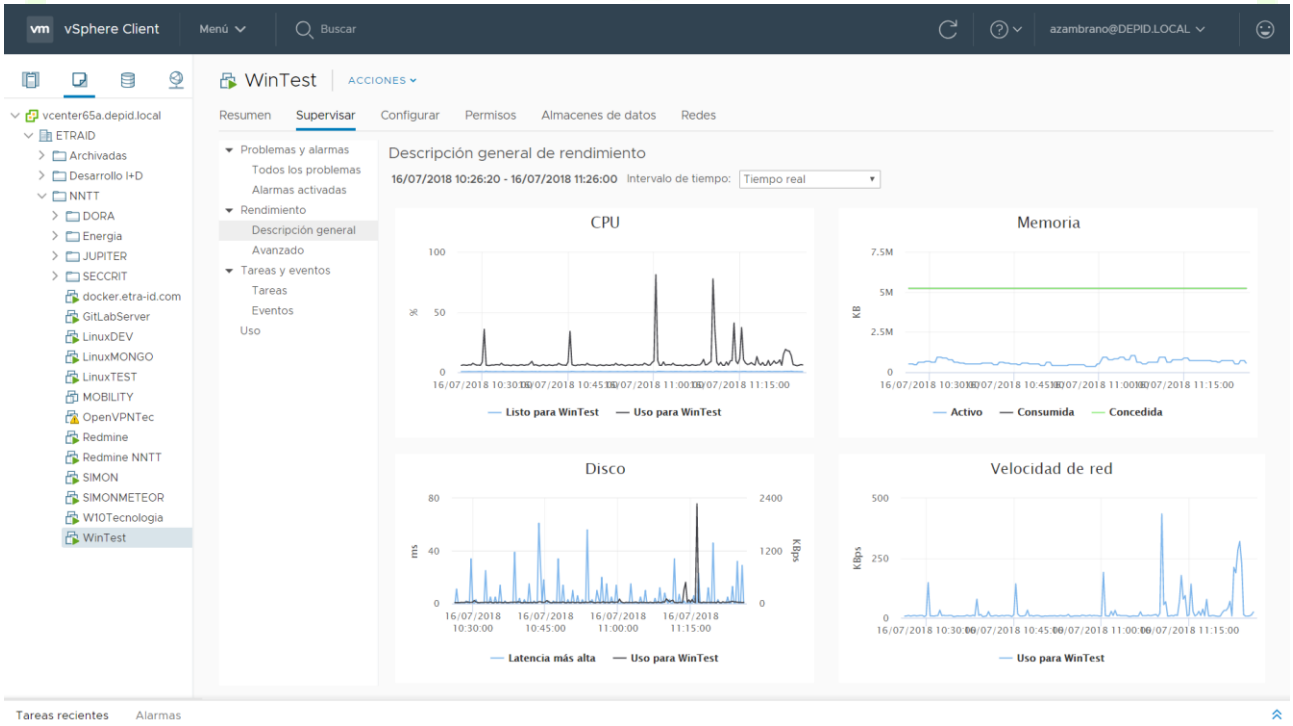



**Figure 2 – Screenshot of wisegridpre.lab.id server on vSphere platform**

Following the architecture of the tools, a number of common services have been installed on those servers and made accessible to the rest of the partners taking part in the project via public URL (protected with the

corresponding access credentials). Data protection principles have been also considered in line with deliverable D3.2.

**Table 10 – Server directions**

| Server | Module | URL |
|---|---|---|
| **wise-gridpre.lab.id** | Internal ESB (RabbitMQ).<br>Two virtual hosts:<br>- /wisecoop<br>- /wisecorp | AMQP: amqp://etra-id.com<br>MQTT: tcp://etra-id.com:1883 |
| | Database server (MongoDB) | mongodb://etra-id.com |
| | Big data processing (Spark Server – 1 master + 1 server) | [internal access only] |
| | WiseCOOP User Interface | https://wisecoop.etra-id.com |
| | WiseCORP User Interface | https://wisecorp.etra-id.com |
| **wintest.lab.id** | Maintenance management module | http://windeptec.etra-id.com/GiManWiseGRIDWSRest |

The described configuration, together with the lab-testing instance of the WiseGRID IOP, allowed the different partners, in charge of the development and testing the different specific modules within each one of the applications, to connect each parts to the others and perform the necessary integration tests to make sure that all modules work together as expected. Particularly, the following points have been covered:

- Connection of real and simulated assets to the WiseGRID IOP, in order to test the data ingestion of the applications;
- Intercommunication among the different modules, by connecting them all via Internet to the internal ESB of each application;
- Hosting most of the modules composing the applications;
- Testing of the KPI engine implementation on a local instance of Spark Server (in parallel to the development of the Big Data Platform);
- Testing technologies that will facilitate deployment of the modules in the pilot sites (Docker and Docker-compose);
- Access to preliminary versions of the User Interfaces of the applications.

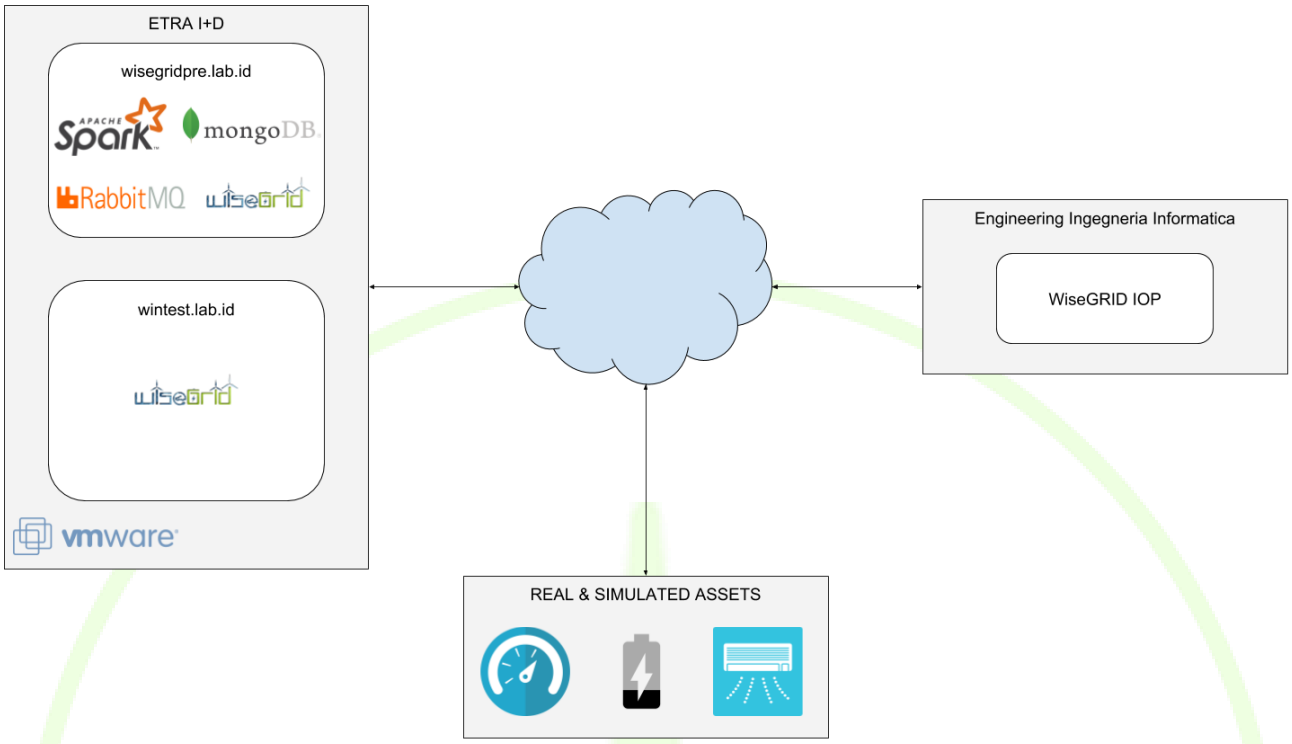



**Figure 3 – Lab-testing platform core modules overview**

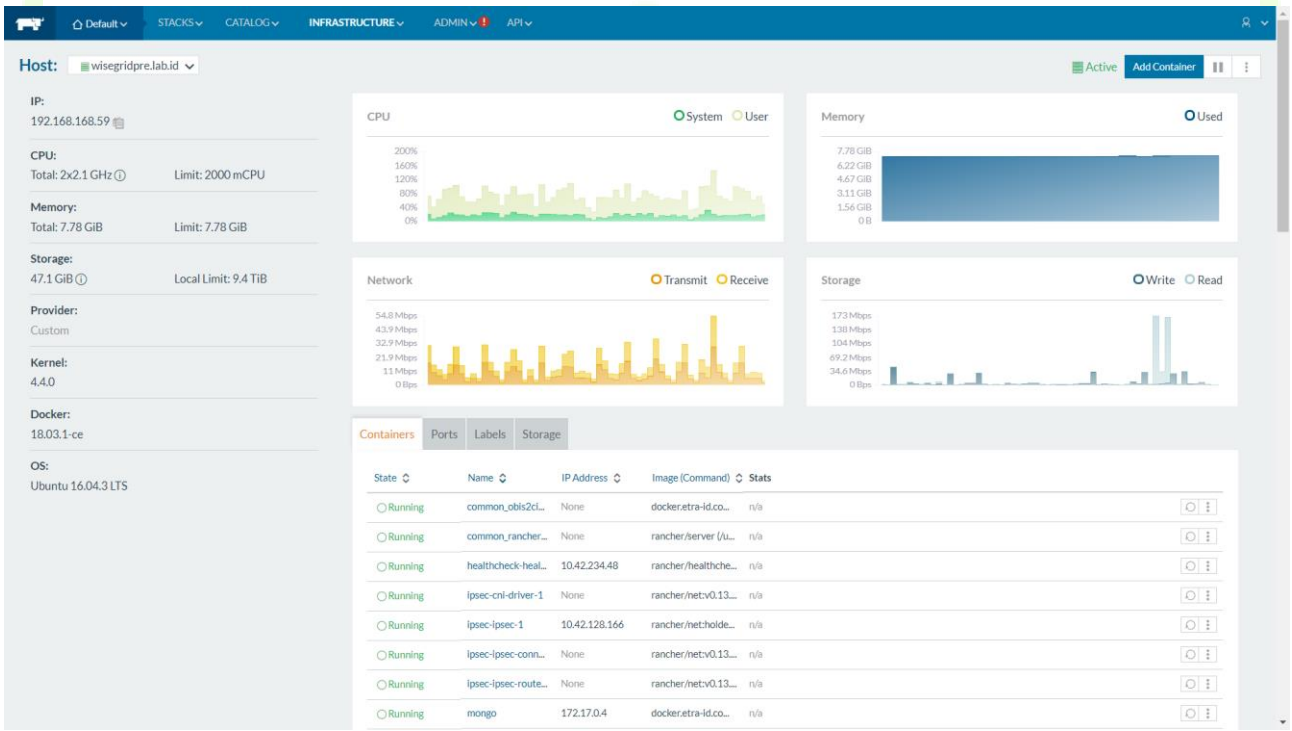



**Figure 4 – Dashboard of wisegridpre.lab.id, all modules installed as Docker containers**

# 3 WISECOOP APPLICATION

## 3.1 IMPLEMENTATION

### 3.1.1 Architecture overview

The architecture of the WiseCOOP application finally implemented does not differ significantly from the architecture presented in the previous deliverable D7.1 WiseCOOP and WiseCORP Apps Design [4]. The architecture is summarized in this section for completeness of this document.

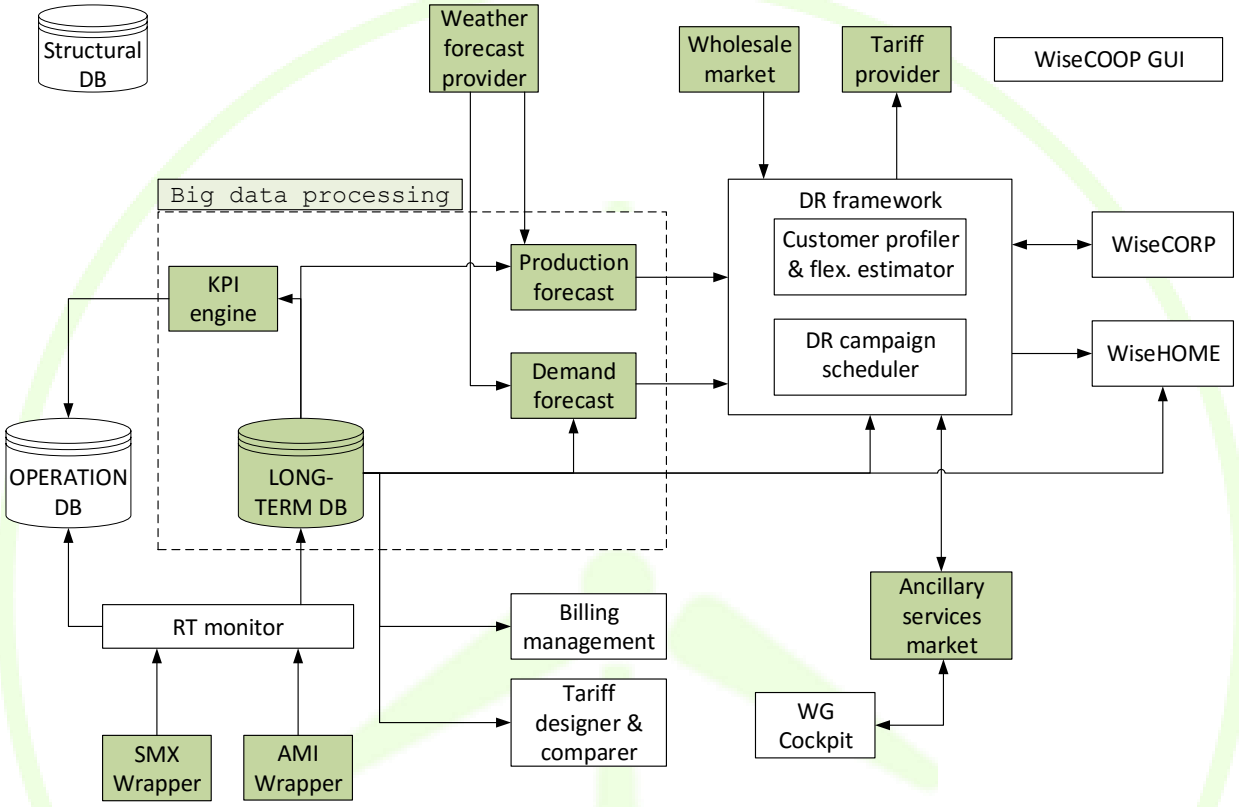



**Figure 5 – Overview of interactions among the modules of the WiseCOOP application**

**Data ingestion**

The first step considered in the design of the application is the data ingestion. The procedure followed is common to other applications in the project, and implies the following steps:

1. Publication of data from *Wrappers* to the *WiseGRID IOP Message Broker*. Following the principle taken in the overall project, data sources publish data to the Interoperable Platform, allowing different application with the corresponding permissions to access to those data flows;
2. Subscription to data flows of interest. In the case of the WiseCOOP application, these data flows are the ones containing energy readings from the members of the portfolio. There are two types of data sources for energy readings that will be integrated in the project: SMXs and AMIs. This subscription is performed by the *RT monitor* module;
3. Store data for further analysis. The *RT monitor* module is in charge of populating both the *Operation* and the *Long-term DB* for further analysis.

**Data analysis**

Under this group, different modules have been defined in order to process the raw data coming from the different data sources in order to get the relevant information out of those. These modules include:

- *KPI engine* module, in charge of extracting different indicators and patterns from the raw data, including profiling of the portfolio members according to different features of their energy demand and production;
- *Demand and production forecast* module, providing forecasts for different aggregations of members of the portfolio.

**Operation and control**

Under this group, different modules have been defined implementing specific tasks in order to fulfil the different functional requirements of the application. These modules include the:

- *Tariff designer* module, allowing the users of the application to define energy tariffs that can be later on used for simulated bills or optimization;
- *Tariff comparer* module, allowing the simulation of energy bills for the members of the portfolio under different pricing schemes;
- *Billing management* module - closely related to the tariff comparer module -, allowing the generation of bills for the members of the portfolio, according to the tariffs those have contracted;
- *DR framework* module, implementing the complete set of functionalities required to enable the design and participation of both implicit and explicit demand response campaigns that will be tested in the project.

**Interaction with other applications**

WiseCOOP will interact with other applications of the project, mainly with the following objectives:

- WiseCORP, WG STaaS/VPP, WiseEVP and WiseHOME will interact with WiseCORP during the participation in implicit and/or explicit demand response campaigns. Since implicit demand response campaigns are articulated by the creation of a dynamic price tariff, participation in implicit demand response campaign is extensible to any other application using tariff pricing as an input considered in its internal optimizations.
- WiseGRID Cockpit, the tool targeting DSO operators, will interact with WiseCOOP through the *Ancillary Services Market*, in order to request support for assisting the correct operation of the distribution grid when required. WiseCOOP participation in these market will be realized by triggering the corresponding explicit Demand Response campaigns on members of the portfolio with capability to participate in those, in order to cover the flexibility required by the DSO.

**Horizontal and support functionalities**

Different modules will be used indirectly by the WiseCOOP application. Summarizing, these modules are data providers that offer information needed by other modules of the application to fulfil their duties, which are reused among different applications developed within the project. The list includes the *Weather Forecast*, *Wholesale Market* and *Tariff Provider* modules, as well as the *Big Data platform* that will support the long-term storage and analysis. In addition, the *WiseCOOP User Interface* is included in this category, providing web-based access to the information and functionalities provided by the other modules.

### 3.1.2 Back-office modules

#### 3.1.2.1 Internal Enterprise Service Bus

As depicted in the architectural overview, the application is actually composed of several modules with well-defined functionalities, which collaborate with each other in order to enable the high-level functionalities of the application. In order to facilitate communication among the modules, it was decided during the design

phase to incorporate an internal Enterprise Service Bus to the application. The selected technology for deploying this communication bus is RabbitMQ, since it covers most of the requirements settled during the design phase and exposed in D7.1.

RabbitMQ has been configured with the following main characteristics:

- Credential-based access control: one credential has been given to each partner requiring access.
- Protocols enabled: AMQP, MQTT and HTTP.
- Virtual hosts: a specific virtual host (/wisecoop) has been configured to partition the communication flows of the modules of this application.

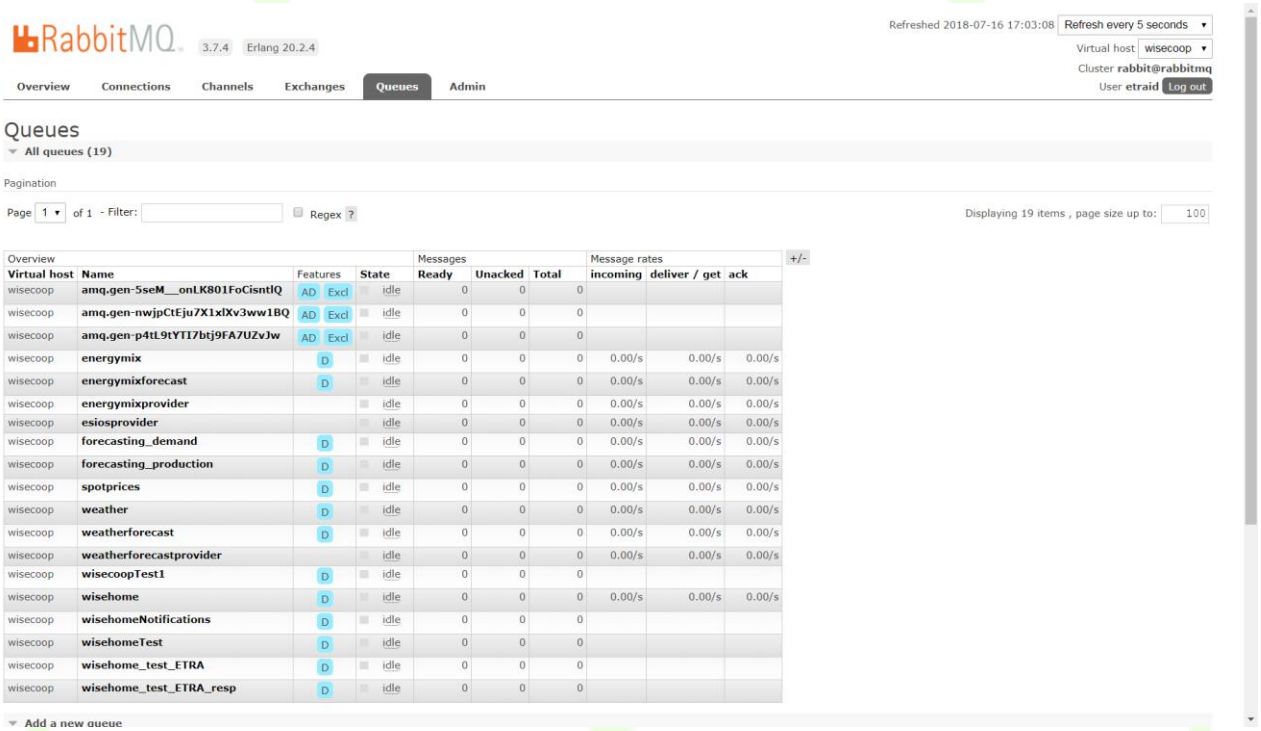



**Figure 6 – Queues created in the internal ESB for exchange of information among modules of WiseCOOP**

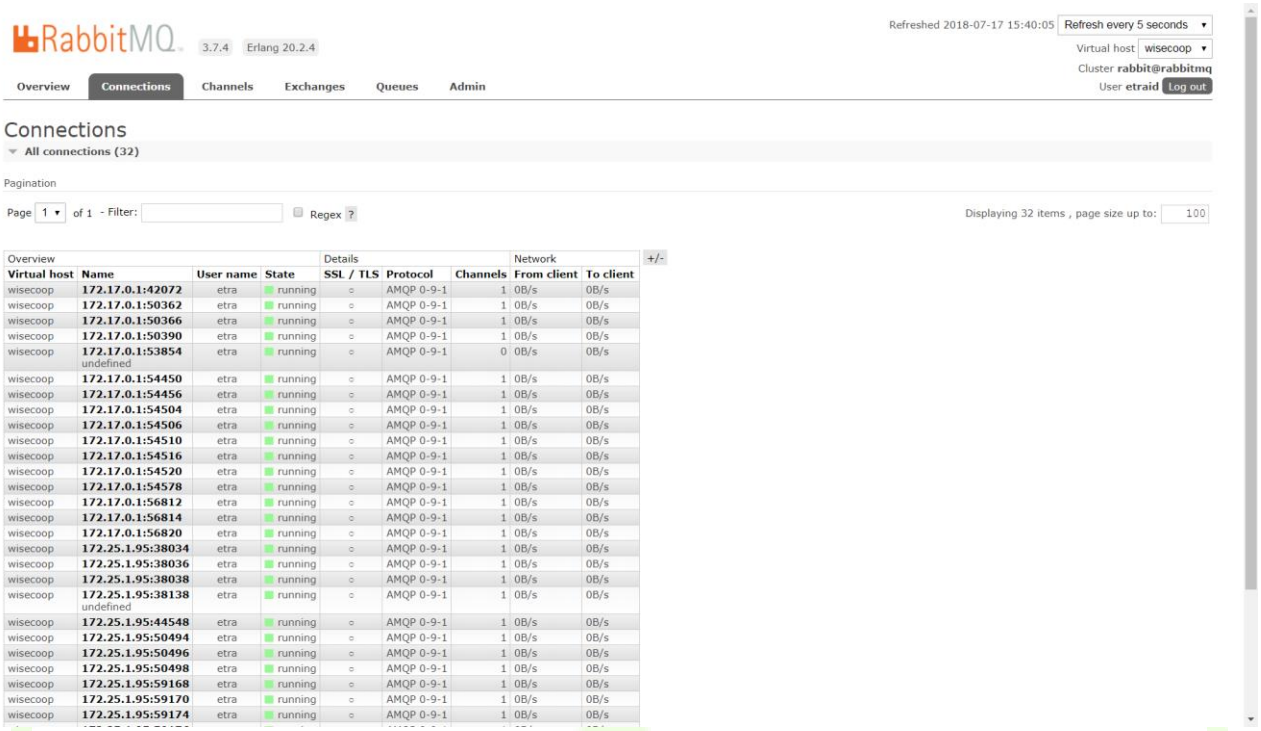



**Figure 7 – List of actives connections (32) to the internal ESB of WiseCOOP**

#### 3.1.2.2 Real-time monitor

The *Real Time monitor* is the horizontal module that will handle the data ingestion for most of the applications of the project. It has been designed in order to fulfil the requirements for data ingestion accordingly to the requirements and the architecture of communications proposed for the applications.

Particularly for WiseCOOP, this module is in charge of tracking and storing in the databases of WiseCOOP the data items shown in the following table.

**Table 11 – Data items tracked by Real-time monitor in WiseCOOP application**

| Data item | Source | Operational DB | Long-term DB |
|---|---|---|---|
| **Energy mix** | ENTSOE energy mix provider | X | X |
| **Energy mix forecast** | ENTSOE energy mix provider | X | X |
| **Energy readings** | Field assets (SMX, AMI systems) | X | X |
| **Weather** | Weather forecast provider | X | X |
| **Weather forecast** | Weather forecast provider | X | X |
| **Wholesale market prices** | Wholesale market | X | |

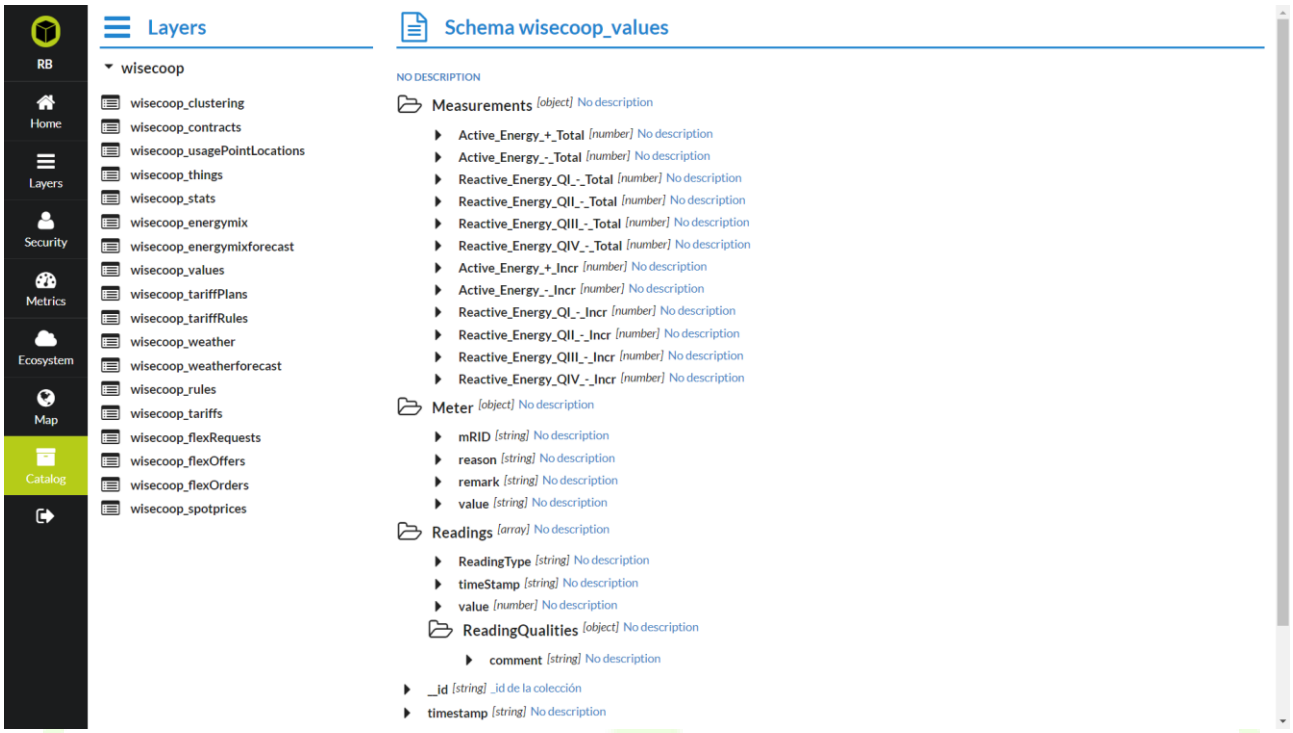



**Figure 8 – Screenshot of the Real-time monitor UI, showing energy readings schema**

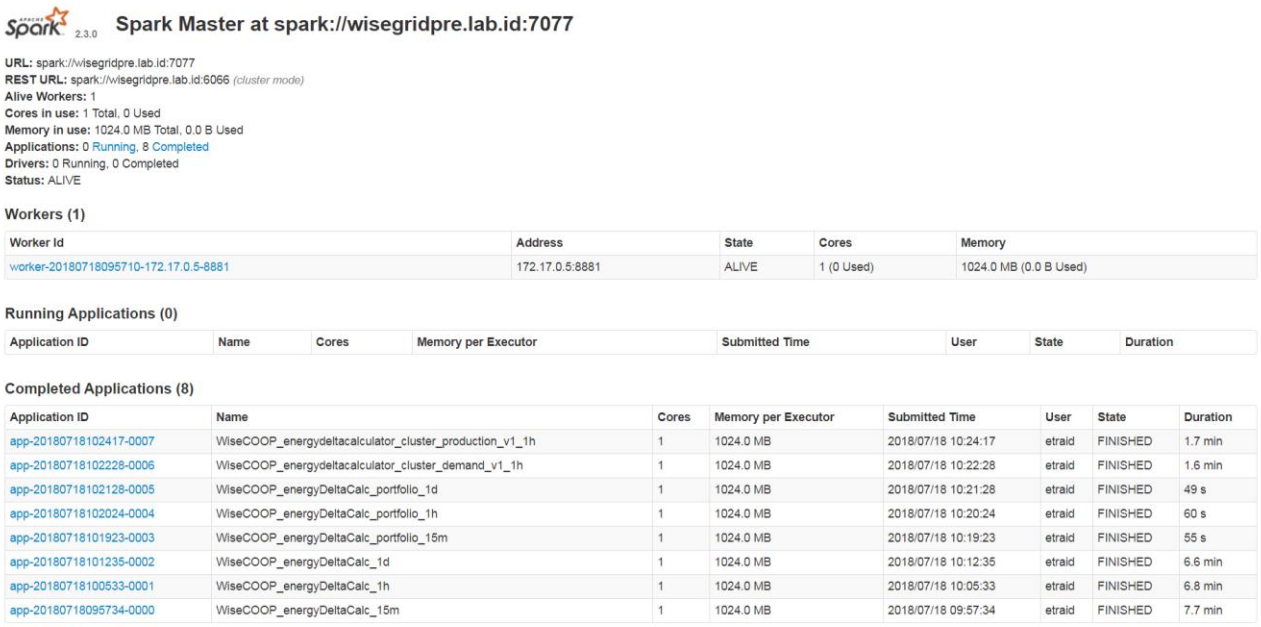
### 3.1.2.3 KPI Engine

The KPI engine of WiseCOOP application has been implemented as a set of Spark jobs that are periodically triggered on the long-term database to perform the necessary calculations and push the results back to different collections of the database.

**Table 12 – WiseCOOP – Spark jobs of the KPI engine**

| Spark job | Module | Description | Result KPIs |
|---|---|---|---|
| **WiseCOOP energyDeltaCalc 15m** | energydeltacalculator-assembly-1.3.jar | Calculates aggregated registers for every meter and every 15 minutes | Energy demand<br>Energy production<br>Equivalent $CO_2$ emissions |
| **WiseCOOP energyDeltaCalc 1h** | energydeltacalculator-assembly-1.3.jar | Calculates aggregated registers for every meter and every hour | Energy demand<br>Energy production<br>Equivalent $CO_2$ emissions |
| **WiseCOOP energyDeltaCalc 1d** | energydeltacalculator-assembly-1.3.jar | Calculates aggregated registers for every meter and every day | Energy demand<br>Energy production<br>Equivalent $CO_2$ emissions |
| **WiseCOOP energyDeltaCalc portfolio 15m** | energydeltacalculator_portfolio-assembly-0.3.jar | Calculates aggregated registers for the whole portfolio of prosumers every 15 minutes | Energy demand<br>Energy production<br>Equivalent $CO_2$ emissions |

| | | | |
|---|---|---|---|
| **WiseCOOP energyDeltaCalc portfolio 1h** | energydeltacalculator_portfolio-assembly-0.3.jar | Calculates aggregated registers for the whole portfolio of prosumers every hour | Energy demand<br>Energy production<br>Equivalent $CO_2$ emissions |
| **WiseCOOP energyDeltaCalc portfolio 1d** | energydeltacalculator_portfolio-assembly-0.3.jar | Calculates aggregated registers for the whole portfolio of prosumers every day | Energy demand<br>Energy production<br>Equivalent $CO_2$ emissions |
| **WiseCOOP clustering demand 2p** | clustering-assembly-0.1.jar | Identifies similar behaviours attending to the demand metrics of the prosumers of the portfolio, classifying those together with other members with similar behaviours | Prosumer profiling (assignment to a group with other members with similar behaviour) |
| **WiseCOOP clustering production** | clustering-assembly-0.1.jar | Identifies similar behaviours attending to the production metrics of the prosumers of the portfolio, classifying those together with other members with similar behaviours | Prosumer profiling (assignment to a group with other members with similar behaviour) |
| **WiseCOOP cluster demand v1 1h** | energydeltacalculator_cluster-assembly-0.3.jar | Calculates hourly average behaviour of the members of each demand-based cluster identified | Energy demand<br>Energy production<br>Equivalent $CO_2$ emissions |
| **WiseCOOP cluster production v1 1h** | energydeltacalculator_cluster-assembly-0.3.jar | Calculates hourly average behaviour of the members of each production-based cluster identified | Energy demand<br>Energy production<br>Equivalent $CO_2$ emissions |



Spark Master at spark://wisegridpre.lab.id:7077

URL: spark://wisegridpre.lab.id:7077
REST URL: spark://wisegridpre.lab.id:6066 (cluster mode)
Alive Workers: 1
Cores in use: 1 Total, 0 Used
Memory in use: 1024.0 MB Total, 0.0 B Used
Applications: 0 Running, 8 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (1)

| Worker Id | Address | State | Cores | Memory |
|---|---|---|---|---|
| worker-20180718095710-172.17.0.5-8881 | 172.17.0.5:8881 | ALIVE | 1 (0 Used) | 1024.0 MB (0.0 B Used) |

Running Applications (0)

| Application ID | Name | Cores | Memory per Executor | Submitted Time | User | State | Duration |
|---|---|---|---|---|---|---|---|

Completed Applications (8)

| Application ID | Name | Cores | Memory per Executor | Submitted Time | User | State | Duration |
|---|---|---|---|---|---|---|---|
| app-20180718102417-0007 | WiseCOOP_energydeltacalculator_cluster_production_v1_1h | 1 | 1024.0 MB | 2018/07/18 10:24:17 | etraid | FINISHED | 1.7 min |
| app-20180718102228-0006 | WiseCOOP_energydeltacalculator_cluster_demand_v1_1h | 1 | 1024.0 MB | 2018/07/18 10:22:28 | etraid | FINISHED | 1.6 min |
| app-20180718102128-0005 | WiseCOOP_energyDeltaCalc_portfolio_1d | 1 | 1024.0 MB | 2018/07/18 10:21:28 | etraid | FINISHED | 49 s |
| app-20180718102024-0004 | WiseCOOP_energyDeltaCalc_portfolio_1h | 1 | 1024.0 MB | 2018/07/18 10:20:24 | etraid | FINISHED | 60 s |
| app-20180718101923-0003 | WiseCOOP_energyDeltaCalc_portfolio_15m | 1 | 1024.0 MB | 2018/07/18 10:19:23 | etraid | FINISHED | 55 s |
| app-20180718101235-0002 | WiseCOOP_energyDeltaCalc_1d | 1 | 1024.0 MB | 2018/07/18 10:12:35 | etraid | FINISHED | 6.6 min |
| app-20180718100533-0001 | WiseCOOP_energyDeltaCalc_1h | 1 | 1024.0 MB | 2018/07/18 10:05:33 | etraid | FINISHED | 6.8 min |
| app-20180718095734-0000 | WiseCOOP_energyDeltaCalc_15m | 1 | 1024.0 MB | 2018/07/18 09:57:34 | etraid | FINISHED | 7.7 min |

**Figure 9 – Screenshot of Spark server with executing WiseCOOP jobs**

#### 3.1.2.3.1 Prosumer profiling (clustering)

The KPI engine of WiseCOOP includes modules with the purpose of analysing the behaviour of the prosumers of the portfolio according to different metrics, and grouping them under groups of members with similar behaviours. This is useful to the user of the WiseCOOP tool, since it allows to identify common behaviour partners that can be used as baseline profiles for different tasks, such as identifying the better tariff for a group of prosumers, identify anomalous behaviours (i.e. prosumers whose behaviour deviates significantly from the baseline) or provide key figures to promote and incentive green behaviour on the prosumers (e.g. motivate individuals to achieve lower $CO_2$ emissions than other similar prosumers).

##### 3.1.2.3.1.1 Demand profiling

The profiling of the prosumers based on the demand analyses the following 4 features using the k-means clustering algorithm:

- Average daily demand for working days, from 08:00 to 18:00
- Average daily demand for working days, from 18:00 to 08:00
- Average daily demand for non-working days, from 08:00 to 18:00
- Average daily demand for non-working days, from 18:00 to 08:00

The optimum number of clusters is automatically determined by following the Elbow method [5].

**Table 13 – Summary of demand-based cluster centres on lab-testing data**


```
{
    "_id" : "5b1bbab48d6d6c32ac7ed83f",
    "type" : "demand_v1",
    "timestamp" : ISODate("2018-06-09T11:32:04.211Z"),
    "clusterCenters" : [
        {
            "wd_p1" : 1693.62962962963,
```

```
            "wd_p2" : 1591.2962962963,
            "nwd_p1" : 780.925925925926,
            "nwd_p2" : 1591.18518518518,
            "members" : [
                …
            ]
        },
        {
            "wd_p1" : 3988.38095238095,
            "wd_p2" : 4167.52380952381,
            "nwd_p1" : 2111.52380952381,
            "nwd_p2" : 5734.23809523809,
            "members" : [
                …
            ]
        },
        {
            "wd_p1" : 4260.75,
            "wd_p2" : 4030.25,
            "nwd_p1" : 1989.5,
            "nwd_p2" : 22515.5,
            "members" : [
                …
            ]
        }
    ],
    "description" : "2-period (08:00 - 18:00) distribution of demand (working/non-working days)"
}
```


#### 3.1.2.3.1.2 Production profiling

The profiling of the prosumers based on their production just considers one feature: the average daily production. Clustering is performed using the k-means clustering algorithm.

The optimum number of clusters is also automatically determined by following the Elbow method [5].

**Table 14 – Summary of production-based cluster centres on lab-testing data**


```
{
    "_id" : "5b1bbab48d6d6c32ac7ed83f",
    "type" : "demand_v1",
```

```
    "timestamp" : ISODate("2018-06-09T11:32:04.211Z"),
    "clusterCenters" : [
        {
            "wd_p1" : 1693.62962962963,
            "wd_p2" : 1591.2962962963,
            "nwd_p1" : 780.925925925926,
            "nwd_p2" : 1591.18518518518,
            "members" : [
                …
            ]
        },
        {
            "wd_p1" : 3988.38095238095,
            "wd_p2" : 4167.52380952381,
            "nwd_p1" : 2111.52380952381,
            "nwd_p2" : 5734.23809523809,
            "members" : [
                …
            ]
        },
        {
            "wd_p1" : 4260.75,
            "wd_p2" : 4030.25,
            "nwd_p1" : 1989.5,
            "nwd_p2" : 22515.5,
            "members" : [
                …
            ]
        }
    ],
    "description" : "2-period (08:00 - 18:00) distribution of demand (working/non-working days)"
}
```


#### 3.1.2.3.1.3 $CO_2$ equivalent emissions profiling

The profiling of the prosumers based on the equivalent $CO_2$ emissions of their demand just considers one feature: the daily equivalent $CO_2$ emissions. Clustering is performed using the k-means clustering algorithm.

The optimum number of clusters is also automatically determined by following the Elbow method [5].

Table 15 – Summary of $CO_2$-based cluster centres on lab-testing data


```
{
    "_id" : "co2_v1",
    "type" : "co2_v1",
    "timestamp" : ISODate("2018-07-25T13:05:52.369Z"),
    "clusterCenters" : [
        {
            "co2Emissions" : 706.427166739829,
            "members" : [
                …
            ]
        },
        {
            "co2Emissions" : 5097.34058807193,
            "members" : [
                …
            ]
        },
        {
            "co2Emissions" : 2128.00813436411,
            "members" : [
                …
            ]
        }
    ],
    "description" : "Average daily CO2 emissions"
}
```


#### 3.1.2.3.1.4 Economic cost profiling

The profiling of the prosumers based on the cost associated to their demand just considers one feature: the daily economiccost. Clustering is performed using the k-means clustering algorithm.

The optimum number of clusters is also automatically determined by following the Elbow method [5].

Table 16 – Summary of economic cost-based cluster centres on lab-testing data


```
{
    "_id" : "cost_v1",
    "type" : "cost_v1",
    "timestamp" : ISODate("2018-07-25T14:06:06.086Z"),
    "clusterCenters" : [
```

```
        {
            "cost" : 0.897837893055555,
            "members" : [
                …
            ]
        },
        {
            "cost" : 0.424609498522727,
            "members" : [
                …
            ]
        },
        {
            "cost" : 1.7924583875,
            "members" : [
                …
            ]
        },
        {
            "cost" : 0.099988480625,
            "members" : [
                …
            ]
        }
    ],
    "description" : "Average daily energy cost"
}
```


#### 3.1.2.3.2 Indicators required by WiseHOME

Additionally, a module has been developed with the objective of calculating and providing the required information for the WiseHOME application. This module uses pre-computed indicators – as exposed above – but also computes additional indicators on the fly as required, by posting aggregation queries to the long-term MongoDB database.

**Table 17 – Capture of a conversation between WiseCORP and WiseHOME**


```
>>
{"header":{"messageType":"REQUEST","conversationID":"ECPDK1004001","rec
ipient":"ECOPOWER","sender":"SampleSender"},"body":[{"reference":"USER"
,"assetKey":"ZIV0034949309","dataType":"TIMESERIES","metricType":"ENERG
Y_CONSUMPTION","startTime":"2018-07-17T10:18:21.939Z","endTime":"2018-
07-18T10:18:21.939Z","sampleTime":"15
MIN","value":null,"metricTimeseries":null},{"reference":"USER","assetKe
```

```
y":"ZIV0034949309","dataType":"TIMESERIES","metricType":"ENERGY_CONSUMP
TION","startTime":"2017-07-17T10:18:21.939Z","endTime":"2017-07-
18T10:18:21.939Z","sampleTime":"15
MIN","value":null,"metricTimeseries":null}]}
<<
{"header":{"conversationID":"ECPDK1004001","messageType":"RESPONSE"},"b
ody":[{"reference":"USER","assetKey":"ZIV0034949309","dataType":"TIMESE
RIES","metricType":"ENERGY_CONSUMPTION","startTime":"2018-07-
17T10:18:21.939Z","endTime":"2018-07-18T10:18:21.939Z","sampleTime":"15
MIN","value":null,"metricTimeseries":[{"timestamp":"2018-07-
17T11:15:00.000Z","value":30},{"timestamp":"2018-07-
17T12:15:00.000Z","value":30},{"timestamp":"2018-07-
17T13:15:00.000Z","value":29},{"timestamp":"2018-07-
17T14:15:00.000Z","value":30},{"timestamp":"2018-07-
17T15:15:00.000Z","value":29},{"timestamp":"2018-07-
17T16:15:00.000Z","value":29},{"timestamp":"2018-07-
17T17:15:00.000Z","value":28},{"timestamp":"2018-07-
17T18:15:00.000Z","value":28},{"timestamp":"2018-07-
17T19:15:00.000Z","value":294},{"timestamp":"2018-07-
17T20:15:00.000Z","value":801},{"timestamp":"2018-07-
17T21:15:00.000Z","value":207},{"timestamp":"2018-07-
17T22:15:00.000Z","value":29},{"timestamp":"2018-07-
17T23:15:00.000Z","value":30},{"timestamp":"2018-07-
18T00:15:00.000Z","value":29},{"timestamp":"2018-07-
18T01:15:00.000Z","value":30},{"timestamp":"2018-07-
18T02:15:00.000Z","value":29},{"timestamp":"2018-07-
18T04:15:00.000Z","value":75},{"timestamp":"2018-07-
18T05:15:00.000Z","value":77},{"timestamp":"2018-07-
18T06:15:00.000Z","value":332},{"timestamp":"2018-07-
18T07:15:00.000Z","value":308},{"timestamp":"2018-07-
18T08:15:00.000Z","value":327},{"timestamp":"2018-07-
18T09:15:00.000Z","value":95}]},{"reference":"USER","assetKey":"ZIV0034
949309","dataType":"TIMESERIES","metricType":"ENERGY_CONSUMPTION","star
tTime":"2017-07-17T10:18:21.939Z","endTime":"2017-07-
18T10:18:21.939Z","sampleTime":"15
MIN","value":null,"metricTimeseries":[]}]}
```


#### 3.1.2.4 Demand and production forecast service

This module of the WiseCOOP has been implemented over a RPC server which makes use of the ESB. In addition, this module makes use of the long-term database of the WiseCOOP, which is implemented over a MongoDB database. The RPC servers (demand and generation forecast) of this module are permanently running to manage the received queries through the RabbitMQ queues enabled to make use of the demand and production forecast.

The ID of the supply point and the period and the horizon of the desired forecast are specified within the message queries. In the case of production forecast, in addition to the defined fields, the type of generation technology is specified.

Once the query is de-serialized and parsed, the forecast module retrieves from the long-term database the necessary information to perform the forecast. To perform the forecast it retrieves information related to the consumed/produced energy, working calendar and weather information related to the queried installation. This information is available in the long-term database, as it is possible to appreciate in the next picture.

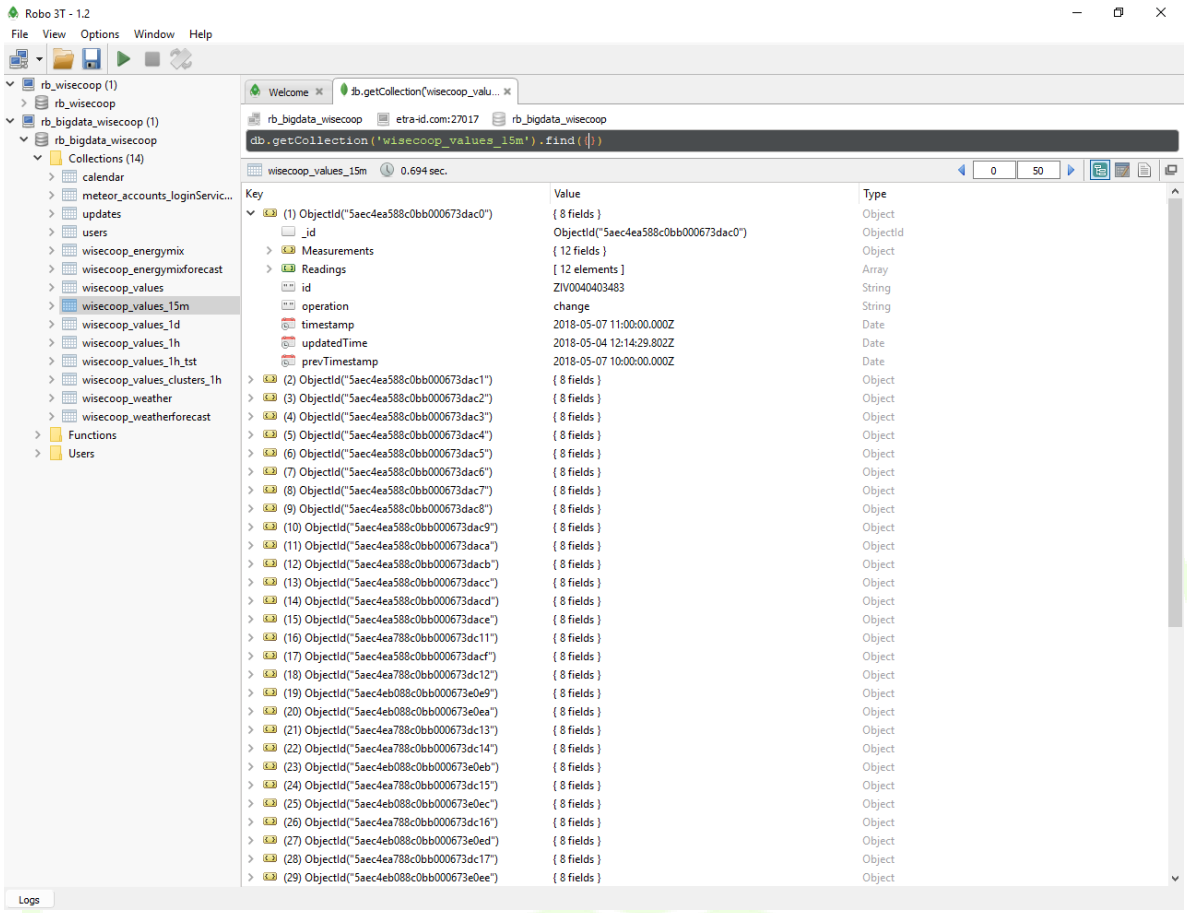



**Figure 10 – WiseCOOP long-term database screenshot**

Once the algorithm is ran, the response provided by this is serialized and sent back through the corresponding RabbitMQ queue, providing the queried information. The next message is an example of the response received by the WiseCOOP, which is printed in the graphic of the forecast view.

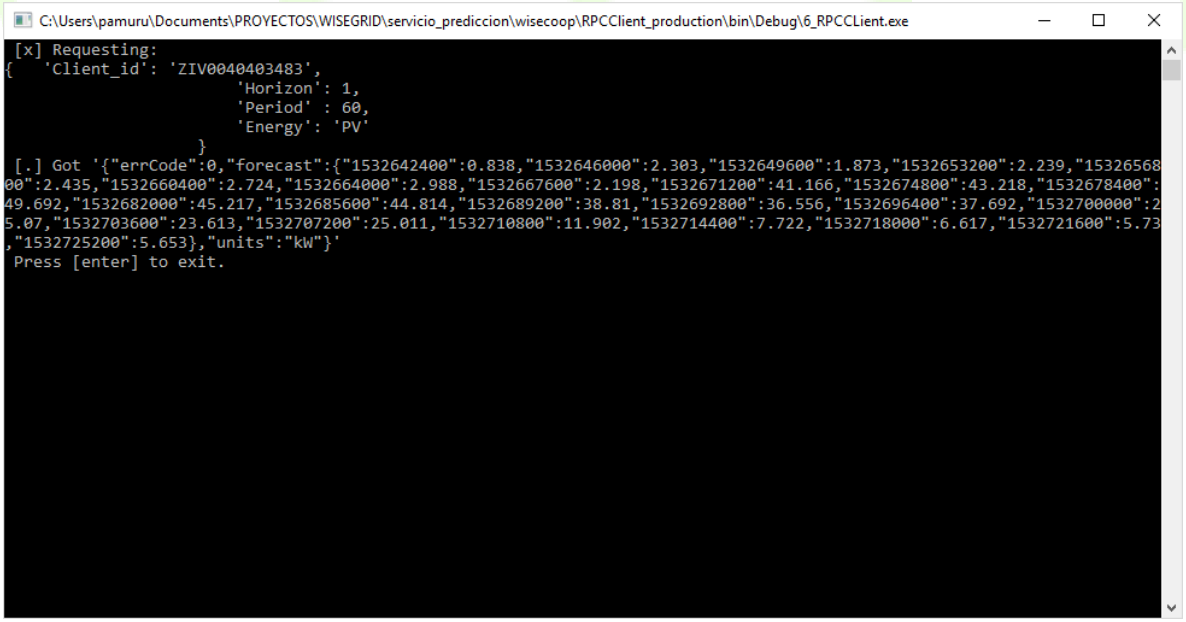



**Figure 11 – Screenshot of forecast response message.**

#### 3.1.2.5 Demand response framework

The Demand Response framework is an end-to-end functionality that spans both the WiseCOOP and the WiseCORP tools. Within WiseCOOP, lie two main components of this functionality that exchange information with WiseCORP.

The first is responsible for collecting the demand flexibility potential of the various buildings that belong to the portfolio of the WiseCOOP operator. For this purpose, WiseCOOP issues commands – such as the following – in order to elicit the flexibility per building. This command is encapsulated in a message that specifies the target buildings (via their SMX ID), the time-frame of interest for the Demand Response action, as well as – optionally – a comfort level indication that can be used as a threshold to avoid the disruption of internal operations in the building. The information specified in the message usually results from the corresponding DSO request for demand response. So, the buildings are selected based on eligibility criteria, such as location of connection point, and the time-frame corresponds to the time of the expected demand shifting/shedding.

**Table 18 – Elicitation of demand flexibility potential per building**

```
{
  "shoIds": [
    "SMX1",
    "SMX2"
  ],
  "comfortLevel": 0.6,
  "timeperiod": 60
}
```

Once the buildings – actually the WiseCORP instance that is responsible for each of the target buildings – respond to the WiseCOOP request, WiseCOOP will calculate the combination of assets that can best fulfil the request of the DSO. Then the tool issues messages, like the following, to inform assets about the action they must undertake in order to comply with the DSO request. This message specifies exactly which devices (venID) should modify their energy consumption by how much (aggregatedPnode) and at what specific time (startTime/activePeriod).

**Table 19 – Request from WiseCOOP for specific asset demand modulation as a result of a DR event**

**Send DR Request**

```
[
  {
    "eiEventDescriptor": {
      "eventID": "1",
      "createdDateTime": "2018-07-13T12:12:12"
    },
    "eiEventSignals": {
      "eiEventSignal": [
        {
          "signalID": "17",
          "startTime": "2018-07-13T12:00:00",
          "activePeriod": "PT15M",
          "eiTarget": {
            "venID": "SMX2",
            "aggregatedPnode": "10"
          }
        },
        {
          "signalID": "97",
```

```
            "startTime": "2018-07-13T12:15:00",
            "activePeriod": "PT15M",
            "eiTarget": {
              "venID": "SMX2",
              "aggregatedPnode": "8"
            }
          },
          {
            "signalID": "107",
            "startTime": "2018-07-13T12:30:00",
            "activePeriod": "PT15M",
            "eiTarget": {
              "venID": "SMX2",
              "aggregatedPnode": "1.5"
            }
          }
        ],
        "numDataSources": "3"
      },
      "activePeriod": "PT45M",
      "eiTarget": {
        "venID": "SMX2",
        "aggregatedPnode": "19.5"
      }
    }
]
```


### 3.1.3 User interface

In this part of the document, the main sections and functionalities of the WiseCOOP GUI are described, including some screenshots of the actual interfaces.

For the implementation of the WiseCOOP unified GUI, the **MeteorJS** web framework has been used. MeteorJS (or simply 'Meteor'), is *"A free and open-source JavaScript web framework written using Node.js. Meteor allows for rapid prototyping and produces cross-platform (Android, iOS, Web) code. It integrates with MongoDB and uses the Distributed Data Protocol and a publish–subscribe pattern to automatically propagate data changes to clients without requiring the developer to write any synchronization code. On the client, Meteor depends on jQuery and can be used with any JavaScript UI widget library".*

On the client part of a Meteor application, a number of plugins and technologies can be used to provide the user a better experience. The main plugins we have used for the client side are:

- **SemanticUI** as CSS framework. CSS frameworks are pre-prepared software frameworks that are meant to allow for easier, more standards-compliant web design using the Cascading Style Sheets language. They are mostly design oriented and unobtrusive. This differentiates these from functional and full JavaScript frameworks. By using this CSS framework we achieve easily a modern and coherent style across the whole user interface. The selection of SemanticUI over other CSS framework is mainly based on our expertise and the fact that this one has been designed to be easily understandable and usable. Other framework tends to become quite hard to use as interfaces becomes bigger.
- **LeafletJS** as the mapping solution for the web client. This JavaScript-based framework provides a wide range of mapping providers to use and offers a big set of plugins to personalize the user interaction with the map and the display of the information. And everything is open source and free.
- **HighchartsJS** is a charting JavaScript framework that helps displaying data in the form of charts for web environments.
- **BlazeJS** for the user interface lay out**.** It is a powerful library for creating user interfaces by writing reactive HTML templates.

For the server part, **MongoDB** has been used as the database for storing real-time data. This is a no-SQL database that stores unstructured information. It is tightly coupled with Meteor. The reactive nature of the data changes in MongoDB database is at the core of the web application.

The web application is protected with a user/password credential system to avoid non-authorized personnel to access sensible information. These credentials are requested before accessing the rest of the application.


**Figure 12 – WiseCOOP Login**

This credential system also permits the definition of different user profiles to grant or deny access to each section of the application independently. This functionality provides an additional level of privacy, as well as flexibility for the system administrator and the operators that use the application.
Once the user has been granted access to the application, diverse functionalities will be available as described in the sections below.

#### 3.1.3.1 Dashboard

The dashboard presents aggregated indicators for the overall portfolio and the last 30 days, namely:

- Daily demand
- Daily production
- Demand share per contract type
- Production share per contract type

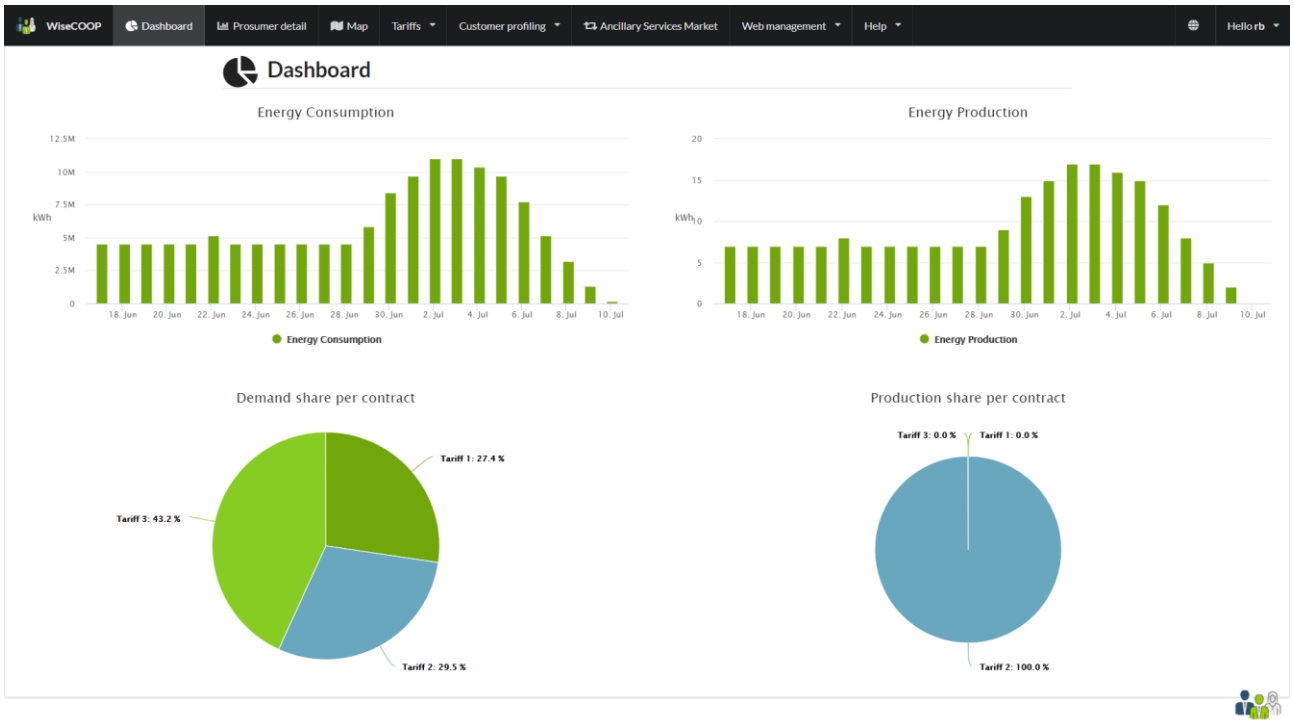



**Figure 13 – WiseCOOP UI – Dashboard**

### 3.1.3.2 Prosumer detail

Under this section, access to raw energy readings is provided. Upon selection of the data to be visualized (asset, date range, metric and integration period), the data is displayed in a chart.

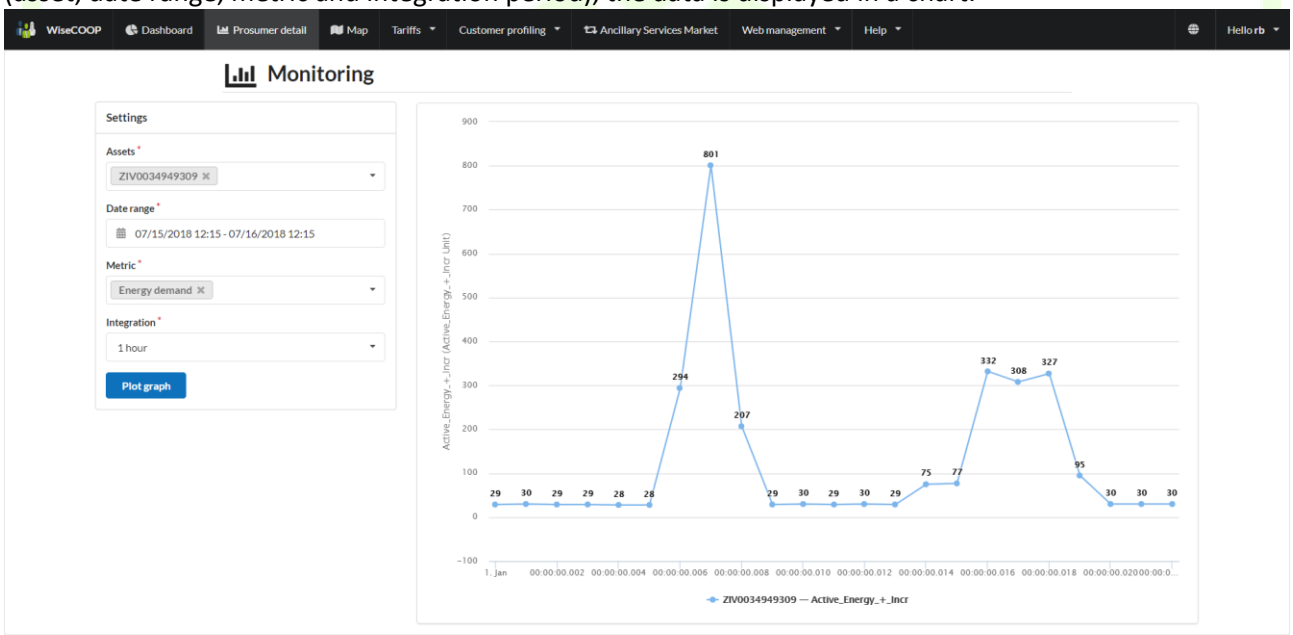



**Figure 14 – WiseCOOP UI – Prosumer detail**

### 3.1.3.3 Geographical demand heat-map

Under this section, a visualization of the demand of the last 30 days is geographically displayed in a map. The demand measured at each geo-referenced asset is aggregated and provided a relative colour (red represents the greatest demand, green the lowest ones).

This visualization can be of use to retailers and aggregators to understand where the members of the portfolio are located, where most of the energy is demanded, identify areas with expansion capabilities, etc.

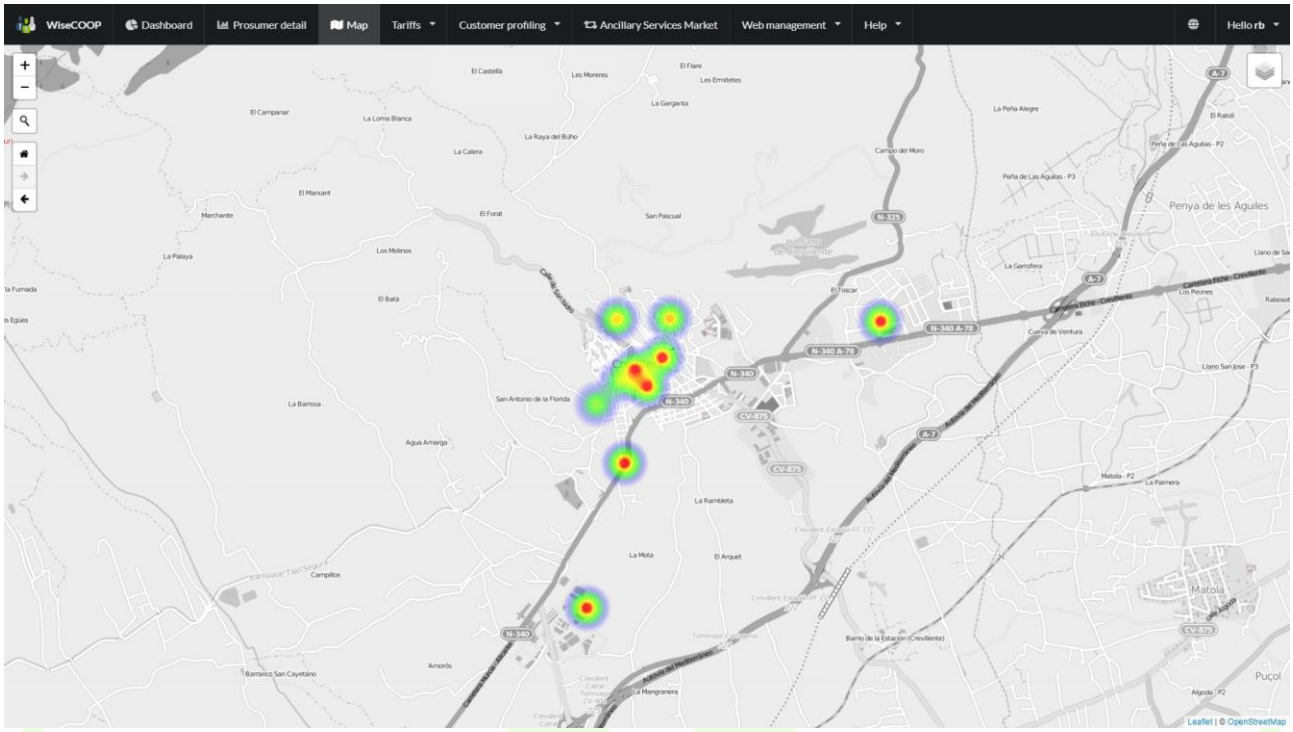



**Figure 15 – WiseCOOP UI – Geographical demand heat-map**

#### 3.1.3.4 Tariff designer

The tariff designer section comprises two different sites: tariff period definition and tariff definition.

##### 3.1.3.4.1 Tariff period definition

This section allows the user of WiseCOOP to introduce in the database new definitions of tariff periods that may be reused among different tariffs. In order to define the periods, the following information is required:

- Name
- Site: pilot site where the tariff period applies (for internal project purposes)
- Country: country the tariff period applies
- Set of period definition rules:
    - Price type: name of the period to which the rule applies (e.g. p1). A single period can be referenced in more than one rule (for instance, several rules are required in order to define different hourly periods for winter and summer time).
    - Application period: specifies whether the rule is defined for a set of hours or a set of days
        - In case the rule applies to a range of hours, the start and end hour (in local time) must be provided.
        - In case the rule applies to a range of days, the start and end date must be provided
    - Special dates: for each type of special day, it defines whether the rule is applied (*true* selected), is not applied (*false* selected) or if the type of day doesn't need to be taken in

consideration (*undefined* selected). The types of special days defined are:

- Saturdays
- Sundays
- Daylight Saving Time (summer time)
- National holidays
- Special holidays (e.g. 6th January in Spain)

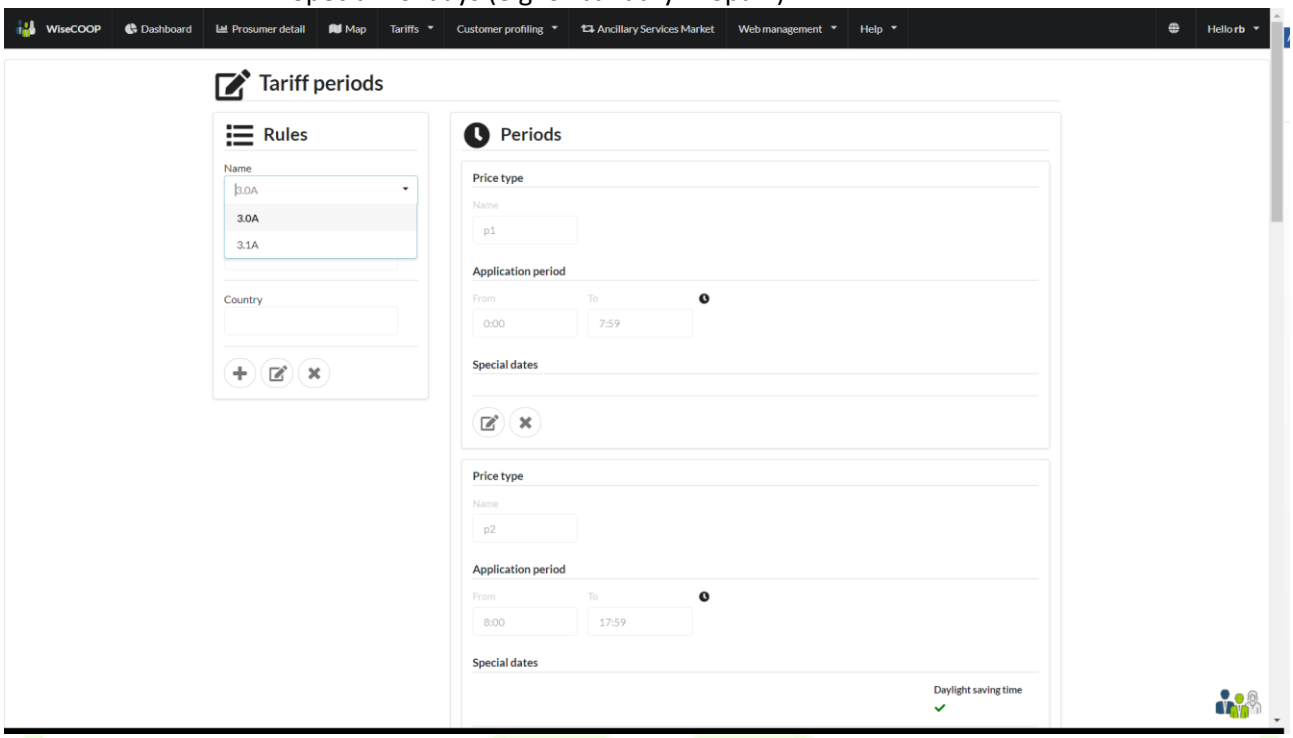



**Figure 16 – WiseCOOP UI – Tariff period definition**

#### 3.1.3.4.2 Tariff definition

This section allows the user of WiseCOOP to create the definition of the tariff. A tariff is composed of the following elements:

- Name
- Tariff period definition that is applicable
- For each tariff period
  - Energy term price (€/kWh)
  - Power term price (annual €/kW in contracted capacity)

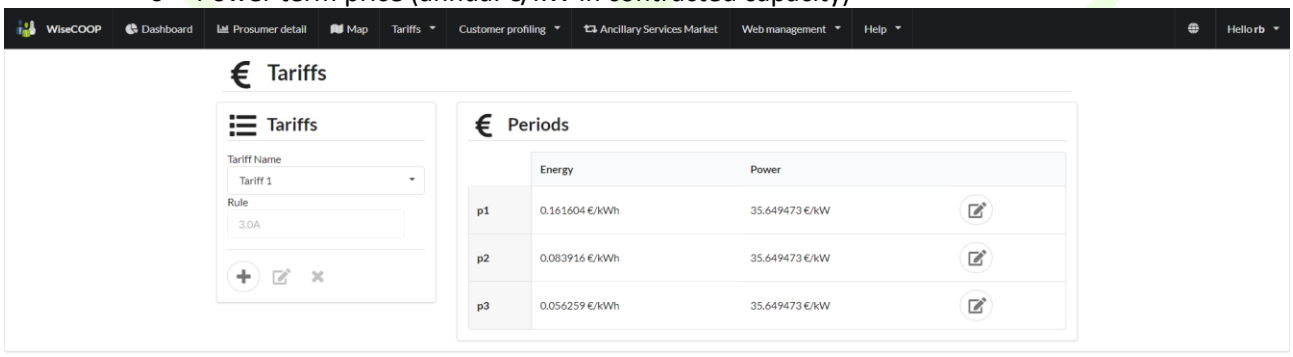



**Figure 17 – WiseCOOP UI – Tariff definition**

#### 3.1.3.5 Portfolio profiling

This section displays the results of the different modules composing the portfolio profiling section of the KPI engine.

The first site displays, for a selected profiling criteria (demand or production profile), the identified groups, their average values for each one of the considered features (cluster centres), and the total share of the portfolio members that take part of that group.

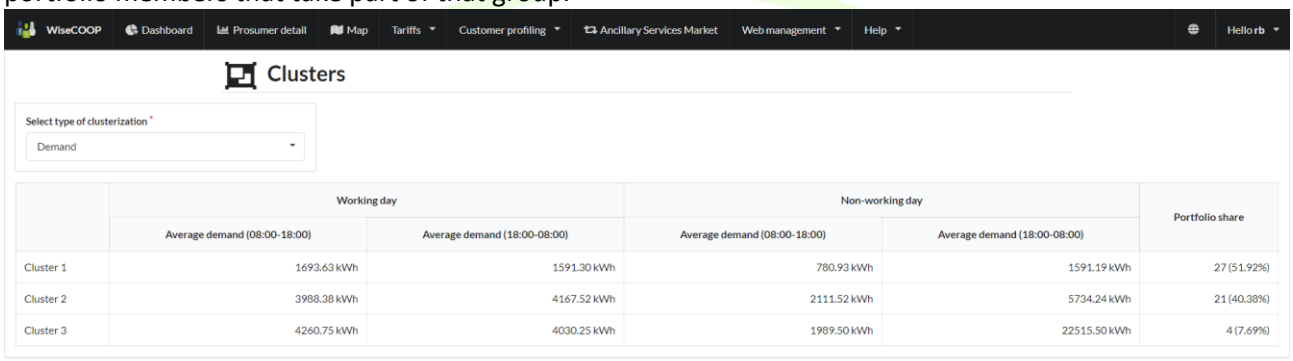



**Figure 18 – WiseCOOP UI – Portfolio profiling, identified groups**

A second site gives further insight in the analysis of the profiling results, in the form of time charts of demand and production. In this section, the user of WiseCOOP needs to select the criteria of the visualization:

- Profiling criteria (demand or production)
- Customer
- Date range

For the selected criteria, two charts are displayed:

- Energy demand chart, comparing the actual demand of the selected customer with the average of the group the customer has been associated to (i.e. the average behaviour of all members with similar demand patterns)
- Energy production chart, comparing the actual production of the selected customer with the average of the group the customer has been associated to (i.e. the average production of all members with similar production patterns)

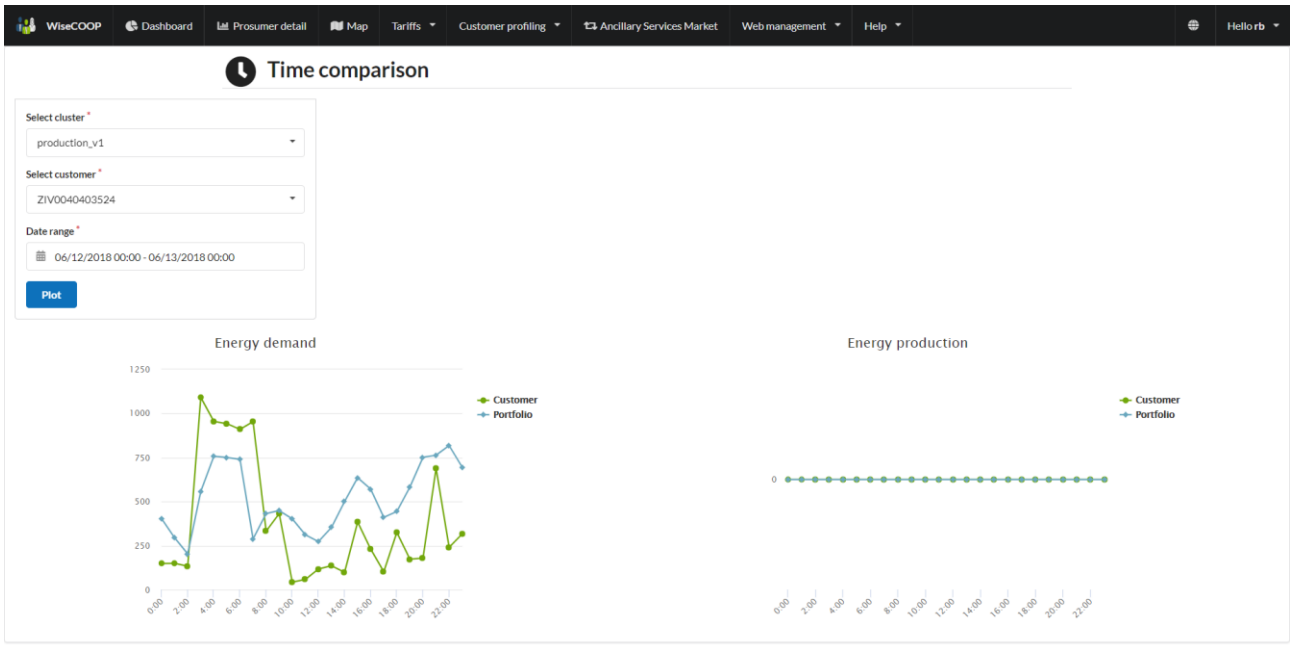



**Figure 19 – WiseCOOP UI – Portfolio profiling, member comparison with average profile**

### 3.1.3.6 Tariff comparer

The tariff optimization section allows the aggregators to use the historical energy demand and production data retrieved from the portfolio of prosumers in energy cost simulations, thus providing valuable input to the tariff selection decision support. Users of WiseCOOP can trigger a new simulation by selecting the prosumer whose data they want to use in the simulation, and a complete month.

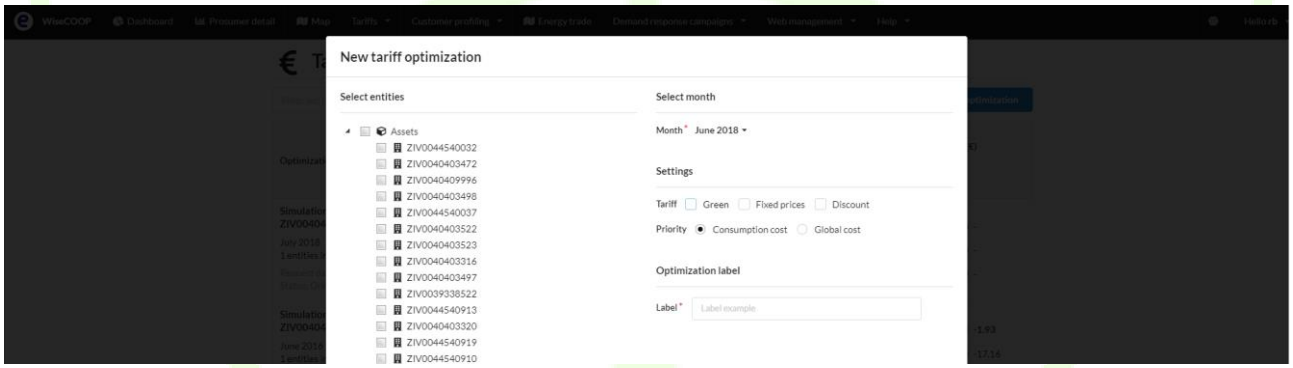



**Figure 20 – WiseCOOP UI – Tariff comparer, selection of criteria**

Once the simulation is triggered, WiseCOOP automatically computes the applicable costs that would result for each one of the tariffs defined within the database. When the simulation finishes, a comprehensive comparison of the results is provided to the aggregator, making it very easy to identify which tariffs would benefit or would increase the costs associated to the energy demand.

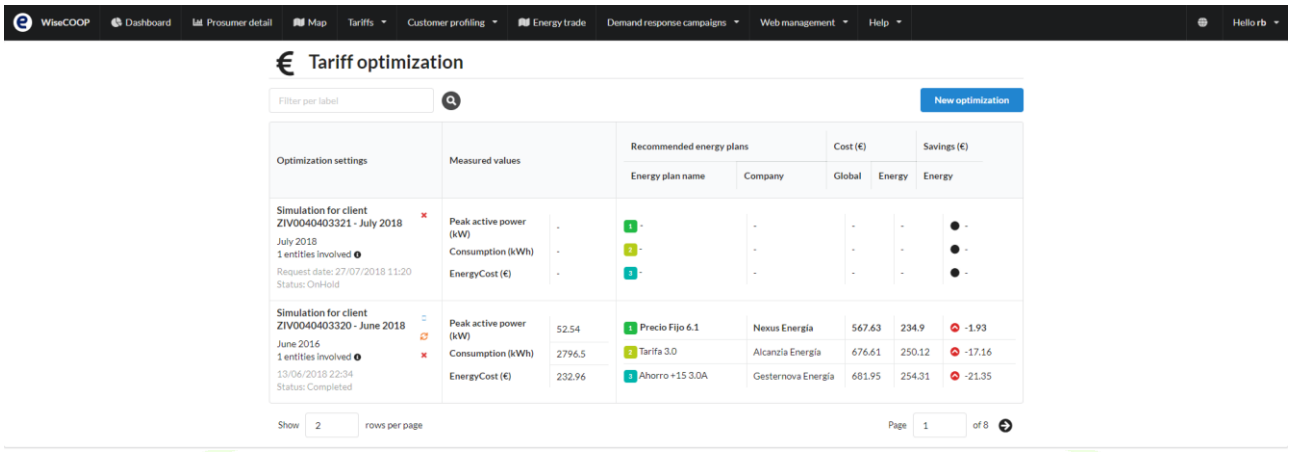



**Figure 21 – WiseCOOP UI – Tariff comparer, results**

In addition, the simulation results include the detail of how the energy bill would look like with each one of the tariffs used in the simulation. These can be used to check the details during the comparison of different tariffs, as relevant information during negotiation of better tariffs with the retailer, or as an assessment of the correctness of the energy bills delivered by the retailer.

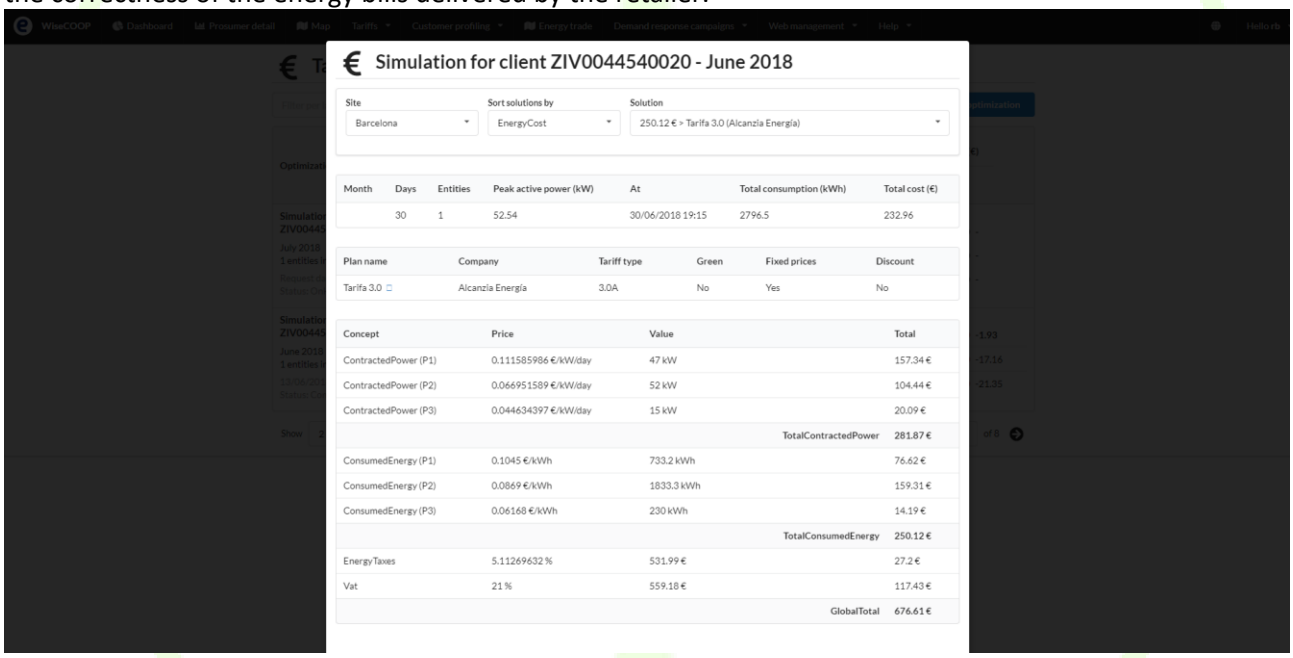



**Figure 22 – WiseCOOP UI - Tariff comparer, simulated bills**

### 3.1.3.7 Energy trade assistant

The energy trade assistant section has the objective of providing to the retailer valuable information about all the information retrieved by WiseCOOP that may assist them in the decision on how to approach the wholesale market. Within this context, the following information is provided for the selected time range:

- Wholesale market prices
- Energy mix of the energy available from wholesale market (RES / non-RES)
- Visualization of the demand forecast (as provided by WiseCOOP) against the actual demand measured by smart meters and AMI systems integrated with the tool. Indicators on deviations among the two are also provided, with the objective of evaluating how the forecasting module can help the

retailer to minimize the deviation between the energy bought from the wholesale market and the actual demand of the portfolio of prosumers

- Visualization of the production forecast (as provided by WiseCOOP) against the actual production measured by smart meters and AMI systems integrated with the tool. Indicators on deviations among the two are also provided, as an indication of the forecasting reliability
- Total economic cost associated to the demand covered by the wholesale market
- Equivalent CO2 emissions of the demand covered by the wholesale market

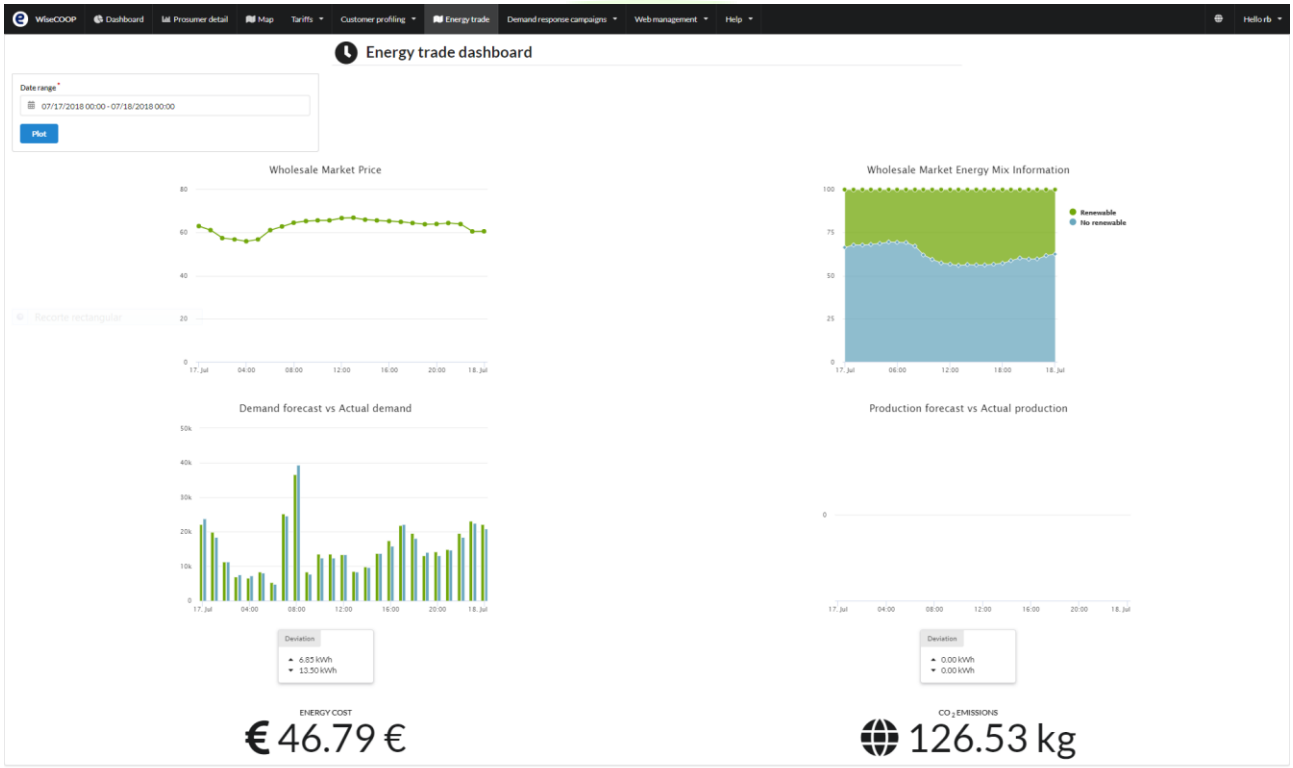



**Figure 23 – WiseCOOP UI – Energy trade assistant**

### 3.1.3.8 Demand response campaigns

#### 3.1.3.8.1 Implicit demand response – dynamic tariff

WiseCOOP includes all features needed to enable aggregators and retailers to implement implicit demand-response strategies by offering to their customers a dynamic tariff that can be modulated accordingly to the interests of the WiseCOOP user. The design of these modules and ideas behind them are described in detail in D10.2. This section of the UI presents the main outcomes of those modules for a selected time range, namely:

- Demand and production forecast, which are the main input for the calculation of the dynamic tariff;
- Imbalance among forecasted demand and production;
- Dynamic price generated with the objective of minimizing the imbalance.

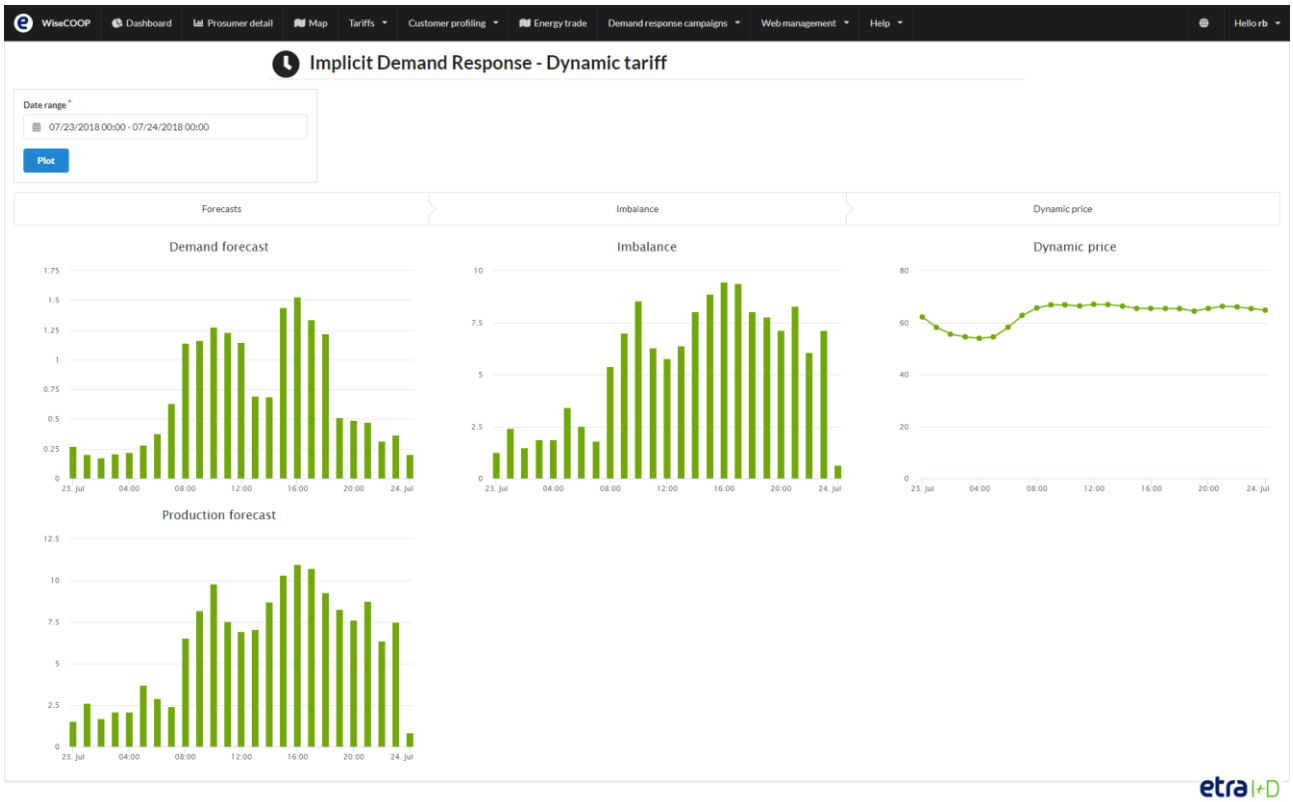



**Figure 24 – WiseCOOP UI – Implicit demand response**

#### 3.1.3.8.2 Explicit demand response

With regards to the participation of the aggregator in the explicit demand response campaigns, this section displays information on the active and finished campaigns initiated by the DSO, providing the following details:

- Congestion point where the demand response campaign is triggered;
- Starting timestamp;
- End timestamp;
- Required flexibility: requested power reduction (or increase) in the congestion area for the given period of time;
- Status: current status of the demand-response campaign (*Request posted*, *Offer sent*, *Order poster*, *DR campaign started*, *DR campaign finished*).

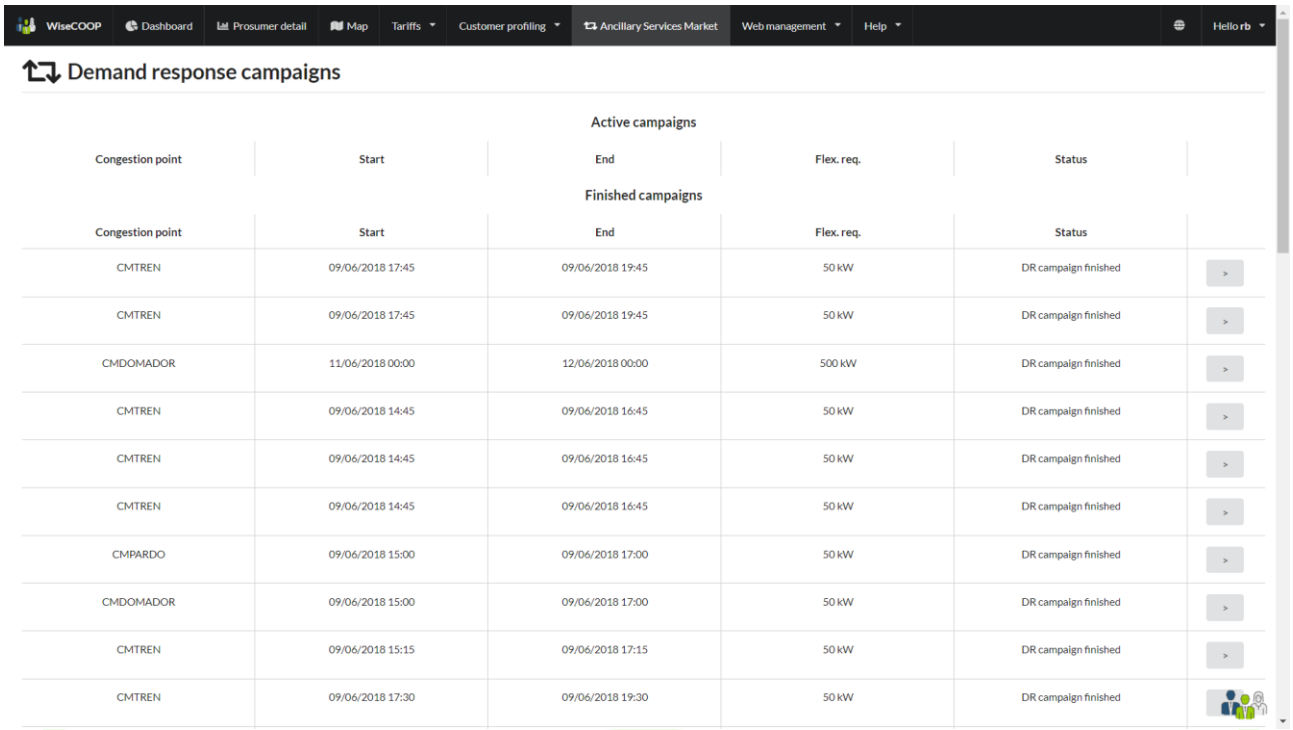



**Figure 25 – WiseCOOP – List of active and finished explicit demand response campaigns**

By selecting one of the campaigns, the corresponding details are displayed, including:

- Starting timestamp;
- Duration of the campaign;
- Required flexibility;
- History of events for this campaign: shows all events that happened in the flexibility market related to this campaign.

**Table 20 – WiseCOOP UI – List of possible status for explicit demand response campaigns**

| | Timestamp | Status | Sender | Recipient | Flexibility | Price |
|---|---|---|---|---|---|---|
| **Flex. request** | Indicates when the request was posted by DSO | Request posted | DSO Operator | Aggregators subscribed to the USEF communications | Indicates the amount of flexibility requested | |
| **Offer sent** | Indicates when the offer was sent to the DSO | Offer sent | Name of the aggregator | DSO Operator | Indicates the amount of flexibility offered | Indicates the price requested for the offered flexibility |
| **Order posted** | Indicates when the order was posted by DSO | Order posted | DSO Operator | Name of the selected aggregator | Indicates the amount of flexibility ordered (the amount offered by the aggregator) | Indicates the price agreed with the aggregator (the price requested by the aggregator) |

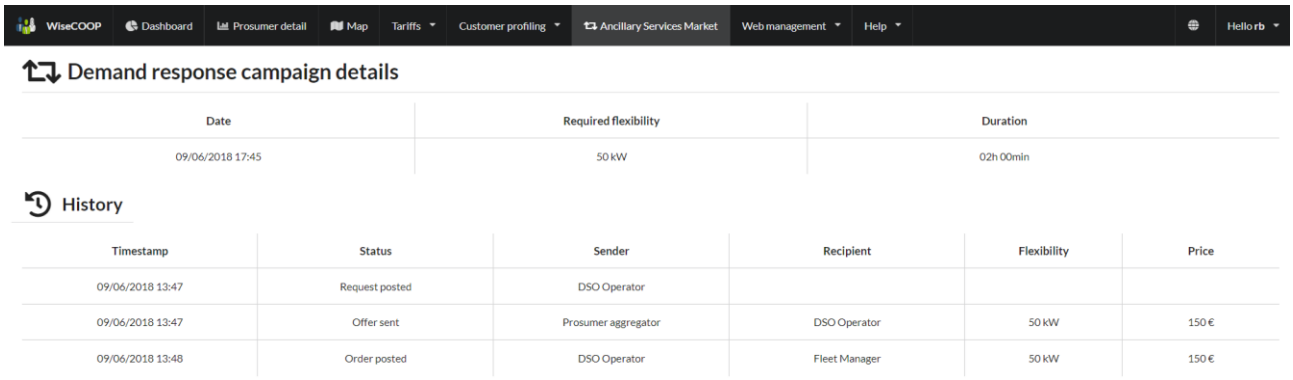


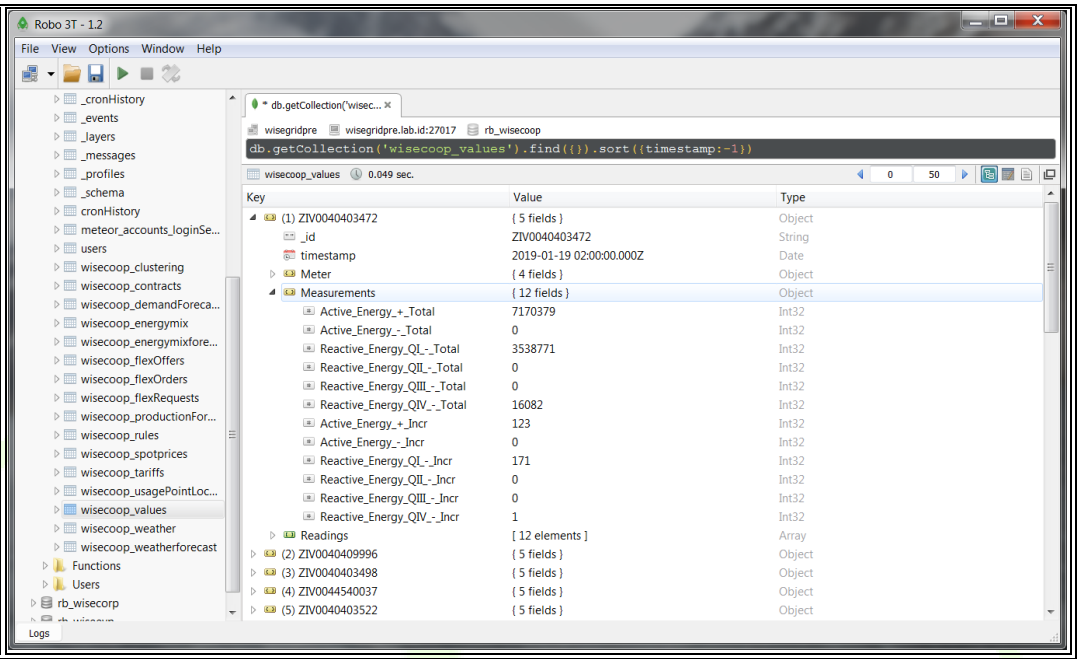

**Figure 26 – WiseCOOP – Details of an explicit demand response campaigns**

## 3.2 LAB-TESTING RESULTS

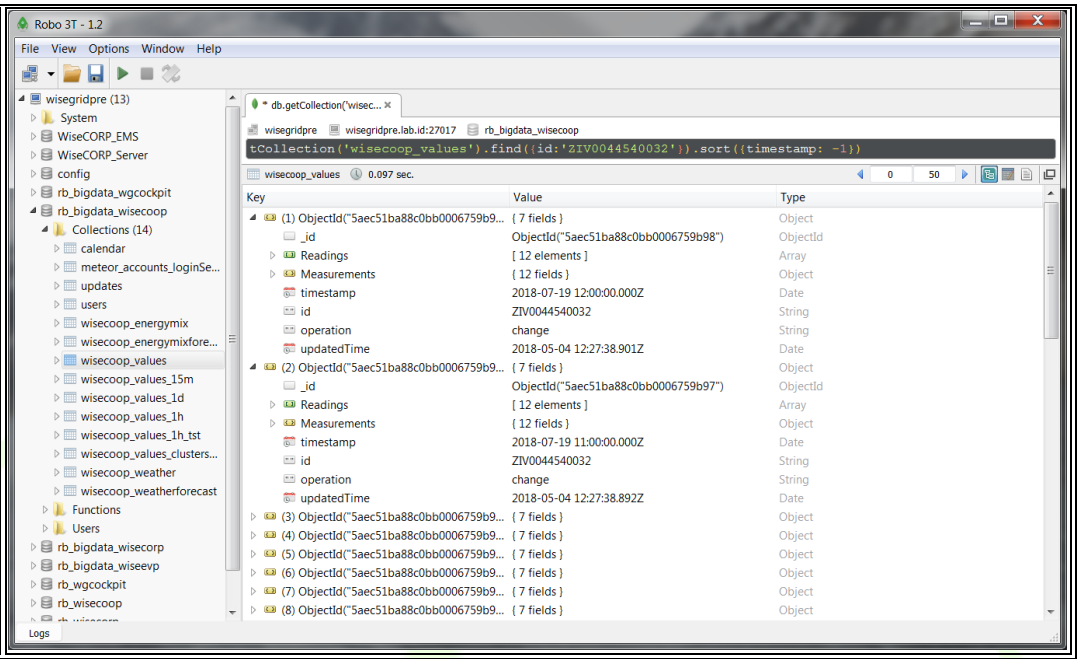
This section contains a set of templates with the definition, objectives, steps and results of all tests executed during this period on the different modules of the tool.

### 3.2.1 RT monitor tests

| | | | |
|---|---|---|---|
| **Name** | RTM001. Read smart meter data from IOP | | |
| **Module under test** | RT Monitor | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running | | |
| **Features to be tested** | Data from SMX is properly collected in the operational database of WiseCOOP | | |
| **Features not to be tested** | | | |
| **Preparation** | Configure one SMX to send data to the lab-testing IOP environment | | |
| **Dependencies** | | | |
| **Steps** | The execution of this test must happen automatically upon publication of data in the IOP | | |
| **Pass criteria** | Data from the SMX is correctly updated in the operational database. Operational database keeps a register of the last values sent by the SMX. | | |
| **Suspension criteria** | | | |
| **Results** | Test successful.<br>The following screenshot shows how the operational DB is populated with data from lab-testing environment SMX | | |

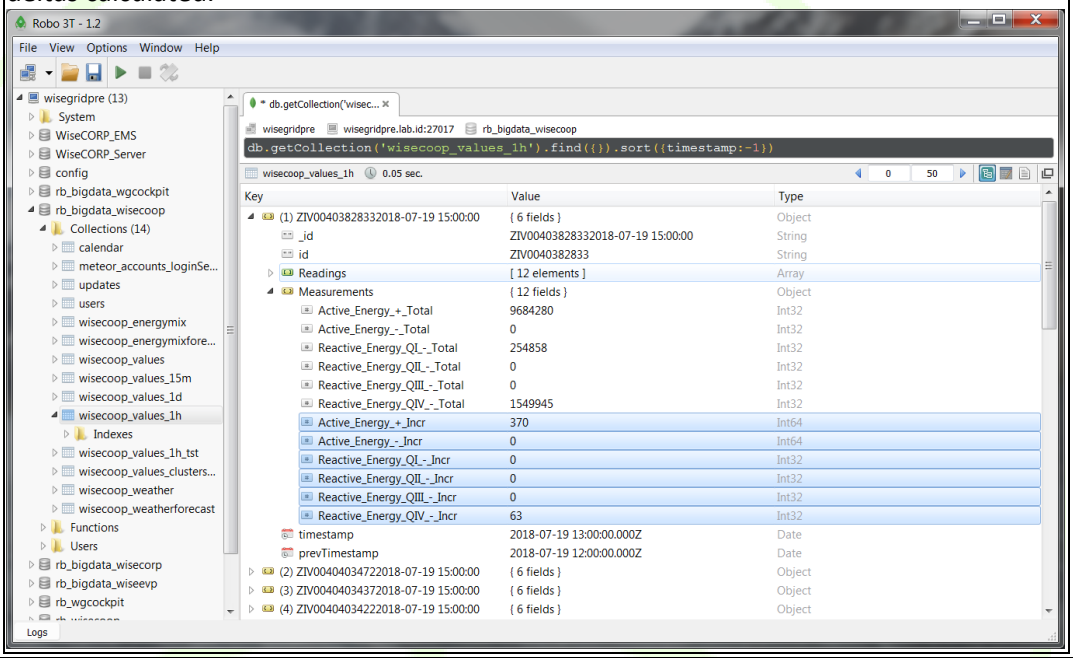
| | |
|---|---|
| |  |

| | | | |
|---|---|---|---|
| **Name** | RTM002. Store smart meter data to long-term DB | | |
| **Module under test** | RT Monitor | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running | | |
| **Features to be tested** | Data from SMX is properly collected in the long-term database of WiseCOOP (big data) | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |
| **Dependencies** | RTM001. Read smart meter data from IOP | | |
| **Steps** | The execution of this test must happen automatically upon publication of data in the IOP | | |
| **Pass criteria** | Data from the SMX is correctly appended to the historic registry held in the long-term database | | |
| **Suspension criteria** | | | |
| **Results** | Test successful.<br>The following screenshot shows how the long-term DB is populated with the history data from a lab-testing environment SMX | | |

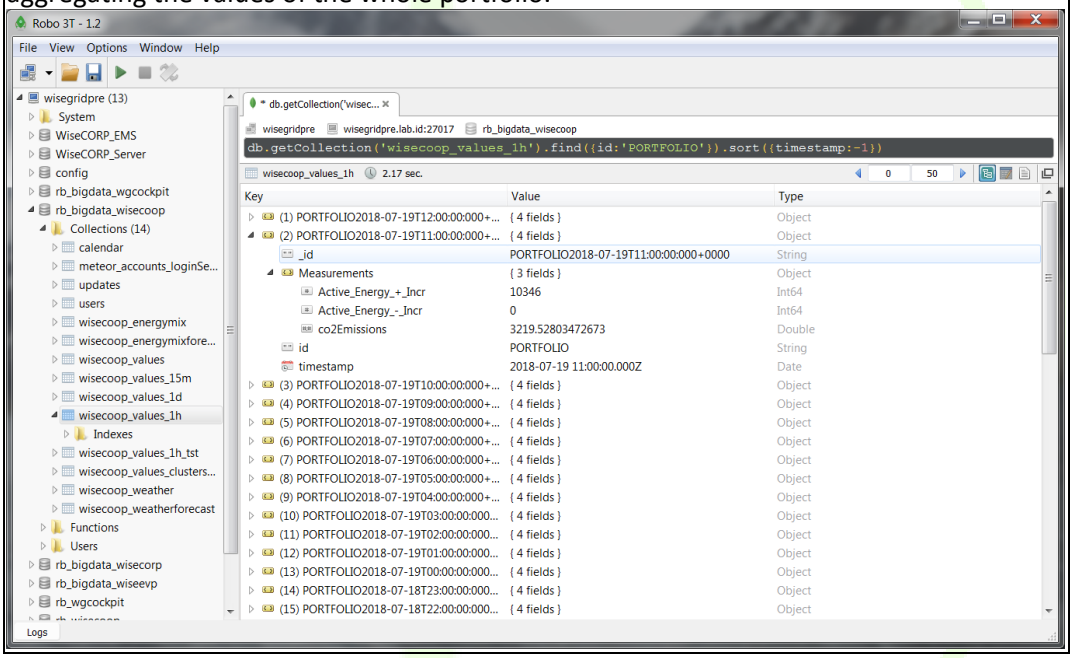



### 3.2.2 KPI engine tests

| | | | |
|---|---|---|---|
| **Name** | KPI001. Individual energy delta calculation | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running<br>RT monitor storing smart meter readings in long-term database<br>Spark server up and running | | |
| **Features to be tested** | Smart meters provide information of the total accumulated energy demand/production. The system therefore needs to calculate the energy deltas across consecutive readings in order to properly monitor the energy demand/production profiles.<br>Three different aggregations of the deltas are considered: quarterly, hourly and daily. | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |
| **Dependencies** | | | |
| **Steps** | 1. Execute Spark job for quarterly delta calculation<br>2. Execute Spark job for hourly delta calculation<br>3. Execute Spark job for daily delta calculation<br>4. Manually inspect long-term database collections with processed data | | |

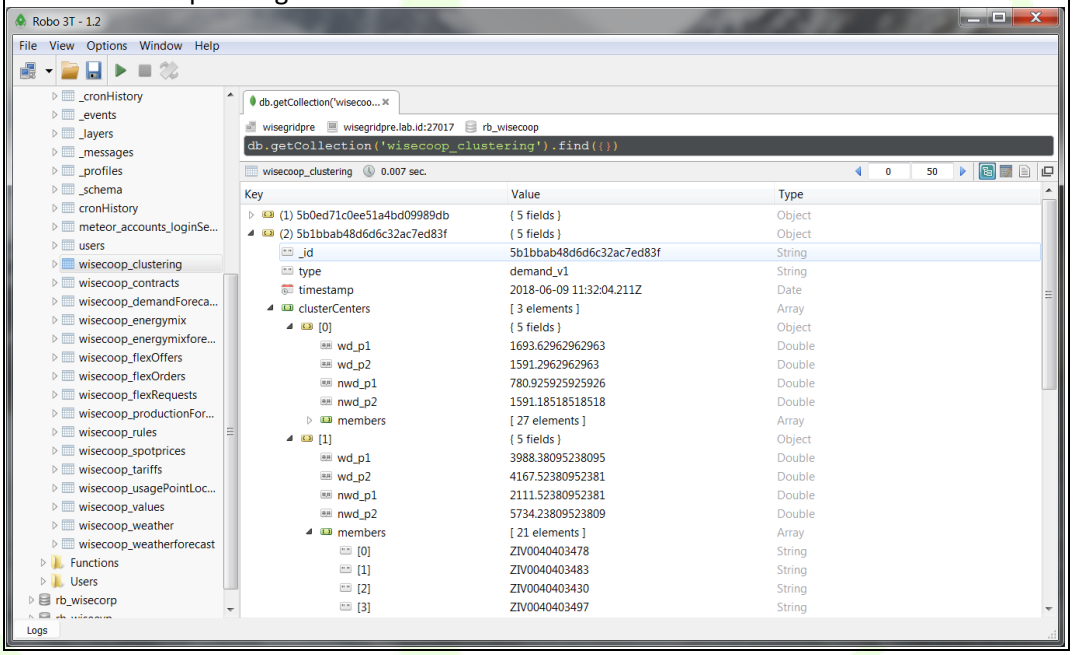
| | |
|---|---|
| **Pass criteria** | - The long-term database contains 3 collections with the quarterly, hourly and daily aggregations<br>- Each collection contains documents that represent the energy deltas for the given period |
| **Suspension criteria** | |
| **Results** | Test successful.<br>The following screenshot shows the three created collections (wisecoop_values_15m\|1h\|1d) as well as an example of the active and reactive energy deltas calculated.<br>  |

| | | | |
|---|---|---|---|
| **Name** | KPI002. Portfolio data aggregation | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running<br>RT monitor storing smart meter readings in long-term database<br>Spark server up and running | | |
| **Features to be tested** | An overview of the demand and production metrics for the whole portfolio is required | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |

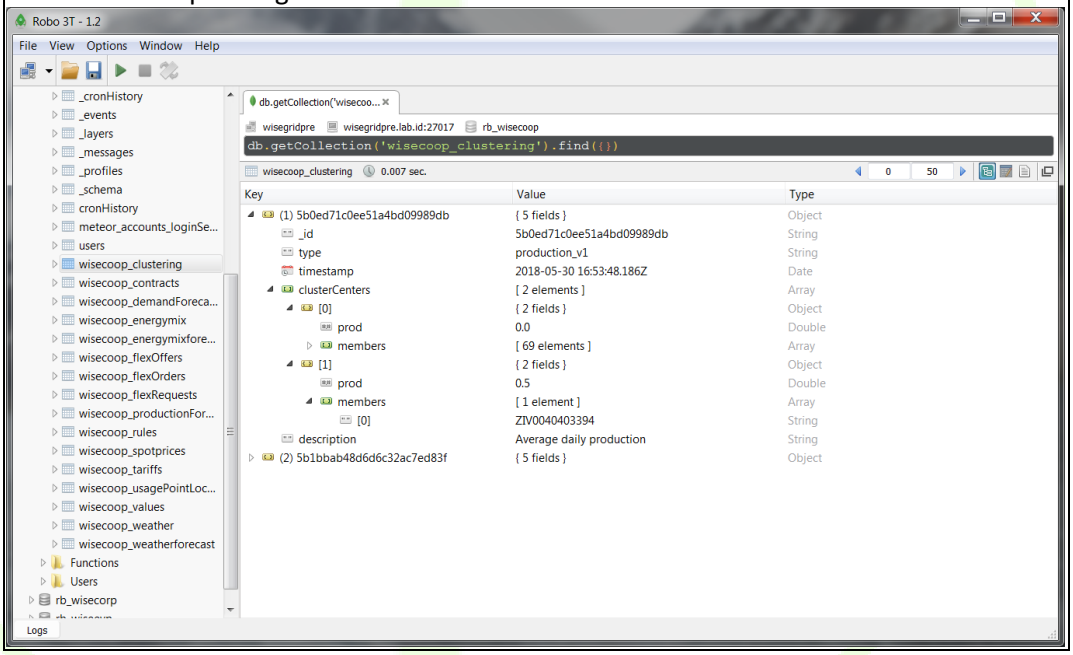
| | |
|---|---|
| **Dependencies** | KPI001. Individual energy delta calculation |
| **Steps** | 1. Execute Spark job for quarterly delta calculation (portfolio)<br>2. Execute Spark job for hourly delta calculation (portfolio)<br>3. Execute Spark job for daily delta calculation (portfolio)<br>4. Manually inspect long-term database collections with processed data |
| **Pass criteria** | - The long-term database contains 3 collections with the quarterly, hourly and daily aggregations<br>- Each collection contains documents for a virtual smart meter named "PORTFOLIO" that represent the energy deltas for the given period of the overall portfolio |
| **Suspension criteria** | |
| **Results** | Test successful.<br>The following screenshot shows an example of a virtual smart meter, with id PORTFOLIO, aggregating the values of the whole portfolio.<br>  |



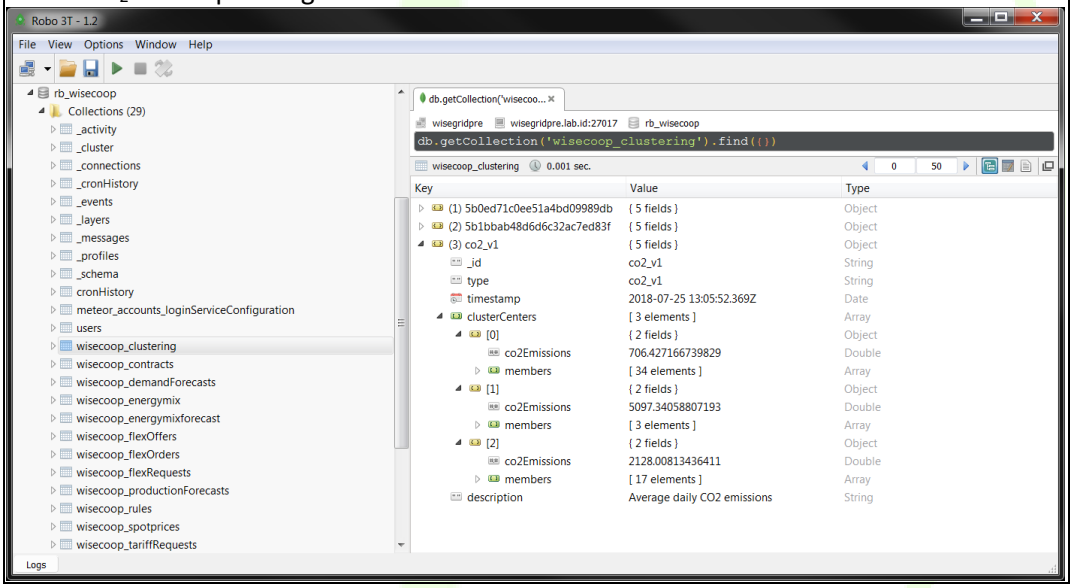
| | | | |
|---|---|---|---|
| **Name** | KPI003. Portfolio profiling, demand-related behaviour | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | Historical readouts for portfolio members stored in the long-term database<br>Spark server up and running | | |
| **Features to be tested** | All portfolio members get classified based on the distribution of the energy demand in the following time slots: | | |

| | |
|---|---|
| | - Working days, 08h-18h<br>- Working days, 18h-08h<br>- Non-working days, 08h-18h<br>- Non-working days, 18h-08h |
| **Features not to be tested** | |
| **Preparation** | |
| **Dependencies** | KPI001. Individual energy delta calculation |
| **Steps** | 1. Execute Spark job for clusters calculation<br>2. Manually inspect operational database clusters collection for results |
| **Pass criteria** | - The operational database contains the cluster centres for each identified group<br>- The operational database contains the list of members for each identified group |
| **Suspension criteria** | |
| **Results** | Test successful.<br>The following screenshot of the operational database shows the results of the execution of the demand profiling module.<br>  |

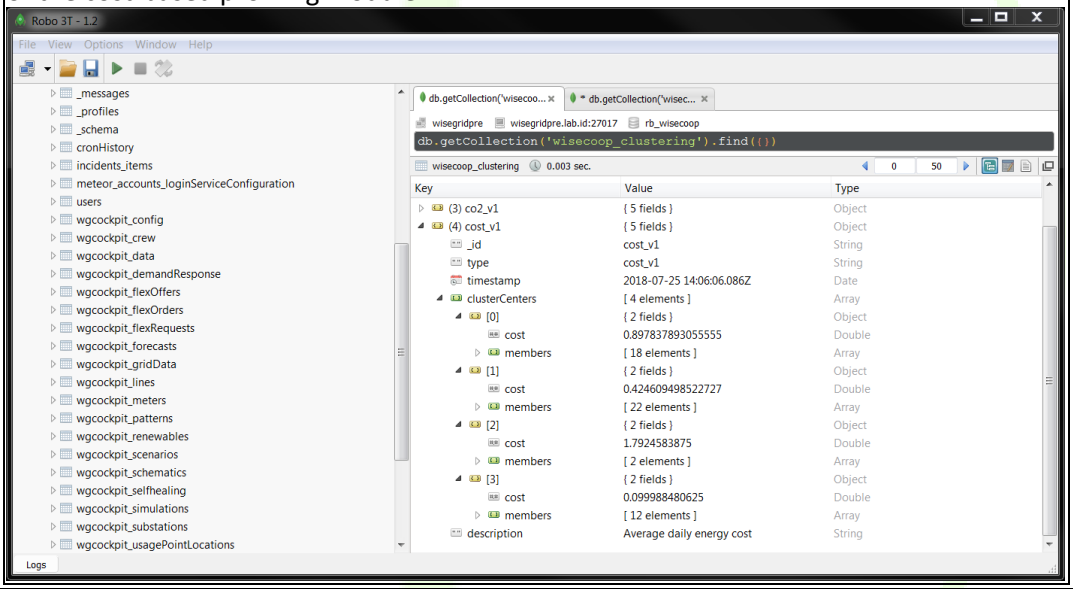
| | | | |
|---|---|---|---|
| **Name** | KPI004. Portfolio profiling, production-related behaviour | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test** | Historical readouts for portfolio members stored in the long-term database | | |

| | |
|---|---|
| **environment** | Spark server up and running |
| **Features to be tested** | All portfolio members get classified based on their daily production |
| **Features not to be tested** | |
| **Preparation** | |
| **Dependencies** | KPI001. Individual energy delta calculation |
| **Steps** | 1. Execute Spark job for clusters calculation<br>2. Manually inspect operational database clusters collection for results |
| **Pass criteria** | - The operational database contains the cluster centres for each identified group (average daily production)<br>- The operational database contains the list of members for each identified group |
| **Suspension criteria** | |
| **Results** | Test successful.<br>The following screenshot of the operational database shows the results of the execution of the demand profiling module.<br>  |

| | | | |
|---|---|---|---|
| **Name** | KPI005. Portfolio profiling, monthly $CO_2$ emissions | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test** | Historical readouts for portfolio members stored in the long-term database | | |

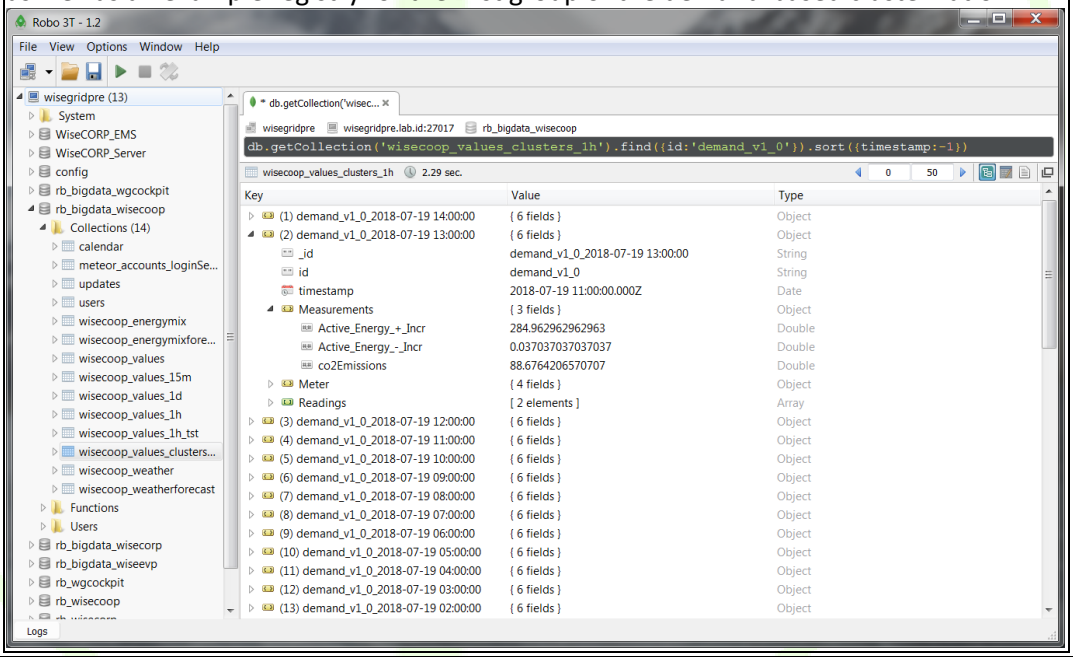
| | |
|---|---|
| **environment** | Spark server up and running |
| **Features to be tested** | All portfolio members get classified based on the monthly equivalent $CO_2$ emissions |
| **Features not to be tested** | |
| **Preparation** | |
| **Dependencies** | KPI001. Individual energy delta calculation |
| **Steps** | 1. Execute Spark job for clusters calculation<br>2. Manually inspect operational database clusters collection for results |
| **Pass criteria** | - The operational database contains the cluster centres for each identified group (average monthly equivalent $CO_2$ emissions)<br>- The operational database contains the list of members for each identified group |
| **Suspension criteria** | |
| **Results** | Test successful.<br>The following screenshot of the operational database shows the results of the execution of the $CO_2$-based profiling module.<br>  |

| | | | |
|---|---|---|---|
| **Name** | KPI006. Portfolio profiling, monthly cost | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | Historical readouts for portfolio members stored in the long-term database<br>Spark server up and running | | |

| | |
|---|---|
| **Features to be tested** | All portfolio members get classified based on the economic cost associated to their demand |
| **Features not to be tested** | |
| **Preparation** | |
| **Dependencies** | KPI001. Individual energy delta calculation |
| **Steps** | 1. Execute Spark job for clusters calculation<br>2. Manually inspect operational database clusters collection for results |
| **Pass criteria** | - The operational database contains the cluster centres for each identified group (average monthly costs)<br>- The operational database contains the list of members for each identified group |
| **Suspension criteria** | |
| **Results** | Test successful.<br>The following screenshot of the operational database shows the results of the execution of the cost-based profiling module.<br>  |



| | | | |
|---|---|---|---|
| **Name** | KPI007. Calculation of average profiles per clusterization and cluster | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | Historical readouts for portfolio members stored in the long-term database<br>Spark server up and running | | |
| **Features to be tested** | For each group of each clusterization, the *average member* is computed, in order to enable comparison of individual behaviour with the group of similar individuals | | |

| | |
|---|---|
| **Features not to be tested** | |
| **Preparation** | |
| **Dependencies** | KPI001. Individual energy delta calculation<br>KPI003. Portfolio profiling, demand-related behaviour<br>KPI004. Portfolio profiling, production-related behaviour<br>KPI005. Portfolio profiling, monthly $CO_2$ emissions<br>KPI006. Portfolio profiling, monthly cost |
| **Steps** | 1. Periodically execute Spark job for clusters average profile calculation<br>2. Manually inspect long-term database clusters collection for results |
| **Pass criteria** | - The long-term database contains registries for one virtual smart meter per clusterization and group<br>- Those smart meter contain the averaged values for energy demand, production, cost and emissions of all members of the group |
| **Suspension criteria** | |
| **Results** | Test successful.<br>The following screenshot of the long-term database shows the collection with the results, as well as an example registry for the first group of the demand-based clusterization.<br>  |



### 3.2.3 Forecast modules tests

| | | | |
|---|---|---|---|
| **Name** | FOR001. Demand/production forecasting training | | |
| **Module under test** | WiseCOOP forecast module | **Resp.** | ITE |

| | |
|---|---|
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market |
| **Test environment** | WiseCOOP forecast module up and running<br>Historical data available in long-term DB |
| **Features to be tested** | WiseCOOP forecast module is trained |
| **Features not to be tested** | |
| **Preparation** | |
| **Dependencies** | RTM002 |
| **Steps** | Perform WiseCOOP forecast training |
| **Pass criteria** | Training MAPE below pre-defined threshold |
| **Suspension criteria** | |
| **Results** | Test successful.<br>WiseCOOP demand forecast model trained |

| | | | |
|---|---|---|---|
| **Name** | FOR002. Demand/Production forecasting | | |
| **Module under test** | WiseCOOP forecast module | **Resp.** | ITE |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | WiseCOOP forecast module up and running<br>Historical data available in long-term DB | | |
| **Features to be tested** | WiseCOOP forecast module performs demand/production forecasting training | | |
| **Features not to be tested** | | | |
| **Preparation** | Train WiseCOOP demand/production forecast module | | |
| **Dependencies** | FOR001<br>RTM002 | | |
| **Steps** | WiseCOOP forecast module | | |
| **Pass criteria** | Prediction MAPE below pre-defined threshold | | |
| **Suspension criteria** | | | |
| **Results** | Test successful.<br>24 hours hourly aggregated load and production prediction | | |

| | |
|---|---|
| **Name** | FOR003. Request message parsing test of WiseCOOP forecast module |

| | | | |
|---|---|---|---|
| **Module under test** | WiseCOOP forecast module | **Resp.** | ITE |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | Development RabbitMQ environment<br>WiseCOOP forecast module up and running<br>Historical data available in long-term DB | | |
| **Features to be tested** | Performance of WiseCOOP forecast module, at parsing forecast queries. | | |
| **Features not to be tested** | | | |
| **Preparation** | Enable RabbitMQ queues, and run WiseCOOP forecast module | | |
| **Dependencies** | | | |
| **Steps** | 1. Receipt of request<br>2. Request parsing<br>3. DB request according to the requested data<br>4. Treatment of the retrieved data | | |
| **Pass criteria** | - The forecast module is able to decode the queries properly<br>- The forecast module is able to retrieve information from the long-term DB with the parsed information | | |
| **Suspension criteria** | | | |
| **Results** | Test successful.<br>The module is able to parse the request messages and process it to retrieve information from the long-term DB. | | |

| | | | |
|---|---|---|---|
| **Name** | FOR004. Forecast response message generation test of WiseCOOP forecast module | | |
| **Module under test** | WiseCOOP forecast module | **Resp.** | ITE |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | RabbitMQ environment<br>WiseCOOP forecast module up and running<br>Historical data available in long-term DB | | |
| **Features to be tested** | Performance of WiseCOOP forecast module, at generating and submitting the demand forecast response. | | |
| **Features not to be tested** | | | |
| **Preparation** | Enable RabbitMQ queues<br>Run the demand forecast module | | |

| | |
|---|---|
| **Dependencies** | |
| **Steps** | 1. Parsing of the forecasting algorithm output<br>2. Generating forecast response message |
| **Pass criteria** | - The forecast module is able to analyse properly the output provided by the forecasting algorithm<br>- The forecast module is able to generate properly the forecast response message |
| **Suspension criteria** | |
| **Results** | Test successful.<br>The module is able to analyse the information provided by the forecast algorithm, and generates the response. |

| | | | |
|---|---|---|---|
| **Name** | FOR005. Forecast is periodically triggered | | |
| **Module under test** | Forecast orchestrator | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | Internal ESB up and running<br>Historical data available in long-term DB | | |
| **Features to be tested** | WiseCOOP periodically posts a demand and a production forecast request per bus to the corresponding queue of the internal ESB | | |
| **Features not to be tested** | | | |
| **Preparation** | Open RabbitMQ monitor, monitor demand and production forecast queues | | |
| **Dependencies** | | | |
| **Steps** | 1. Execute forecast orchestrator module | | |
| **Pass criteria** | - Periodically, every hour, one request per smart meter appears in the demand and production forecast queues<br>- Requests claim next 24 hours hourly prediction | | |
| **Suspension criteria** | | | |
| **Results** | Test successful<br>The following extract of logs of the Docker container wisecoop_forecastbridge_demand_1 shows that one forecast query for each asset is being posted every hour.<br>`etraid@wisegridpre:~$ docker logs -t --tail 100 wisecoop_forecastbridge_demand_1 \| grep querying \| grep ZIV0036406533` | | |

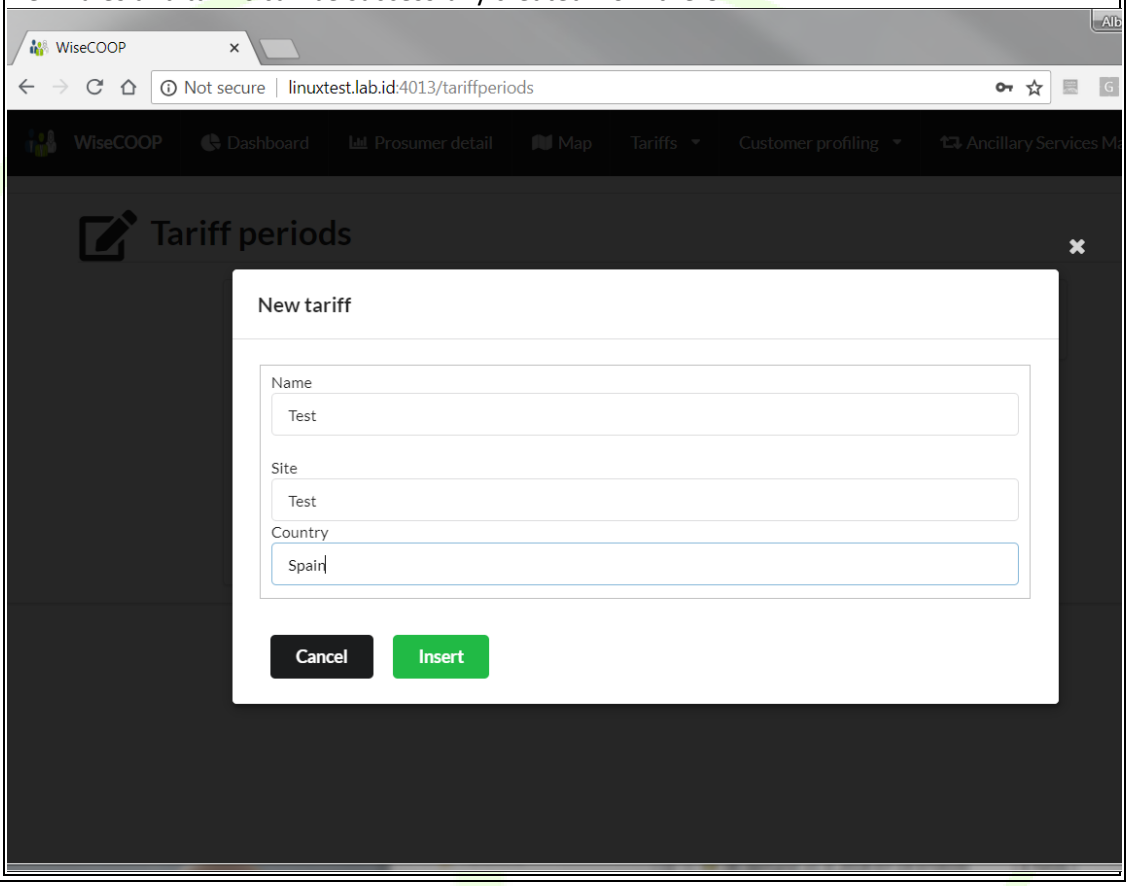
| | |
|---|---|
| | 2018-07-18T13:37:55.132492293Z [ZIV0036406533] querying...<br>2018-07-18T14:37:55.420413088Z [ZIV0036406533] querying...<br>2018-07-18T15:37:56.411212047Z [ZIV0036406533] querying...<br>2018-07-18T16:37:54.949073578Z [ZIV0036406533] querying...<br>2018-07-18T17:37:54.528547460Z [ZIV0036406533] querying...<br>2018-07-18T18:37:55.309216318Z [ZIV0036406533] querying...<br>2018-07-18T19:37:54.674470058Z [ZIV0036406533] querying...<br>2018-07-18T20:37:54.560616957Z [ZIV0036406533] querying...<br>2018-07-18T21:37:54.313838874Z [ZIV0036406533] querying...<br>2018-07-18T22:37:54.998547669Z [ZIV0036406533] querying...<br>2018-07-18T23:37:55.138371328Z [ZIV0036406533] querying... |

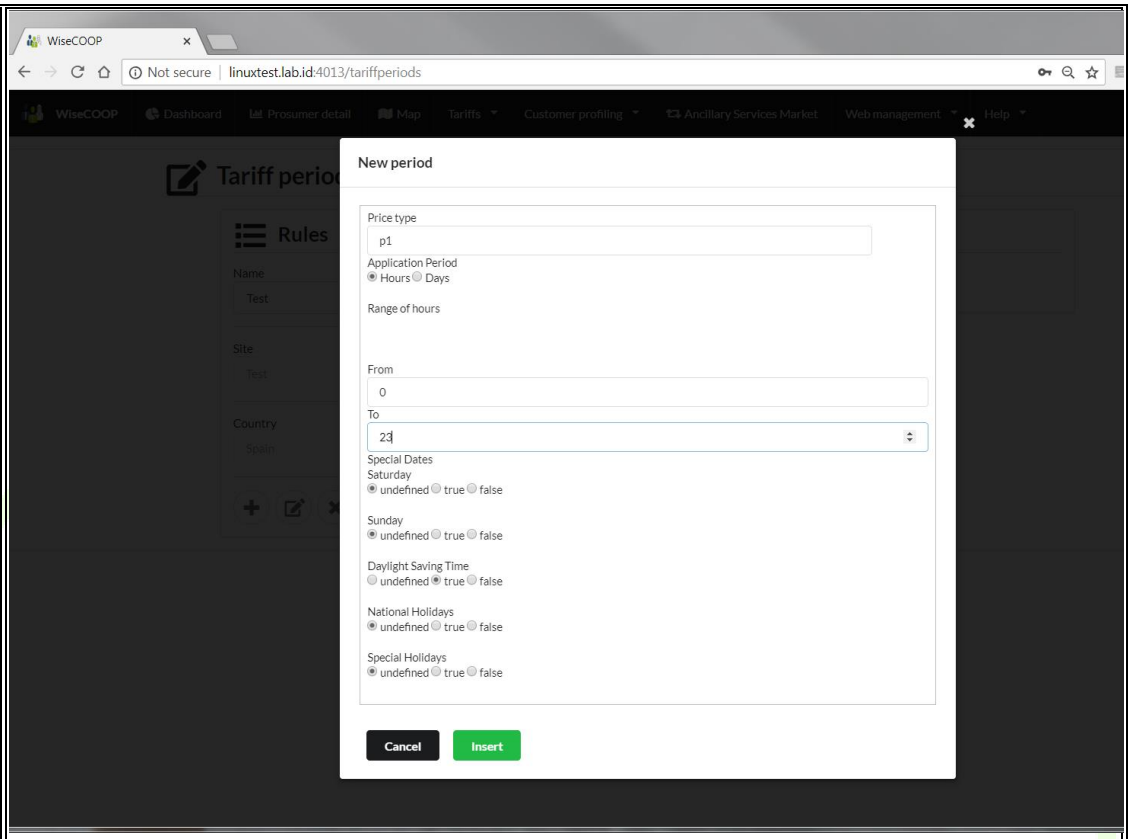
| | | | |
|---|---|---|---|
| **Name** | FOR006. Forecast results are saved to operational DB | | |
| **Module under test** | Forecast orchestrator | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | Internal ESB up and running<br>Historical data available in long-term DB<br>Demand and production forecast modules up and running | | |
| **Features to be tested** | WiseCOOP receives the results of the forecast module, formats them following the same format used to store real-time data, and stores the in the operational database | | |
| **Features not to be tested** | | | |
| **Preparation** | Open operational database, query next 24 hours of demand/production forecasts | | |
| **Dependencies** | | | |
| **Steps** | 1. Execute forecast orchestrator module | | |
| **Pass criteria** | Periodically, every hour, next 24 hours forecast metrics get updated in the operational database | | |
| **Suspension criteria** | | | |

| | |
|---|---|
| **Results** | Test successful<br>The following extract of logs of the Docker container wisecoop_forecastbridge_demand_1 shows that the forecast module answers to the posted requests.<br>`etraid@wisegridpre:~$ docker logs -t --tail 1000 wisecoop_forecastbridge_demand_1 \| grep "forecast received" \| grep ZIV0036406533`<br>`2018-07-18T16:40:36.196281873Z [ZIV0036406533] forecast received...`<br>`2018-07-18T17:40:38.580263496Z [ZIV0036406533] forecast received...`<br>`2018-07-18T18:40:37.195192766Z [ZIV0036406533] forecast received...`<br>`2018-07-18T19:40:39.462108051Z [ZIV0036406533] forecast received...`<br>`2018-07-18T20:40:41.464303878Z [ZIV0036406533] forecast received...`<br>`2018-07-18T21:40:37.059925621Z [ZIV0036406533] forecast received...`<br>`2018-07-18T22:40:22.290691821Z [ZIV0036406533] forecast received...`<br>`2018-07-18T23:40:10.960955447Z [ZIV0036406533] forecast received...` |

### 3.2.4 Tariff designer and comparer tests

| | | | |
|---|---|---|---|
| **Name** | TDC001. Create a tariff | | |
| **Module under test** | Tariff designer and comparer | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | WiseCOOP UI up and running | | |
| **Features to be tested** | WiseCOOP facilitates to retailers the ability to define tariffs from the UI | | |
| **Features not to be tested** | | | |
| **Preparation** | Open WiseCOOP UI<br>Open WiseCOOP operational database | | |
| **Dependenci** | | | |

| | |
|---|---|
| **es** | |
| **Steps** | 1. Define a new rule with the WiseCOOP UI<br>2. Define a new tariff associated to that rule with the WiseCOOP UI |
| **Pass criteria** | The database contains registries with the definition of both items |
| **Suspension criteria** | |
| **Results** | Test successful<br>New rules and tariffs can be successfully created from the UI<br>  |




| | | | |
|---|---|---|---|
| **Name** | TDC002. Edit existing tariff | | |
| **Module under test** | Tariff designer and comparer | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | WiseCOOP UI up and running | | |
| **Features to be tested** | WiseCOOP facilitates to retailers the ability to define tariffs from the UI | | |
| **Features not to be tested** | | | |
| **Preparation** | Open WiseCOOP UI<br>Open WiseCOOP operational database | | |
| **Dependencies** | TDC001. Create a tariff | | |
| **Steps** | 1. Edit a rule with the WiseCOOP UI<br>2. Edit a tariff associated to that rule with the WiseCOOP UI | | |
| **Pass criteria** | The database registries are modified accordingly | | |
| **Suspension criteria** | | | |

| | |
|---|---|
| **Results** | Test successful<br>Editing of tariffs is reflected in the database |

### 3.2.5 DR framework tests

| | | | |
|---|---|---|---|
| **Name** | DRF001. Obtain portfolio demand forecast | | |
| **Module under test** | Elasticity estimation | **Resp.** | HYP |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | WiseCOOP up and running, including DB and forecast orchestrator | | |
| **Features to be tested** | Retrieval of the aggregated day-ahead demand of the retailer portfolio from the long-term DB of the WiseCOOP tool | | |
| **Features not to be tested** | | | |
| **Preparation** | Launch WiseCOOP, including its internal ESB, long-term DB and Forecast Orchestrator | | |
| **Dependencies** | FOR005 | | |
| **Steps** | 1. Request demand forecast from long-term DB via internal ESB<br>2. Listen to queue for response message | | |
| **Pass criteria** | Reception of portfolio demand forecast | | |
| **Suspension criteria** | | | |
| **Results** | Test successful | | |

| | | | |
|---|---|---|---|
| **Name** | DRF002. Obtain generation forecast | | |
| **Module under test** | Elasticity estimation | **Resp.** | HYP |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | WiseCOOP up and running, including DB and forecast orchestrator | | |
| **Features to be tested** | Retrieval of the day-ahead generation forecast of the retailer portfolio from the long-term DB of the WiseCOOP tool | | |
| **Features not to be tested** | | | |
| **Preparation** | Launch WiseCOOP, including its internal ESB, long-term DB and Forecast Orchestrator | | |

| | |
|---|---|
| **Dependencies** | FOR005 |
| **Steps** | 1. Request generation forecast from long-term DB via internal ESB<br>2. Listen to queue for response message |
| **Pass criteria** | Reception of generation forecast message |
| **Suspension criteria** | |
| **Results** | Test successful |

| | | | |
|---|---|---|---|
| **Name** | DRF003. Price signal calculation | | |
| **Module under test** | Price calculator | **Resp.** | HYP |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | WiseCOOP & IOP up and running | | |
| **Features to be tested** | Estimation of the day-ahead 24-hour retail electricity price forecast based on the availability of renewable generation | | |
| **Features not to be tested** | | | |
| **Preparation** | Launch WiseCOOP & IOP | | |
| **Dependencies** | DRF001, DRF002 | | |
| **Steps** | 1. Estimate the imbalance time-series at the level of the retailer portfolio<br>2. Fix discrete price levels<br>3. Estimate retail price per hour based on the elasticity of demand and the projected portfolio imbalance at the specific time interval | | |
| **Pass criteria** | Generation of retail price time signal with 24 entries. Low prices should correspond to time intervals with high renewable generation and vice versa. | | |
| **Suspension criteria** | | | |
| **Results** | Test successful | | |

| | | | |
|---|---|---|---|
| **Name** | DRF004. Dispatch price signal | | |
| **Module under test** | Price calculator | **Resp.** | HYP |
| **Module** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and | | |

| | |
|---|---|
| **requirement** | active participation into energy market |
| **Test environment** | WiseCOOP & IOP up and running |
| **Features to be tested** | Dispatching of the price signal (encoded in the format specified in D10.2) to a queue in the IOP so that other WiseGRID products can receive it. |
| **Features not to be tested** | |
| **Preparation** | Launch WiseCOOP & IOP |
| **Dependencies** | DRF003 |
| **Steps** | Push the price signal message to the specified queue of the IOP MQTT broker |
| **Pass criteria** | The message containing the price signal is available in the MQTT broker queue and any WiseGRID product listening to the specific queue can receive it. |
| **Suspension criteria** | |
| **Results** | Test successful |

| | | | |
|---|---|---|---|
| **Name** | DRF005. Elicitation of demand flexibility per building | | |
| **Module under test** | Aggregated flex estimation | **Resp.** | HYP |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | WiseCOOP, WiseCORP & IOP up and running | | |
| **Features to be tested** | Elicitation and collection of the demand flexibility potential from the all the buildings that belong to the portfolio of the WiseCOOP operator | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |
| **Dependencies** | | | |
| **Steps** | 1. Send message to the target buildings to elicit their demand flexibility potential<br>2. Listen for the response messages per WiseCORP instance<br>3. Message decoding and collection of flexibility time-series per building<br>4. Calculation of aggregated portfolio demand flexibility | | |
| **Pass criteria** | Generation of an aggregated demand flexibility potential profile that is in-line with the demand flexibility that can be provided per building | | |
| **Suspension criteria** | | | |
| **Results** | Test successful | | |

| | | | |
|---|---|---|---|
| **Name** | DRF006. Estimate & dispatch device commands | | |
| **Module under test** | Optimization & dispatcher | **Resp.** | HYP |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | WiseCOOP, WiseCORP & IOP up and running | | |
| **Features to be tested** | Breakdown of demand flexibility requested from the DSO into the optimal flexibility per building device, based on the information retrieved about the demand flexibility potential. | | |
| **Features not to be tested** | | | |
| **Preparation** | WiseCOOP up & running | | |
| **Dependencies** | DRF005 | | |
| **Steps** | 1. Collect demand flexibility per building<br>2. Analyse building-level demand flexibility based on eligibility criteria specified in the flexRequest<br>3. Calculate which building devices can deliver the target flexibility in the most optimal manner<br>4. Dispatch commands with demand modification per device | | |
| **Pass criteria** | Dispatch of request for demand modification that are according to the flex potential that has been specified by WiseCORP for each asset and which optimises a global objective function. | | |
| **Suspension criteria** | | | |
| **Results** | Test successful | | |

# 4 WISECORP APPLICATION

## 4.1 IMPLEMENTATION

### 4.1.1 Architecture overview

The WiseCORP architecture of the application finally implemented does not differ significantly from the architecture presented in the previous deliverable D7.1 WiseCOOP and WiseCORP Apps Design [4]. The architecture is summarized in this section for completeness of this document.

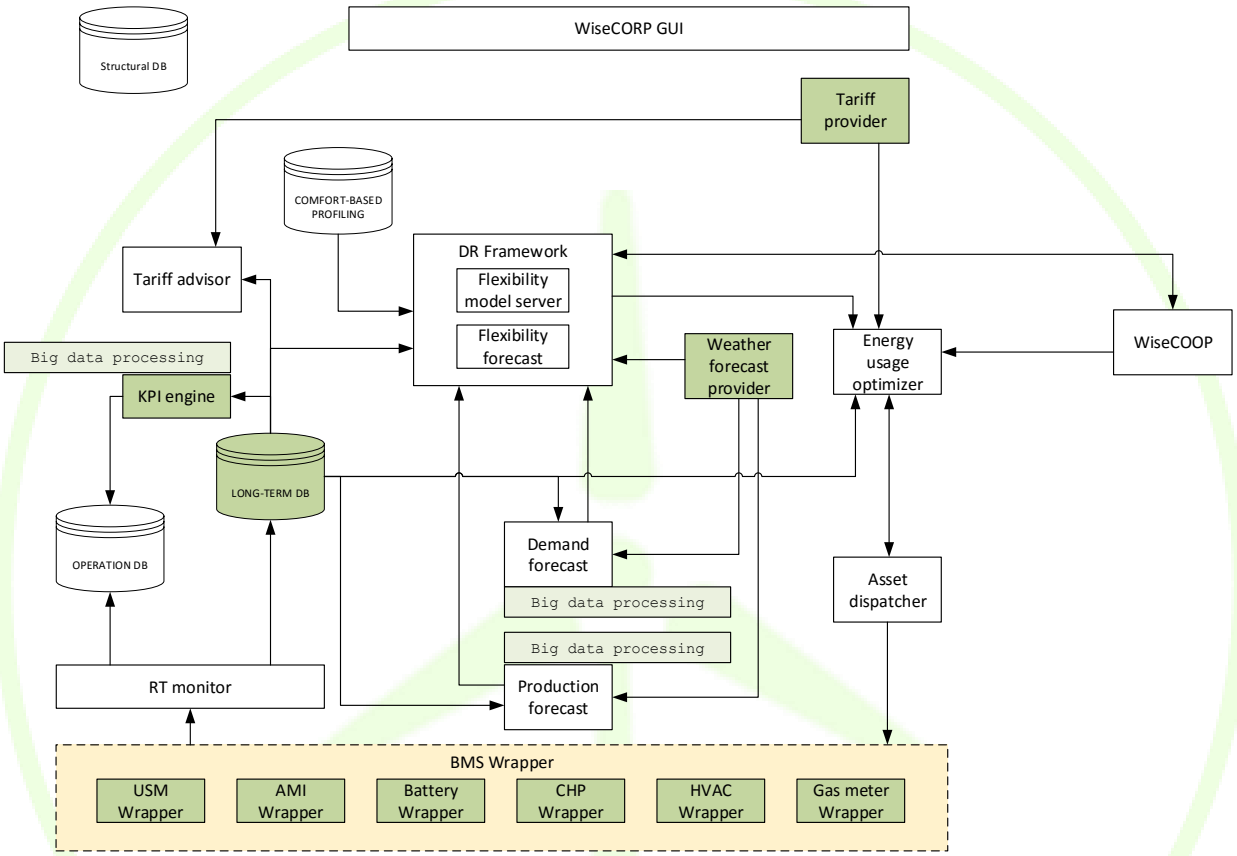



**Figure 27 – Overview of interactions among the modules of the WiseCORP application**

**Data ingestion**

The first step considered in the design of the application is the data ingestion. The procedure followed is common to other applications in the project, and implies the following steps:

1. Publication of data from *Wrappers* to the *WiseGRID IOP Message Broker*. Following the principle taken in the overall project, data sources publish data to the Interoperable Platform, allowing different application with the corresponding permissions to access to those data flows
2. Subscription to data flows of interest. In the case of the WiseCORP, these data flows include two main types: energy readings of the building (both demand and production, possibly including sub-metering), and status from sensors and controllable assets within the building. This subscription is performed by the *RT monitor* module
3. Store data for further analysis. The *RT monitor* module is in charge of populating both the *Operation* and the *Long-term DB* for further analysis with the data received from the different sources

**Data analysis**

Under this group, different modules have been defined in order to process the raw data coming from the different data sources in order to get the relevant information out of those. These modules include:

- *KPI engine* module, in charge of extracting different indicators and patterns from the raw data, mainly related to the energy demand distribution in time to support the Facility Manager in the analysis of further actions to reduce demand, energy costs or to promote self-consumption in the facilities;
- *Demand and production forecast* module, providing forecasts for the buildings monitored by the tool.

**Operation and control**

Under this group, different modules have been defined implementing specific tasks in order to fulfil the different functional requirements of the application. These modules comprise:

- *Tariff advisor* module, allowing the facility managers to simulate their bills with different tariffs, thus comparing with historical data which tariffs are more beneficial;
- *Energy Usage Optimizer* module, which will elaborate the schedule of the different controllable assets of the building according to their energy requirements model, in order to optimize the objective of the facility manager – mainly reduce economic costs or environmental impact;
- *Asset Dispatcher* module, dealing with the communication with the different controllable assets and ensuring those follow the calculated schedule;
- *DR framework* module, implementing the complete set of functionalities required to enable the participation of the facility in explicit demand response campaigns that will be tested in the project, keeping the occupants' comfort in the core.

**Interaction with other applications**

The main interaction that will be implemented within WiseCORP will be with the WiseCOOP application. WiseCOOP participates in the Ancillary Services Market of the DSO to offer the flexibility of the members of its portfolio to support the DSO to maintain the quality of the supply in the distribution grid. The aggregator (using WiseCOOP) must therefore be able to select and send signals to the aggregated members to request them to shift their demand accordingly to meet the agreement with the DSO. The prosumers with more potential to offer significant modulation of their demand are those with bigger energy requirements, those targeted by the WiseCORP application.

**Horizontal and support functionalities**

Different modules will be used indirectly by the WiseCORP application. Summarizing, these modules are data providers that offer information needed by other modules of the application to fulfil their duties, which are reused among different applications developed within the project. The list includes the *Weather Forecast*, the *Tariff Provider* module, as well as the *Big Data platform* that will support the long-term storage and analysis. In addition, the *WiseCORP User Interface* is included in this category, providing web-based access to the information and functionalities provided by the other modules. Additionally, notification mechanisms (such as email, Telegram or Twitter) will be implemented in order to notify the facility manager of significant events occurring in the system (e.g. triggering of explicit demand response campaigns).

### 4.1.2 Back-office modules

#### 4.1.2.1 Internal Enterprise Service Bus

The description of this module it is also explained in Section 3.1.2.1.

#### 4.1.2.2 Asset dispatcher

The Asset Dispatcher module is a software module in charge of executing the appropriate schedule of the

controllable assets, by continuously comparing the actual setpoint executed by the assets with the planned one. The corresponding commands to configure the appropriate setpoints are triggered whenever deemed required, upon detection of a mismatch between the schedule and the status.

This module has been developed using the .NET Framework 4.5, and gets triggered every minute in order to check that the current set points of the assets (stored in the operational database by the *Real-time monitor*) correspond to the scheduled one (also stored in the operational database by the *Energy usage optimizer module* and the demand response framework). Commands are dispatched using the MQTT-based protocol defined in the D7.1 [4].

**Table 21 – Log of command dispatched by Asset dispatcher via MQTT**

```
>> [HVAC001/SHIC01/0-1-160-7-0-1] { "_id" : "0-1-160-7-0-1", "assetID" :
"HVAC001", "value" : "26", "unit" : "grdC", "status" : 1, "captureTime"
: ISODate("2018-07-18T14:35:16.379Z"), "description" : "modbus", "mode"
: "cooling", "command" : "auto", "state" : "manual" }
```

#### 4.1.2.3 Real-time monitor

As in WiseCOOP, the *Real Time monitor* is the horizontal module that will handle the data ingestion for most of the applications of the project. It has been designed in order to fulfil the requirements for data ingestion accordingly to the requirements and the architecture of communications proposed for the applications.

Particularly for WiseCORP, this module is in charge of tracking and storing in the databases of WiseCORP the data items shown in the following table.

**Table 22 – Data items tracked by Real-time monitor in WiseCORP application**

| Data item | Source | Operational DB | Long-term DB |
|---|---|---|---|
| **Energy mix** | ENTSOE energy mix provider | X | X |
| **Energy mix forecast** | ENTSOE energy mix provider | X | X |
| **Energy readings** | Field assets (SMX, AMI systems) | X | X |
| **Weather** | Weather forecast provider | X | X |
| **Weather forecast** | Weather forecast provider | X | X |
| **Asset status (HVAC, batteries, sensors, CHP…)** | Field assets | X | X |

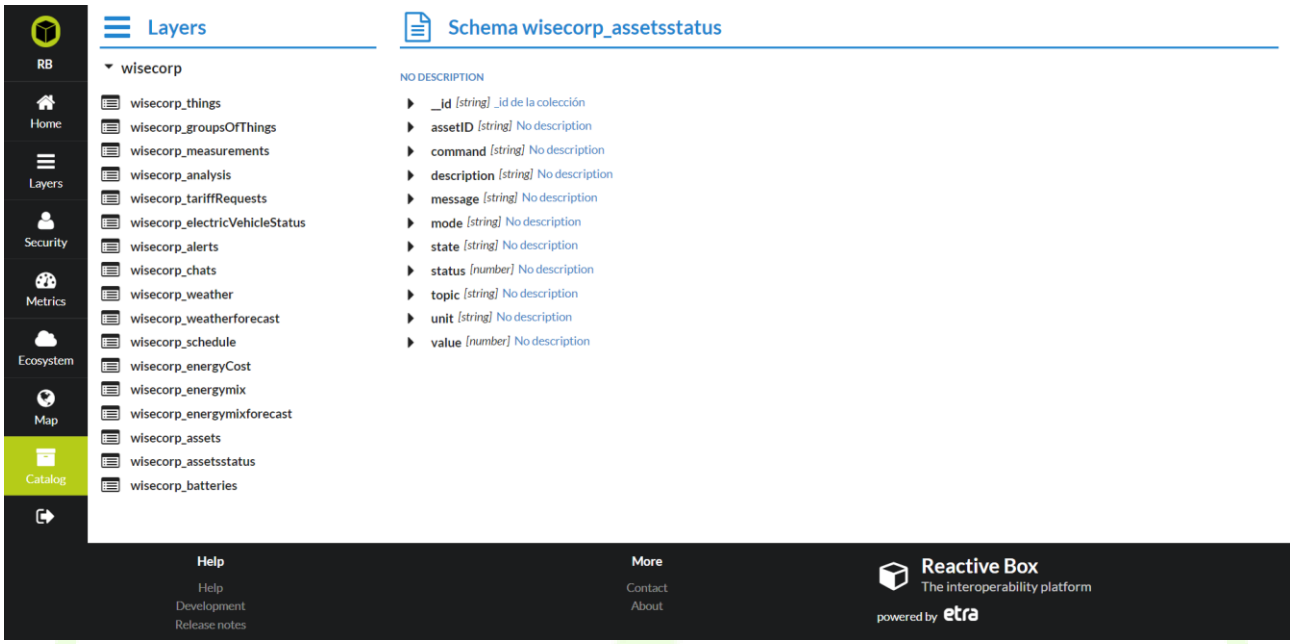


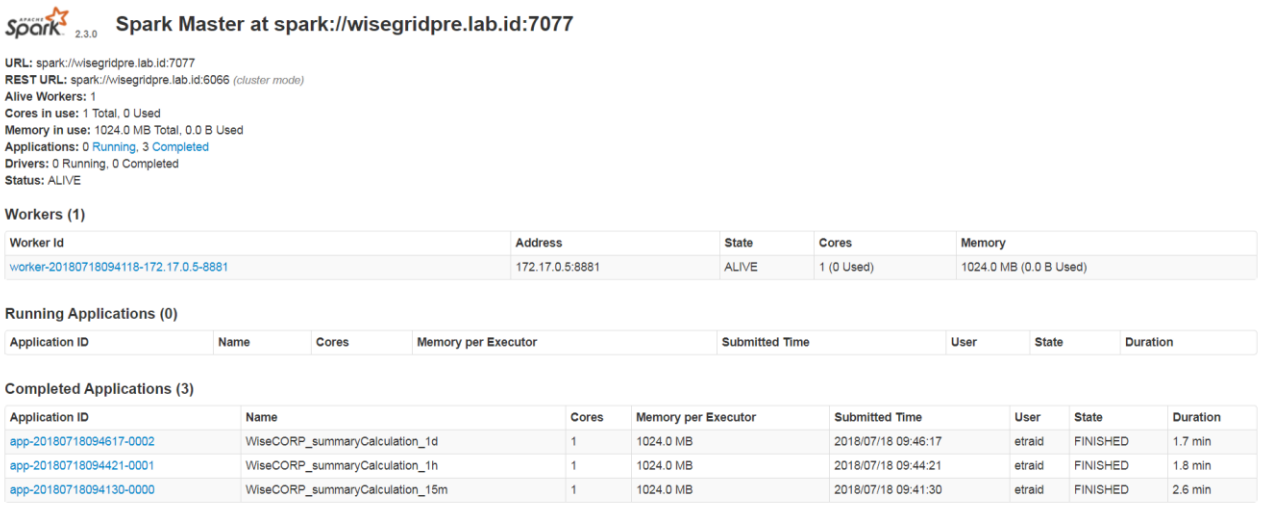

**Figure 28 – Screenshot of the Real-time monitor UI, showing asset status schema**

#### 4.1.2.4 KPI engine

The KPI engine of WiseCORP application has been implemented as a set of Spark jobs that are periodically triggered on the long-term database to perform the necessary calculations and push the results back to different collections of the database.

**Table 23 – WiseCORP – Spark jobs of the KPI engine**

| Spark job | Module | Description | Result KPIs |
|---|---|---|---|
| **WiseCORP summaryCalculator 15m** | summarycalculator-assembly-0.1.jar | Calculates aggregated registers for every meter and every 15 minutes | Energy demand<br>Energy production<br>Equivalent $CO_2$ emissions<br>Associated cost |
| **WiseCORP summaryCalculator 1h** | summarycalculator-assembly-0.1.jar | Calculates aggregated registers for every meter and every hour | Energy demand<br>Energy production<br>Equivalent $CO_2$ emissions<br>Associated cost |
| **WiseCORP summaryCalculator 1d** | summarycalculator-assembly-0.1.jar | Calculates aggregated registers for every meter and every day | Energy demand<br>Energy production<br>Equivalent $CO_2$ emissions<br>Associated cost |



Spark Master at spark://wisegridpre.lab.id:7077

URL: spark://wisegridpre.lab.id:7077
REST URL: spark://wisegridpre.lab.id:6066 (cluster mode)
Alive Workers: 1
Cores in use: 1 Total, 0 Used
Memory in use: 1024.0 MB Total, 0.0 B Used
Applications: 0 Running, 3 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (1)

| Worker Id | Address | State | Cores | Memory |
|---|---|---|---|---|
| worker-20180718094118-172.17.0.5-8881 | 172.17.0.5:8881 | ALIVE | 1 (0 Used) | 1024.0 MB (0.0 B Used) |

Running Applications (0)

| Application ID | Name | Cores | Memory per Executor | Submitted Time | User | State | Duration |
|---|---|---|---|---|---|---|---|

Completed Applications (3)

| Application ID | Name | Cores | Memory per Executor | Submitted Time | User | State | Duration |
|---|---|---|---|---|---|---|---|
| app-20180718094617-0002 | WiseCORP_summaryCalculation_1d | 1 | 1024.0 MB | 2018/07/18 09:46:17 | etraid | FINISHED | 1.7 min |
| app-20180718094421-0001 | WiseCORP_summaryCalculation_1h | 1 | 1024.0 MB | 2018/07/18 09:44:21 | etraid | FINISHED | 1.8 min |
| app-20180718094130-0000 | WiseCORP_summaryCalculation_15m | 1 | 1024.0 MB | 2018/07/18 09:41:30 | etraid | FINISHED | 2.6 min |

**Figure 29 – Screenshot of Spark server with executed WiseCORP jobs**

#### 4.1.2.5 Demand and production forecast service

This module of WiseCORP has been implemented over a RPC server which makes use of the ESB. In addition, this module makes use of the long-term database of WiseCORP, which is implemented over a MongoDB database. The RPC servers (demand and generation forecast) of this module are permanently running to manage the received queries through the RabbitMQ queues enabled to make use of the demand and production forecast.

Within the message queries are specified the ID of the supply point, and the period and the horizon of the desired forecast. In the case of production forecast, in addition of the defined fields, it is specified the type of generation technology.

Once the query is de-serialized and parsed, the forecast module retrieves from the long-term database the necessary information to perform the forecast. To perform the forecast it is retrieved information related to the consumed/produced energy, working calendar and weather information related to the queried installation, being this information available in the long-term database, as it is possible to appreciate in the next picture.

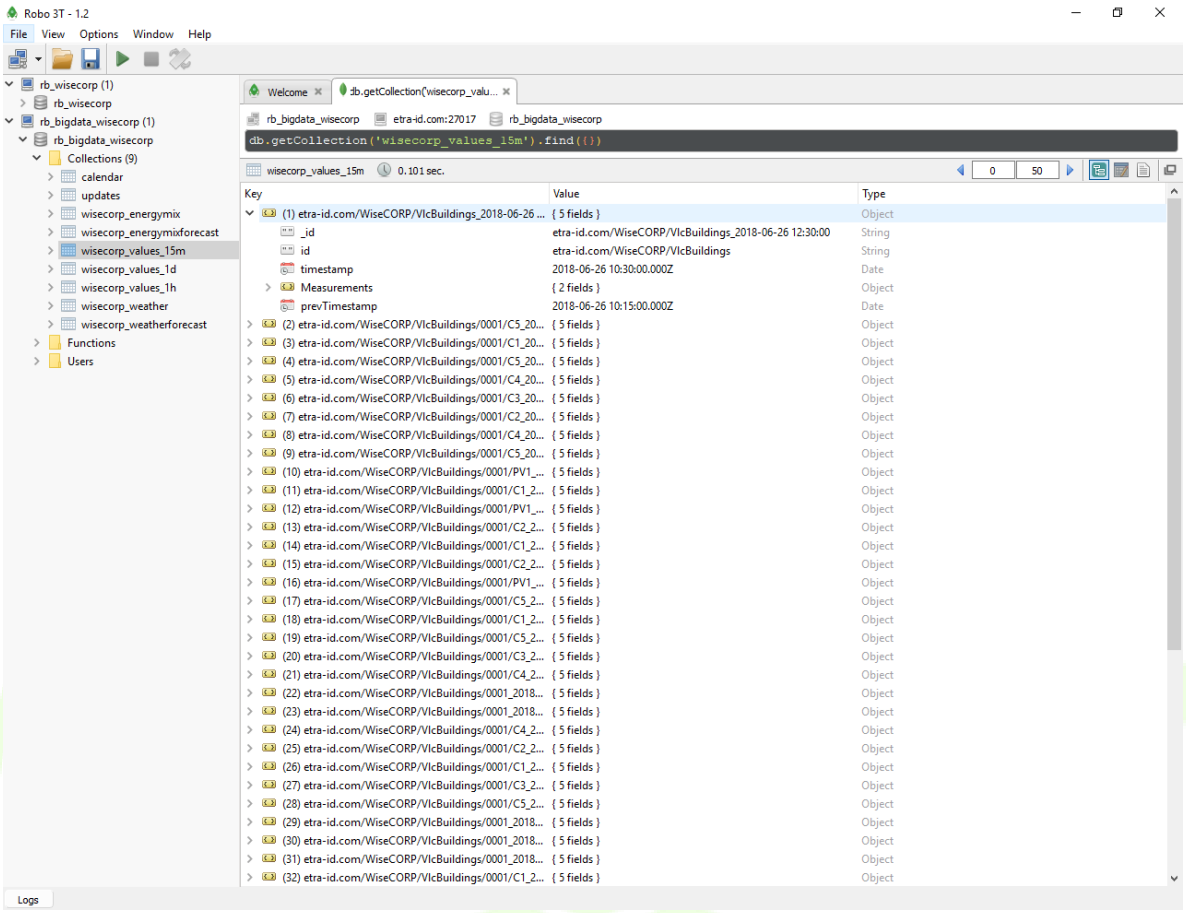



**Figure 30 – WiseCORP long-term database screenshot**

Once the algorithm is ran, the response is serialized and sent back through the corresponding RabbitMQ queue, providing the queried information. The next message is an example of the response received by the WiseCORP, which is printed in the graphic of the forecast view.

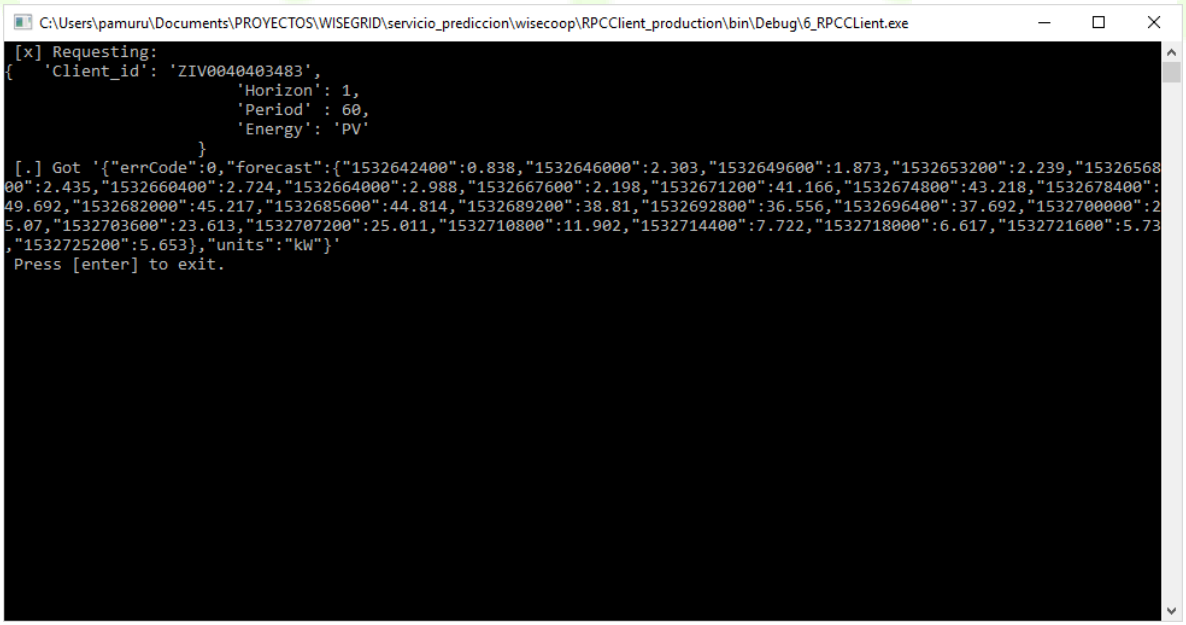



**Figure 31 – Screenshot of forecast response message.**

#### 4.1.2.6 Comfort-based demand flexibility forecast

The comfort-based demand flexibility forecast component is responsible for estimating the potential demand flexibility of the building, while keeping the building occupants under comfortable indoor conditions. It monitors occupant preferences as well as indoor conditions and continuously estimates the available demand flexibility that could be offered to an aggregator – if asked. The message below is a sample response from WiseCORP to such a request of an aggregator – the WiseCOOP tool in particular. WiseCORP specifies per building (shoID) and per time interval (interval, in multitudes of 15 minutes), the demand flexibility potential. The reference time is the time of reception of the demand by WiseCOOP.

**Table 24 – Sample response from WiseCORP to the demand flexibility request from WiseCOOP (as shown in Table 18)**


```
RESPONSE
{
    "shoIdList": [
        {
            "shoId": "SMX1",
            "flexPotentialList": [
                {
                    "flexibility": 7.047032,
                    "interval": 1
                },
                {
                    "flexibility": 6.575823,
                    "interval": 2
                },
                {
                    "flexibility": 14.912035,
                    "interval": 3
                },
                {
                    "flexibility": 15.432471,
                    "interval": 4
                }
            ]
        },
        {
            "shoId": "SMX2",
            "flexPotentialList": [
                {
                    "flexibility": 20.054508,
                    "interval": 1
                },
                {
                    "flexibility": 16.763426,
                    "interval": 2
                },
                {
                    "flexibility": 3.182806,
                    "interval": 3
                },
```

```
                    {
                        "flexibility": 0.9159729,
                        "interval": 4
                    }
                ]
            }
        ]
}
```


#### 4.1.2.7 Energy usage optimizer

The energy usage optimizer is a module developed in Matlab, wrapped in a .NET Framework 4.5 application for facilitating the deployment. This module is triggered once at the end of the day (23:30) in order to calculate the optimum schedule of the controllable assets for the next day. The next 24 hours are divided in equally long slots (24 slots of 1 hour by default), and results are provided as the optimum setpoints for each controllable asset in each slot accordingly to the modelled constraints.

Current model has been developed considering generic characteristics of controllable assets, with the objective of being able to apply it to the majority of cases that will be found in the project. Nevertheless, specific customizations of this module for considering pilot-site related constraints and characteristics is still a possibility likely to be taken in the roadmap of the project.

**Table 25 – WiseCORP – Data inputs of the Energy Usage optimizer**

| Input | Rationale |
|---|---|
| **Demand forecast** | Forecasted demand for the next day is basic data for performing the evaluation of the optimum schedule of the controllable assets. This data accounts for the demand of the non-controllable assets of the building |
| **Production forecast** | Forecasted production for the next day is basic data for performing the evaluation of the optimum schedule of the controllable assets. This data accounts for the production of the non-controllable energy sources of the building |
| **Battery characteristics and status** | In those case where the building accounts with batteries in the list of controllable assets, its characteristics (capacity, max. charge power, max. discharge power, state of charge by the end of the day) need to be taken into account |
| **Controllable demand assets characteristics** | Under this term, information about the controllable assets that impose demand to the building is considered (e.g. industrial machines). For each asset, the following information is considered:<br>• Power of the asset when operating<br>• Time interval when it can operate (e.g. during non-working hours)<br>• Minimum and maximum period of time the asset must operate the following day (e.g. between 2 and 3 hours) |
| **Controllable production assets characteristics** | Under this term, information about the controllable production assets is considered (e.g. CHP). For each asset, the following information is considered:<br>• Max. power of the asset<br>• Associated costs (economic and environmental) bound to the operation of the asset<br>• Time interval when it can operate (e.g. during non-working hours) |
| **Energy price** | If the module is configured to perform economic optimization, the objective function is defined as the total costs of the energy demanded from the grid, whose calculation is based in the energy price |
| **Energy mix** | If the module is configured to perform environmental impact optimization, the objective |

| | |
|---|---|
| | function is defined as the total equivalent $CO_2$ emissions of the energy demanded from the grid and self-produced, whose calculation is based in the energy mix of the grid |
| **Contracted capacity** | Contracted active power capacity of the building that must not be surpassed by the resulting schedule |

**Table 26 – WiseCORP – Data outputs of the Energy Usage optimizer**

| Input | Rationale |
|---|---|
| **Value of objective function** | Total foreseen economic cost or $CO_2$ emissions of the optimized schedule |
| **Battery schedule** | Optimum set of charge/discharge setpoints to be commanded to the battery the following day |
| **Controllable demand schedule** | Optimum schedule for the controllable demand assets for the next 24 hours (ON/OFF periods) |
| **Controllable production schedule** | Optimum schedule for the controllable production assets (setpoints) for the next 24 hours |

### 4.1.3 User interface

In this part of the document, the main sections and functionalities of the WiseCORP GUI are described, including some screenshots of the actual interfaces.

The implementation of WiseCORP was dperformed in the same way than WiseCOOP (Section 3.1.3)

The web application is protected with a user/password credential system to avoid non-authorized personnel to access sensible information. These credentials are requested before accessing the rest of the application.

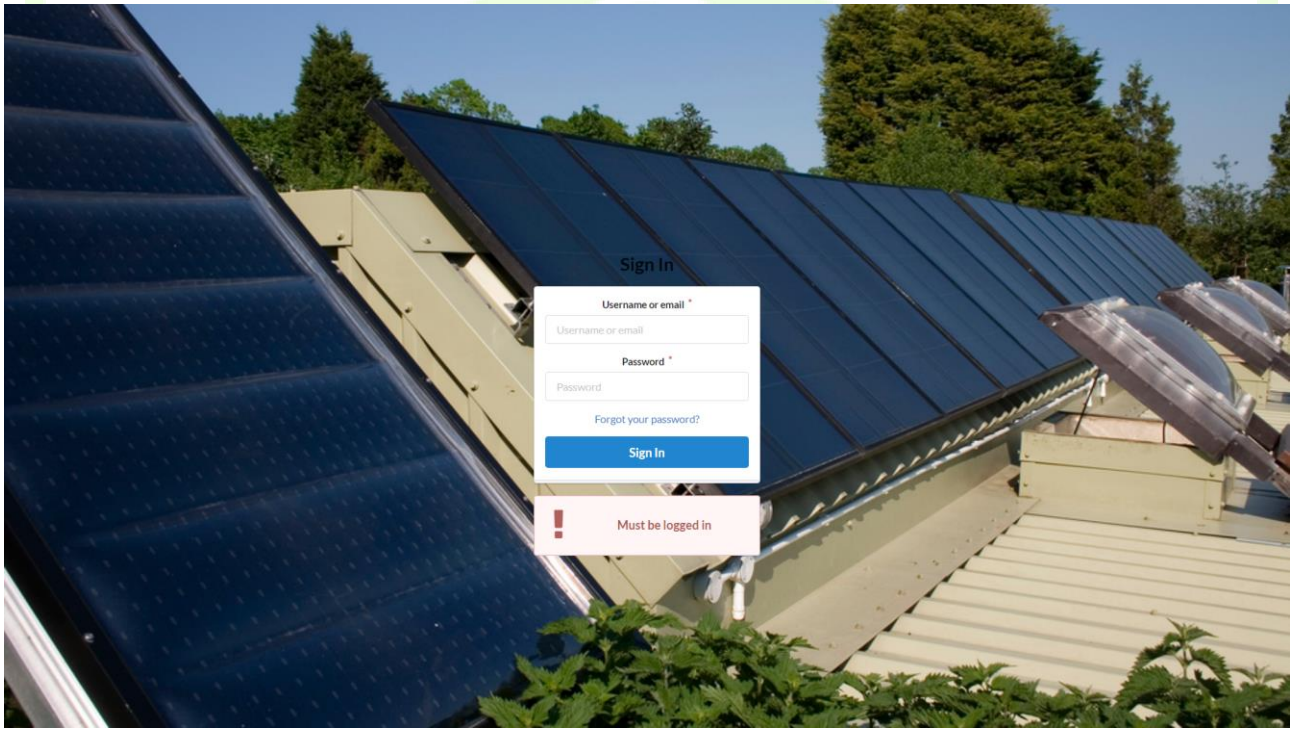



**Figure 32 – WiseCORP Login**

This credential system also permits the definition of different user profiles to grant or deny access to each section of the application independently. This functionality provides an additional level of privacy, as well as flexibility for the system administrator and the operators that make use of the application.

Once the user has been granted access to the application, diverse functionalities will be available as described in the sections below.

#### 4.1.3.1 Dashboard

The dashboard presents data and indicators per building calculated over the las 30 days, namely:

- Name of the building
- Address
- Self-consumption ratio
- $CO_2$ emissions
- Economic cost of the energy
- Energy demand
- Energy production

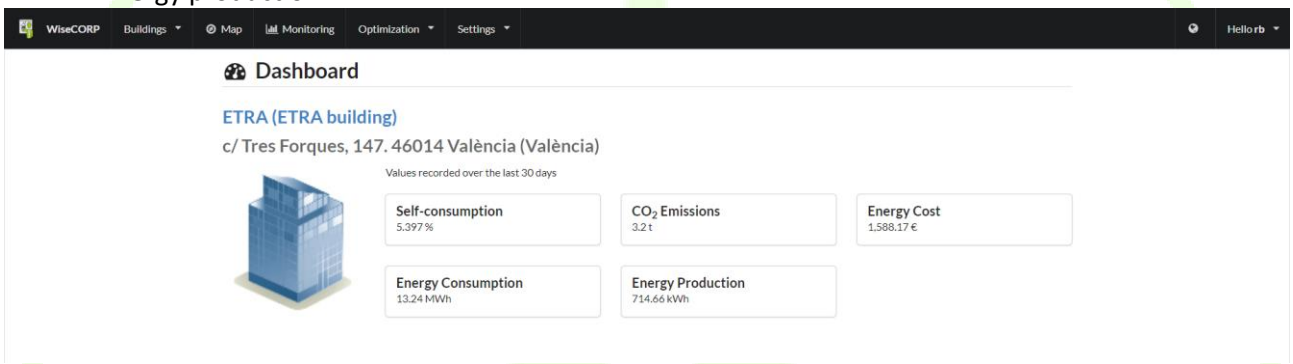



**Figure 33 – WiseCORP UI – Dashboard**

#### 4.1.3.2 Building details

This section provides an insight and detailed indicators for each one of the buildings managed by WiseCORP. Particularly, when a building is selected, for each month the following information is provided:

- Self-consumption ratio
- $CO_2$ emissions
- Economic cost of the energy
- Energy demand
- Energy production
- Chart with daily demand
- Chart with daily production
- Energy consumption share per tariff period

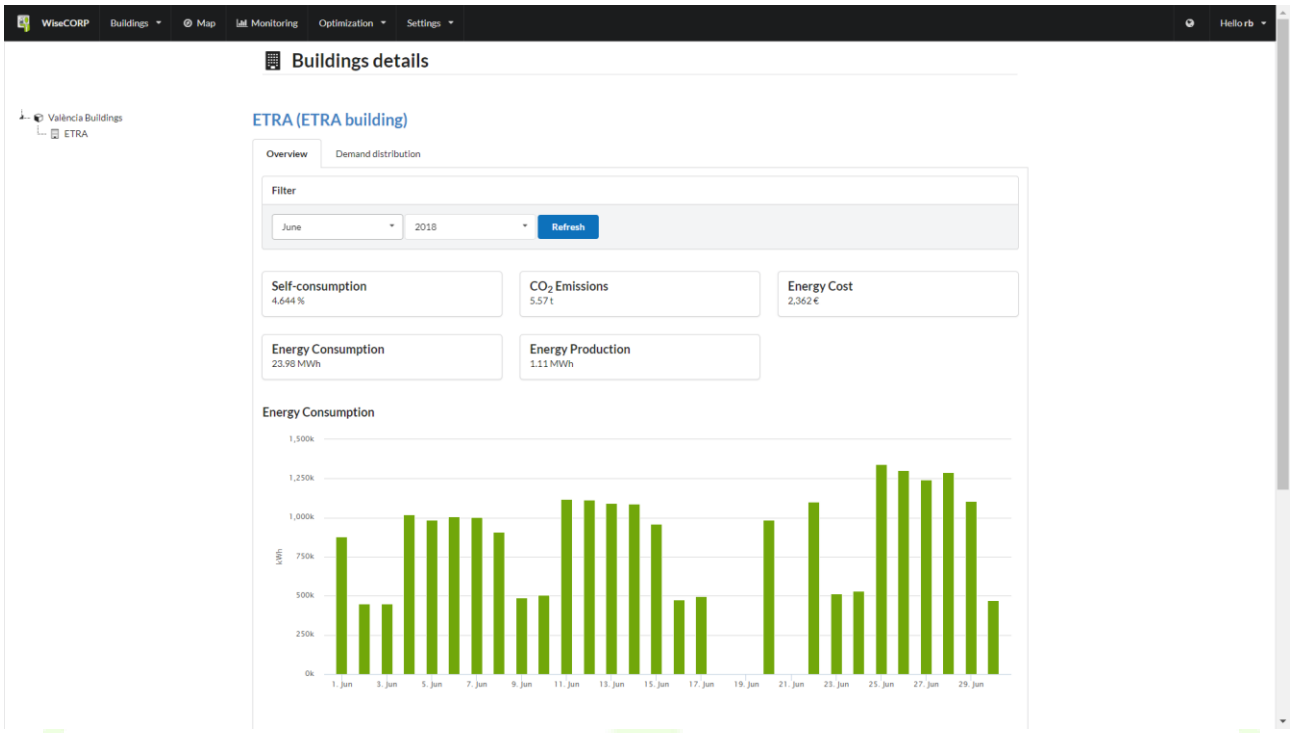



Figure 34 – WiseCORP UI – Building energy usage details (i)

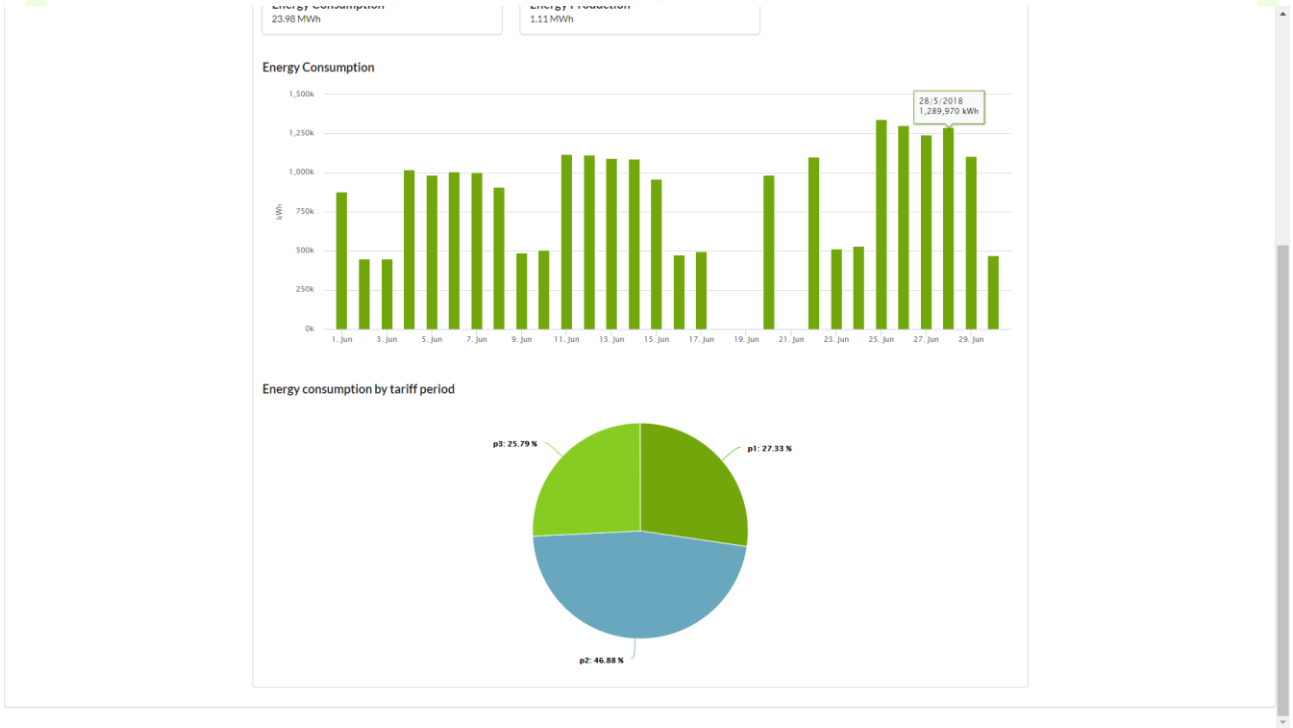



Figure 35 – WiseCORP UI – Building energy usage details (ii)

A second tab of the same section displays a calendar view of the hourly demand and production of the building over the selected week. A colour gradient scale is used to make it possible to easily identify the periods where demand or production is concentrated.

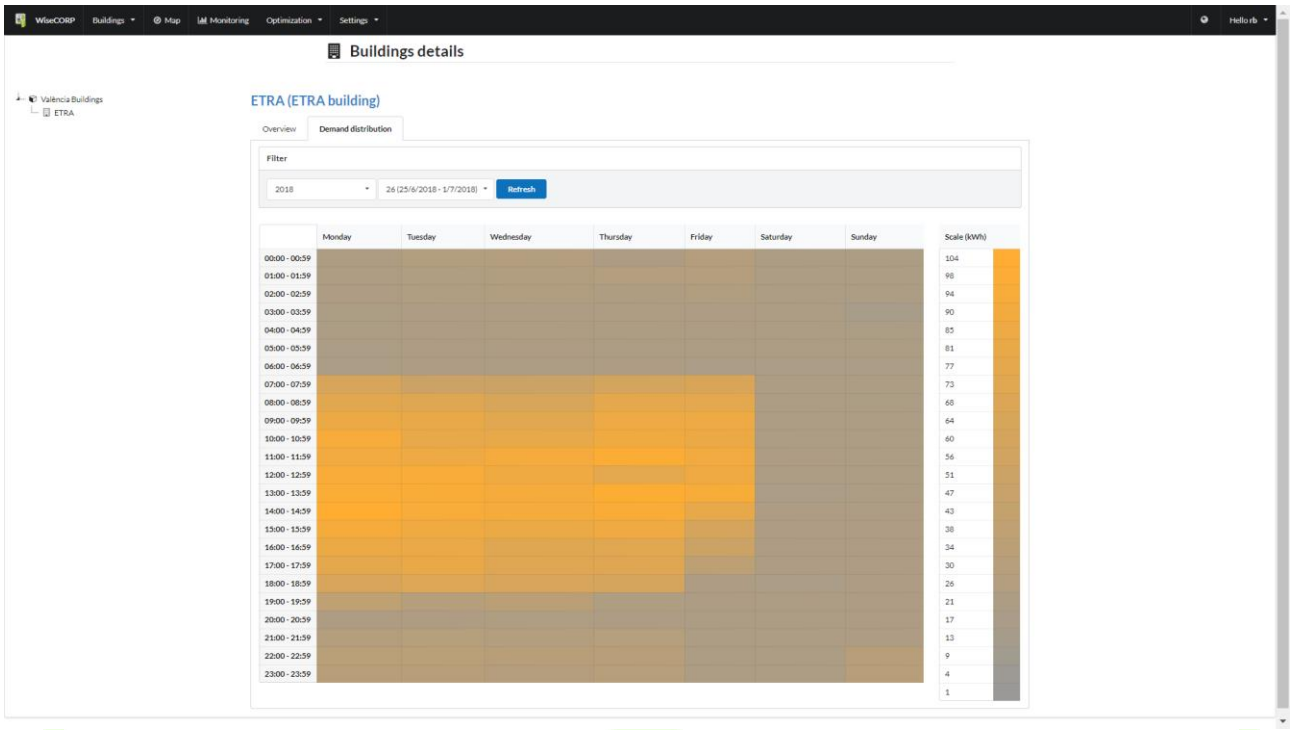



**Figure 36 – WiseCORP UI – Building energy usage details (iii)**

Additionally, a section about automation is also included in the User Interface. This section provides an overview of the schedule planned and executed for each one of the controllable assets of the building (HVAC, batteries, controllable loads, CHP, etc.). The information displayed is actually the output of the Energy Usage Optimizer – the module that calculates the optimum schedule day-ahead – and the demand-response framework – which sets the necessary changes to the original schedule in order to accomplish the requirements of participation in demand-response campaigns.

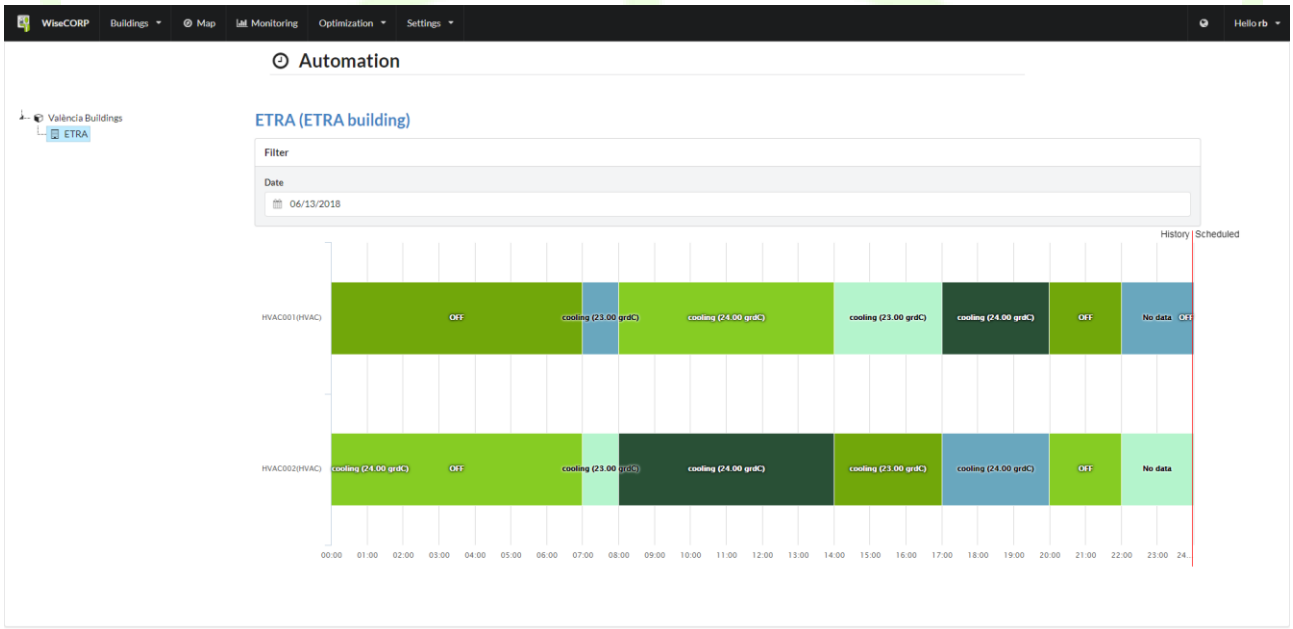



**Figure 37 – WiseCORP UI – Building automation details**

#### 4.1.3.3 Map

Under this section, a geographical representation of the managed buildings is presented. The map displays the location of the different buildings. By clicking in each one of them, the different sensors measuring production and demand in the building are presented, together with a short summary of the last set of values produced by those.

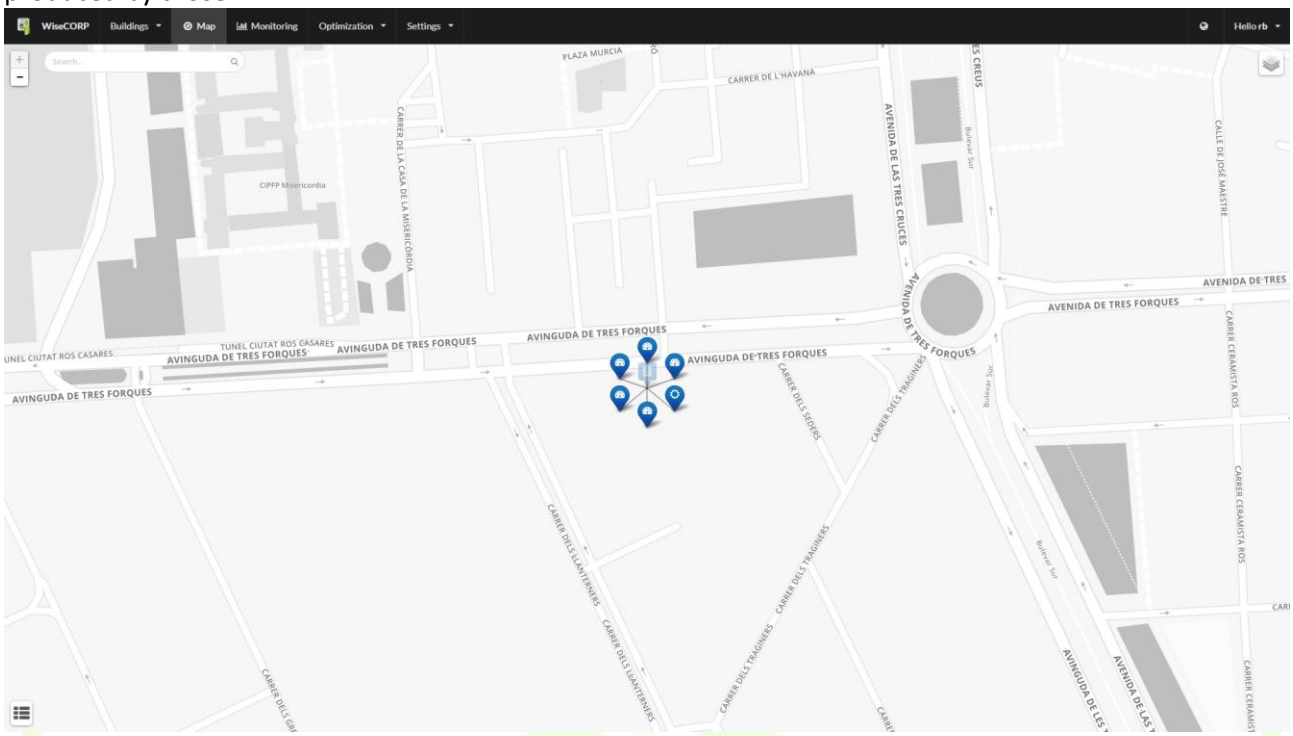



**Figure 38 – WiseCORP UI – Representation of the buildings in a map**

By clicking in each one of the sensors, the site gives access to the representation of all data for the last 24 hours and 30 days.

#### 4.1.3.4 Monitoring

The monitoring section allows the Facility Manager to analyse in detail the different metrics retrieved from the building. Upon selection of a particular asset, data range, set of metrics (active power, energy demand, voltage…) and integration period (15 minutes, 1 hour, 1 day), a chart is displayed showing the required data.

The selection form is versatile enough to allow the Facility Manager to print simultaneously and compare data from different metrics or sensors.

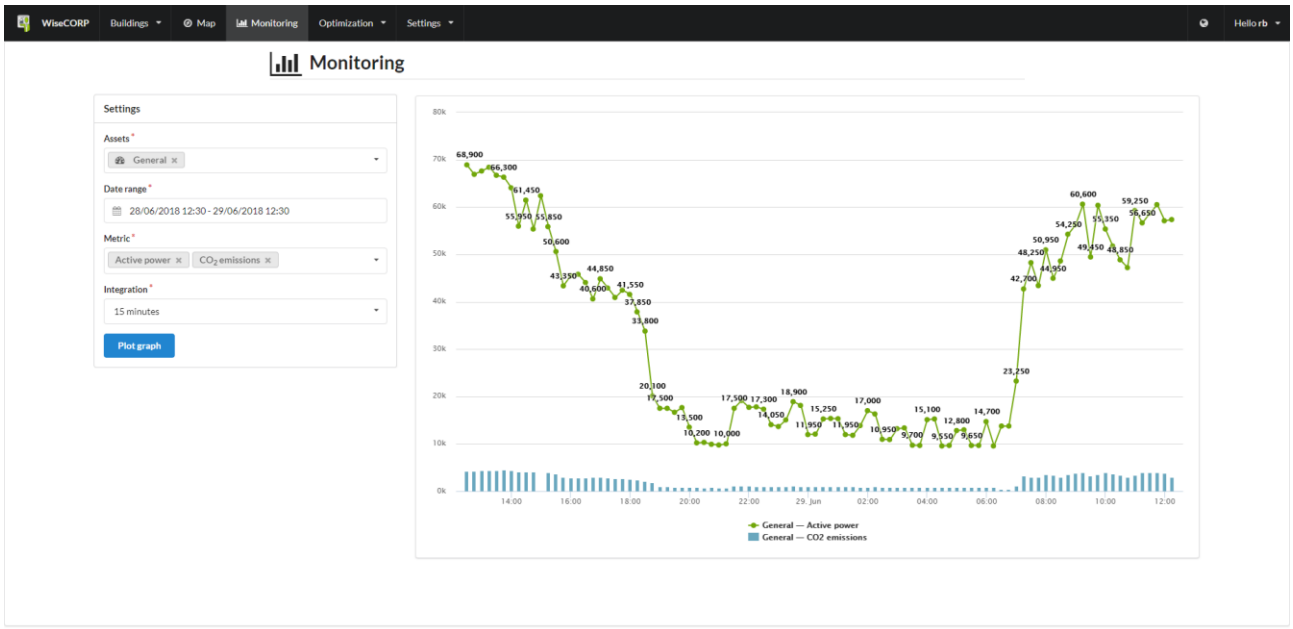



Figure 39 – WiseCORP UI – Monitoring

#### 4.1.3.5 Tariff optimization

The tariff optimization section allows the Facility Manager to use the historical energy demand and production data retrieved from the sensors in the different building in energy cost simulations. Facility Manager can trigger a new simulation by selecting the building or set of sensors whose data wants to analyse, and a complete month.

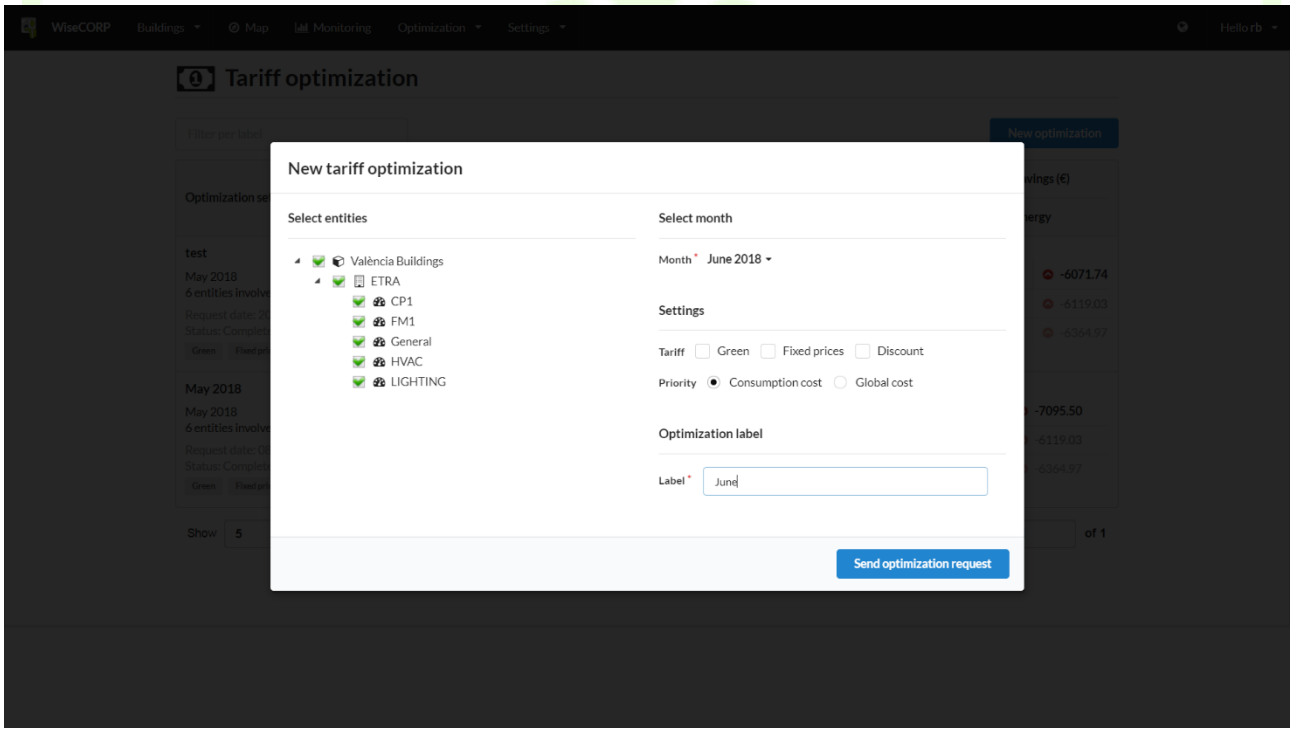



Figure 40 – WiseCORP UI – Tariff optimization, selection of criteria

Once the simulation is triggered, WiseCORP automatically computes the applicable costs that would result for each one of the tariffs defined within the WiseCOOP application (whose definition is shared in the IOP via

the tariff provider module). When the simulation is finishes, a comprehensive comparison of the results is provided to the Facility Manager, making it very easy to identify which tariffs would benefit or would increase the costs associated to the energy demand.

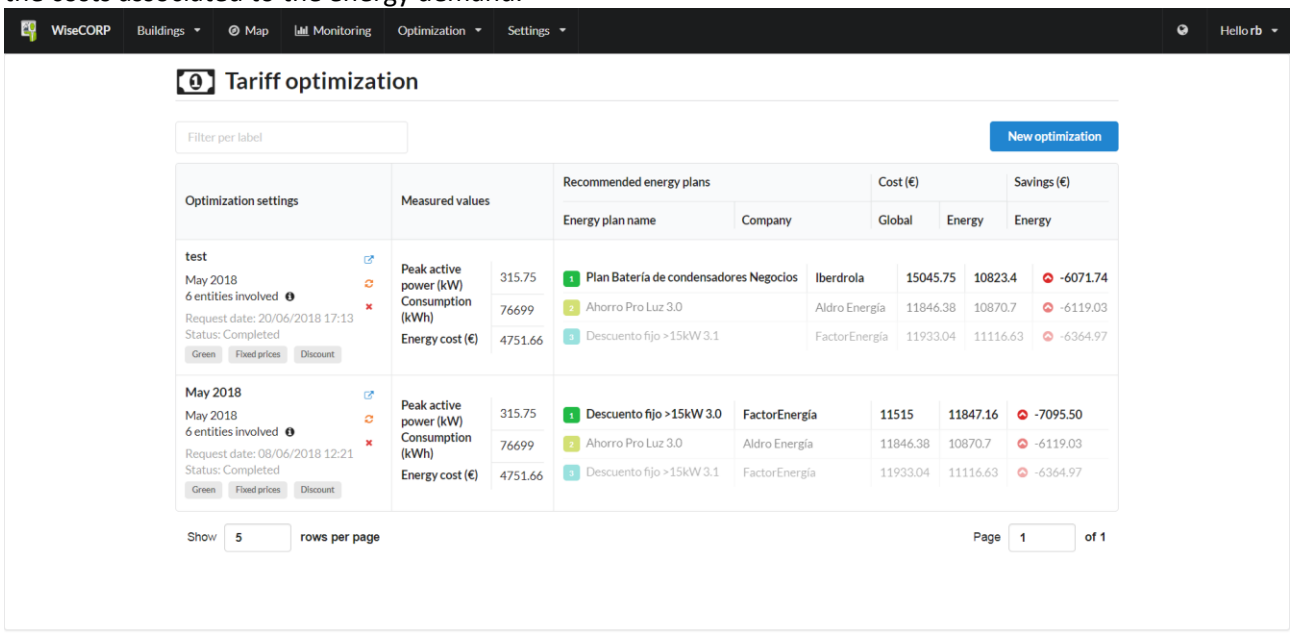



Figure 41 – WiseCORP UI – Tariff optimization, results

In addition, the simulation results include the detail of how the energy bill would look like with each one of the tariffs used in the simulation. These can be used to check the details during the comparison of different tariffs, as relevant information during negotiation of better tariffs with the retailer, or as an assessment of the correctness of the energy bills delivered by the retailer.

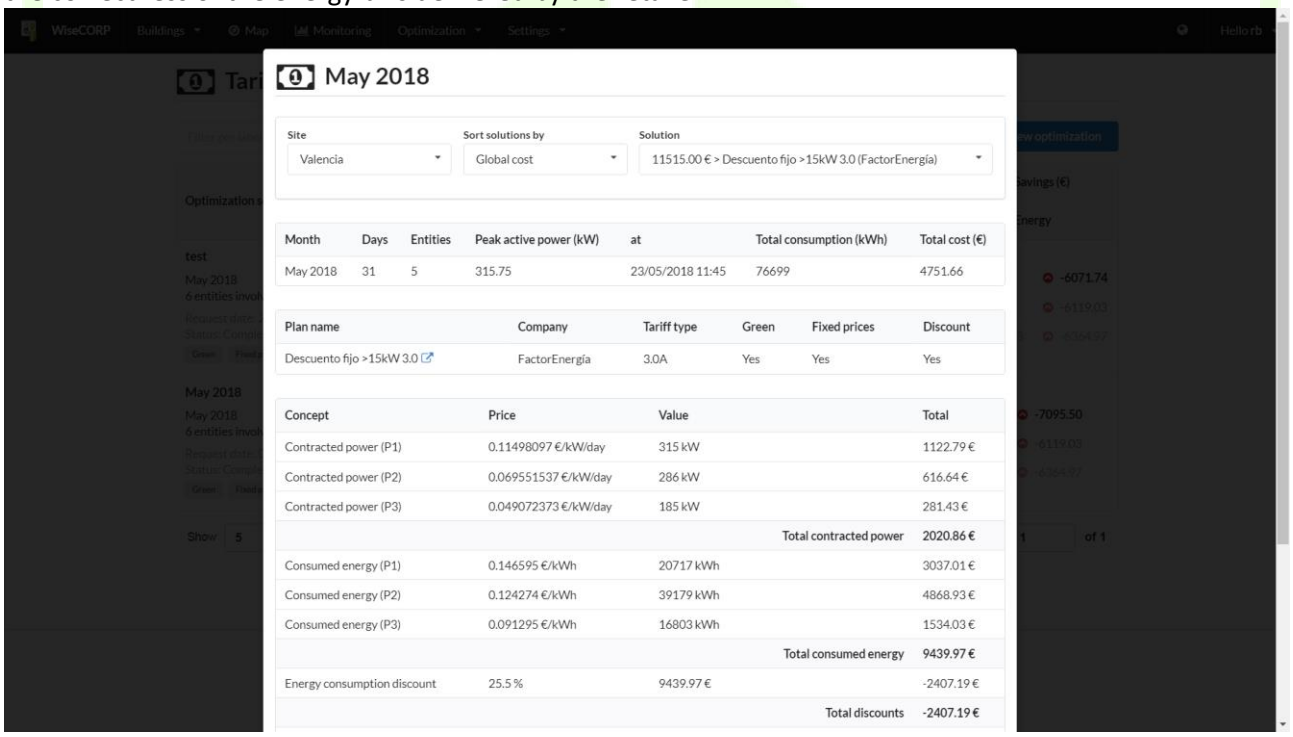



Figure 42 – WiseCORP UI - Tariff optimization, simulated bills

#### 4.1.3.6 Demand response

The Demand response section of the UI provides details on the participation of the facilities in the different demand-response modes enabled by the WiseGRID project. Therefore, upon selection of one building information is presented in two separated tabs.

The first tab is named “Implicit -Dynamic price”, and gives details of the participation of the facility assets in implicit demand-response campaigns, which in the framework of the project have been implemented by means of dynamic tariffs. As explained in detail in D10.2 [6], retailers/aggregators can produce dynamic tariffs for their portfolio of users by taking advantage of the Implicit DR module of the WiseCOOP application. These dynamic tariffs get afterwards communicated to other applications of the WiseGRID ecosystem, enabling those to use the energy price in their specific optimization logic. In the context of WiseCORP, the Energy Usage Optimizer module is in fact capable of using the prices defined in this dynamic tariff to calculate day-ahead the optimum schedule for the different controllable assets, which will minimize the overall energy costs for the facility manager. In consequence, this section of the UI shows the relevant information around this use case: the current day energy price and the optimum schedule for the assets.

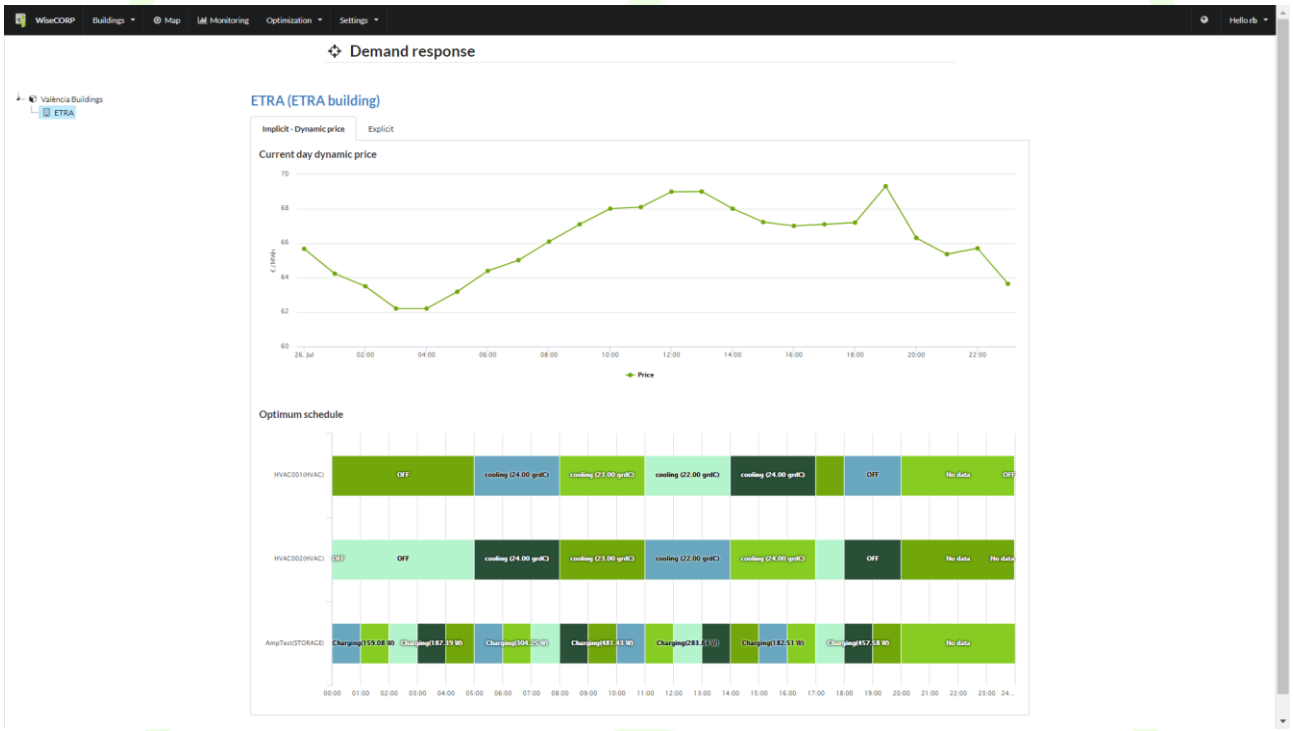



Figure 43 – WiseCORP – Implicit Demand Response

The second tab focuses on the second type of demand-response mechanism enabled within the WiseGRID project, explicit demand response. In this context, the aggregator, using WiseCOOP, is authorized to explicitly request a certain amount of flexibility from the controllable assets of the facility. The Demand-Response framework is in position of calculating how the schedule of the assets shall be changed in order to meet with the new constraints imposed by the participation in the explicit demand-response campaign. This section of the UI therefore focuses on displaying the relevant changes produced in the schedule of the assets as a result of the execution of those campaigns.

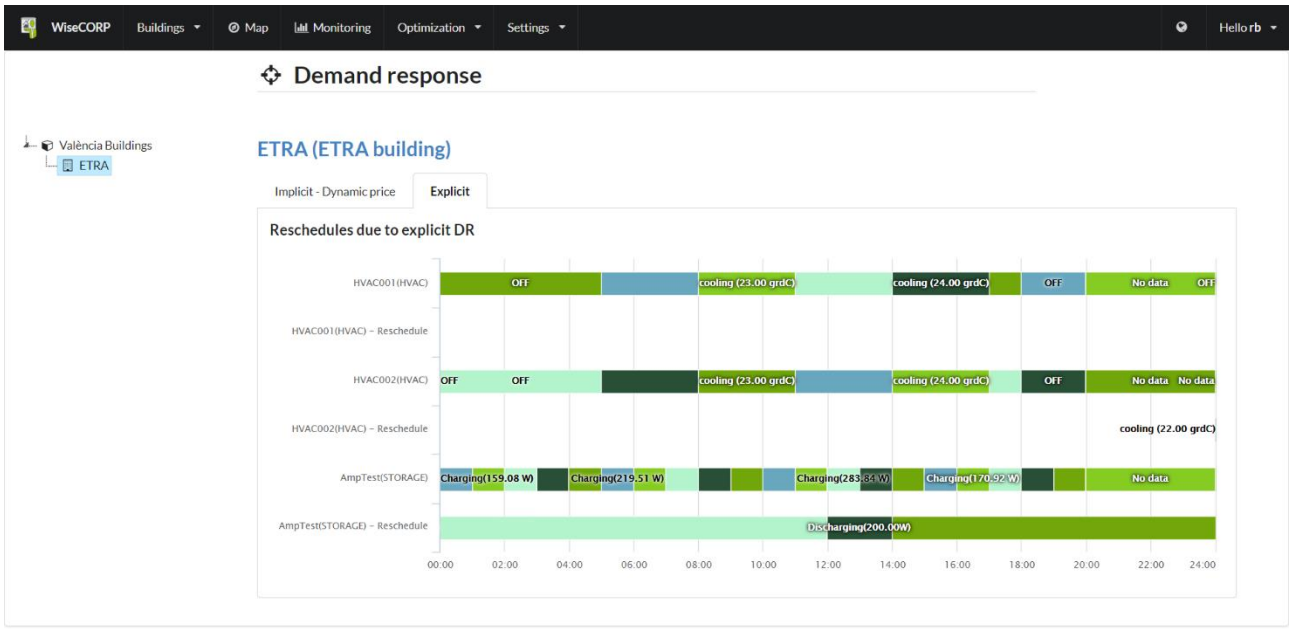


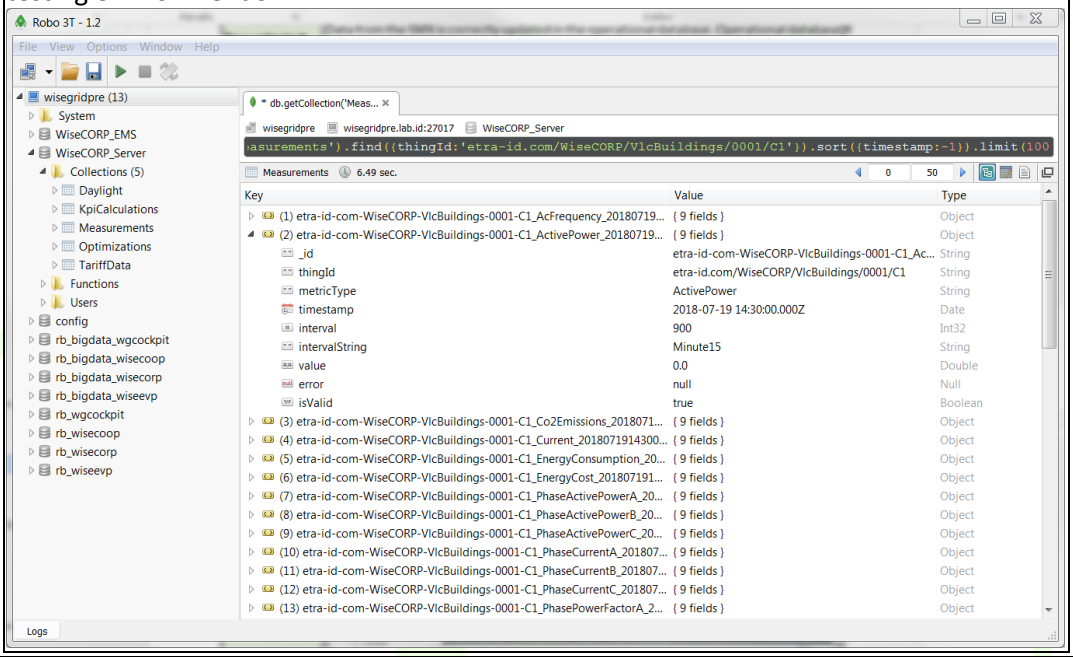

**Figure 44 – WiseCORP – Explicit Demand Response**

## 4.2 LAB-TESTING RESULTS

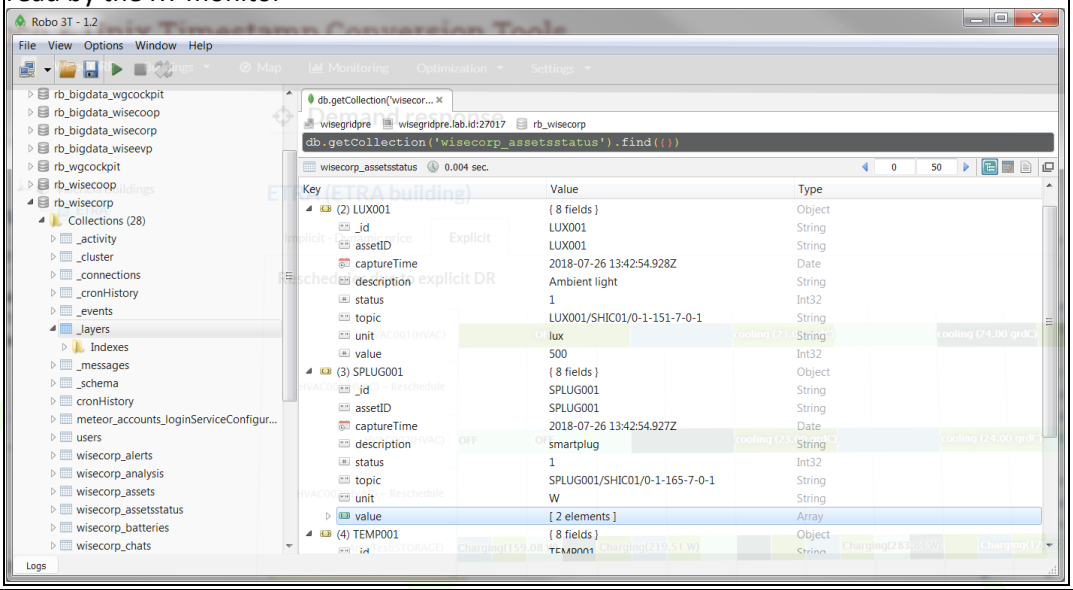
This section contains a set of templates with the definition, objectives, steps and results of all tests executed during this period on the different modules of the tool.

### 4.2.1 RT monitor tests

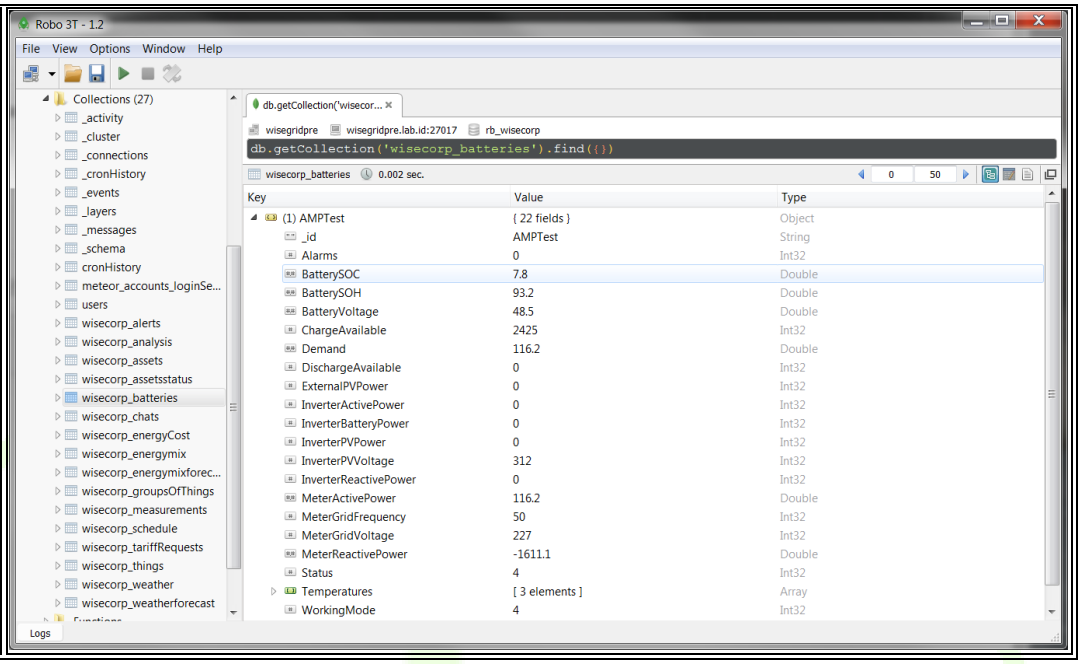
| | | | |
|---|---|---|---|
| **Name** | RTM001. Read smart meter data from IOP | | |
| **Module under test** | RT Monitor | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running | | |
| **Features to be tested** | Data from SMX is properly collected in the operational database of WiseCORP | | |
| **Features not to be tested** | | | |
| **Preparation** | Configure one SMX to send data to the lab-testing IOP environment | | |
| **Dependencies** | | | |
| **Steps** | The execution of this test must happen automatically upon publication of data in the IOP | | |
| **Pass criteria** | Data from the SMX is correctly updated in the operational database. Operational database keeps a register of the last values sent by the SMX. | | |
| **Suspension criteria** | | | |

| | |
|---|---|
| **Results** | Test successful.<br>The following screenshot shows how the operational DB is populated with data from lab-testing environment SMX<br>  |

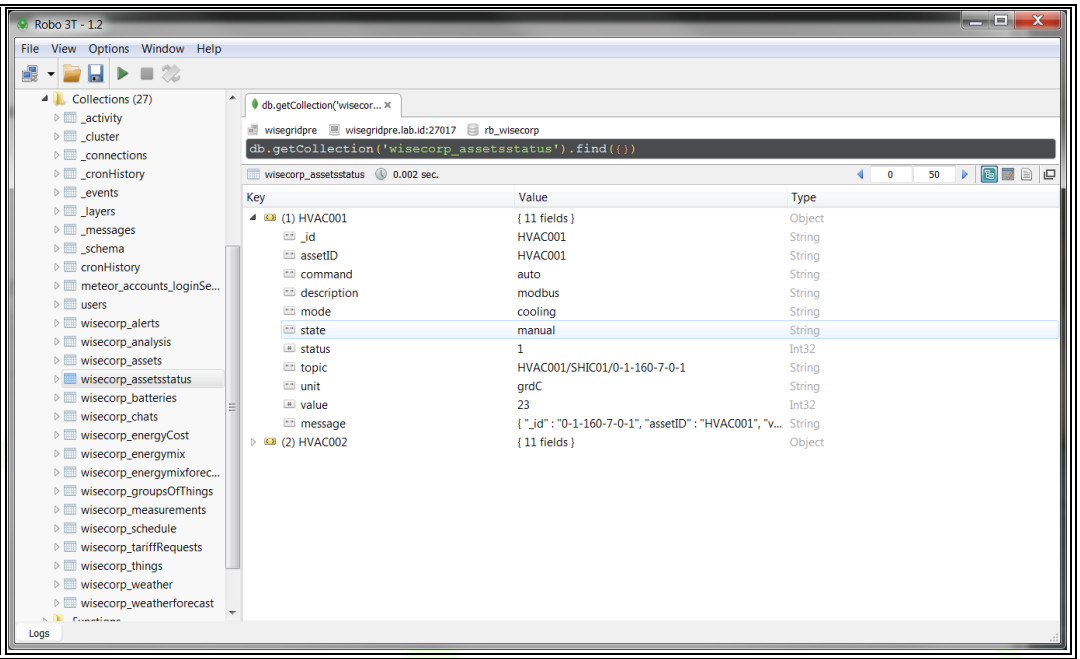
| | | | |
|---|---|---|---|
| **Name** | RTM002. Read sensor data from IOP | | |
| **Module under test** | RT Monitor | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | Sensor wrapper simulator<br>IOP platform up and running | | |
| **Features to be tested** | Data from sensor Wrapper is properly collected in the operational database of WiseCORP | | |
| **Features not to be tested** | | | |
| **Preparation** | Configure one sensor wrapper simulator to send data to the lab-testing IOP environment | | |
| **Dependencies** | | | |
| **Steps** | The execution of this test must happen automatically upon publication of data in the IOP | | |
| **Pass criteria** | Data from the sensor is correctly updated in the operational database. Operational database keeps a register of the last values sent by the sensor. | | |
| **Suspension criteria** | | | |
| **Results** | Test successful<br>Sensor data (temperature, lighting and smart plug) sent by the simulators is successfully | | |

| | |
|---|---|
| | read by the RT monitor<br>  |

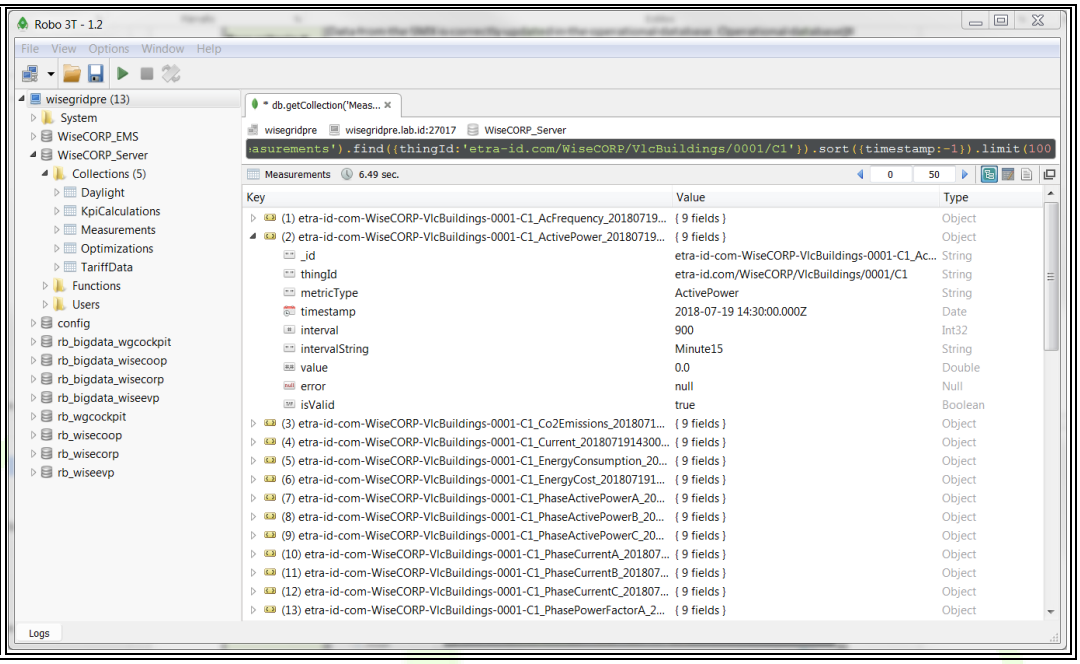
| | | | |
|---|---|---|---|
| **Name** | RTM003. Read battery data from IOP | | |
| **Module under test** | RT Monitor | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | Battery running and sending information to IOP<br>IOP platform up and running | | |
| **Features to be tested** | Data from battery is properly collected in the operational database of WiseCORP | | |
| **Features not to be tested** | | | |
| **Preparation** | Configure one battery to send data to the lab-testing IOP environment | | |
| **Dependencies** | | | |
| **Steps** | The execution of this test must happen automatically upon publication of data in the IOP | | |
| **Pass criteria** | Data from the battery is correctly updated in the operational database. Operational database keeps a register of the last values sent by the battery. | | |
| **Suspension criteria** | | | |
| **Results** | Test successful.<br>The following screenshot shows how the operational DB is populated with data from lab-testing environment battery from VARTA | | |

| | |
|---|---|
| |  |

| | | | |
|---|---|---|---|
| **Name** | RTM004. Read HVAC data from IOP | | |
| **Module under test** | RT Monitor | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | HVAC Wrapper simulator<br>IOP platform up and running | | |
| **Features to be tested** | Data from HVAC is properly collected in the operational database of WiseCORP | | |
| **Features not to be tested** | | | |
| **Preparation** | Configure one HVAC simulator to send data to the lab-testing IOP environment | | |
| **Dependencies** | | | |
| **Steps** | The execution of this test must happen automatically upon publication of data in the IOP | | |
| **Pass criteria** | Data from the HVAC is correctly updated in the operational database. Operational database keeps a register of the last values sent by the SMX. | | |
| **Suspension criteria** | | | |
| **Results** | Test successful.<br>The following screenshot shows how the operational DB is populated with data from lab-testing environment wrapper simulators | | |

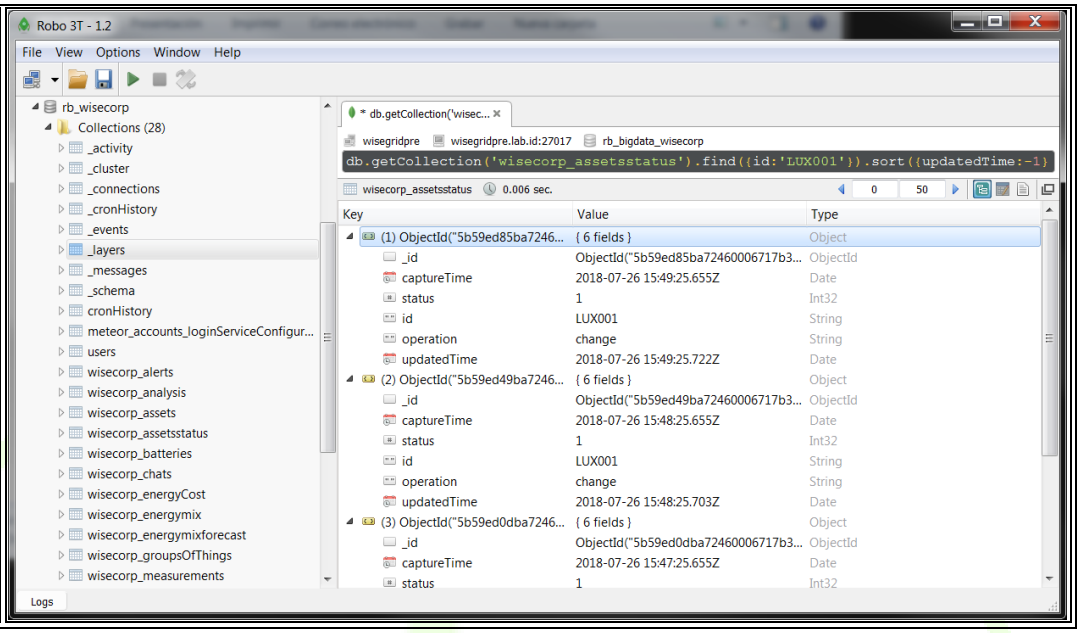
| | |
|---|---|
| |  |

| | | | |
|---|---|---|---|
| **Name** | RTM005. Store smart meter data to Long-term DB | | |
| **Module under test** | RT Monitor | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running | | |
| **Features to be tested** | Data from SMX is properly collected in the long-term database of WiseCORP (big data) | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |
| **Dependencies** | RTM001. Read smart meter data from IOP | | |
| **Steps** | The execution of this test must happen automatically upon publication of data in the IOP | | |
| **Pass criteria** | Data from the SMX is correctly appended to the historic registry held in the long-term database | | |
| **Suspension criteria** | | | |
| **Results** | Test successful.<br>The following screenshot shows how the operational DB is populated with data from lab-testing environment SMX | | |

| | |
|---|---|
| |  |

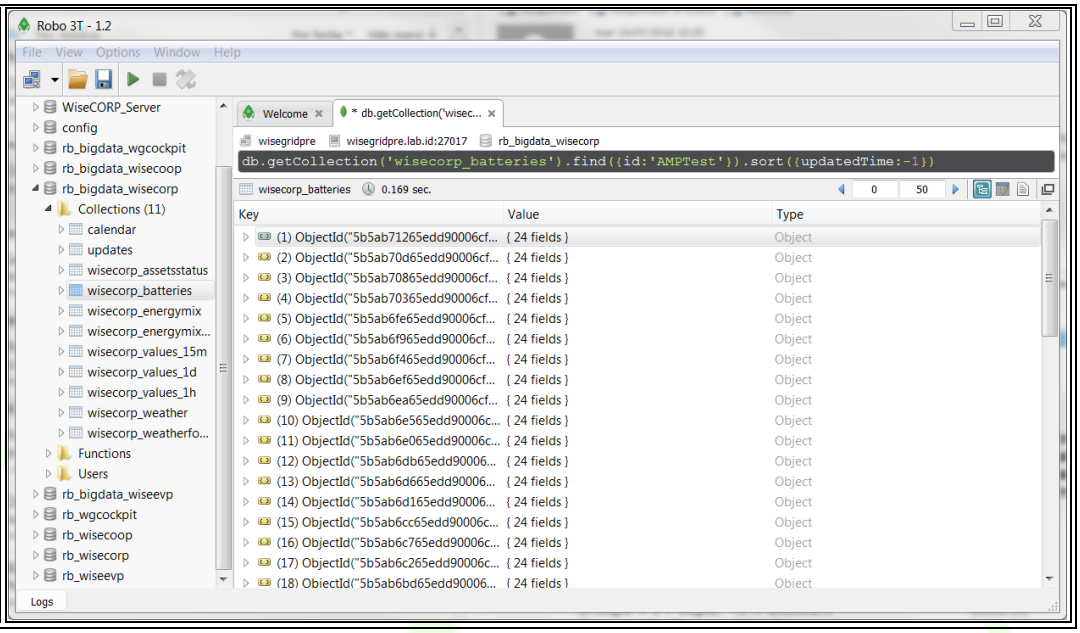


| | | | |
|---|---|---|---|
| **Name** | RTM006. Store sensor data to Long-term DB | | |
| **Module under test** | RT Monitor | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running | | |
| **Features to be tested** | Data from sensor is properly collected in the long-term database of WiseCORP (big data) | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |
| **Dependencies** | RTM002. Read sensor data from IOP | | |
| **Steps** | The execution of this test must happen automatically upon publication of data in the IOP | | |
| **Pass criteria** | Data from the sensor is correctly appended to the historic registry held in the long-term database | | |
| **Suspension criteria** | | | |
| **Results** | Test successful<br>History of changes of sensor data gets successfully stored to the long-term DB | | |

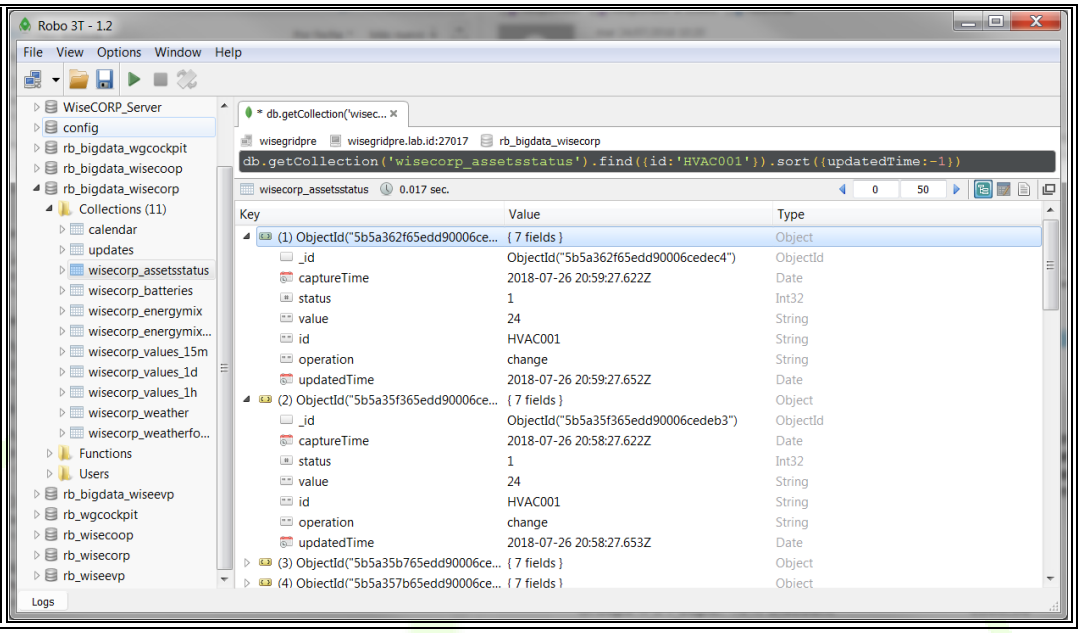
| | |
|---|---|
| |  |



| | | | |
|---|---|---|---|
| **Name** | RTM007. Store battery data to long-term DB | | |
| **Module under test** | RT Monitor | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | Battery running and sending information to IOP<br>IOP platform up and running | | |
| **Features to be tested** | Data from battery is properly collected in the long-term database of WiseCORP (big data) | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |
| **Dependencies** | RTM004. Read battery data from IOP | | |
| **Steps** | The execution of this test must happen automatically upon publication of data in the IOP | | |
| **Pass criteria** | Data from the battery is correctly appended to the historic registry held in the long-term database | | |
| **Suspension criteria** | | | |
| **Results** | Test successful<br>The long-term database successfully stores all information sent by the batteries monitored by WiseCORP | | |

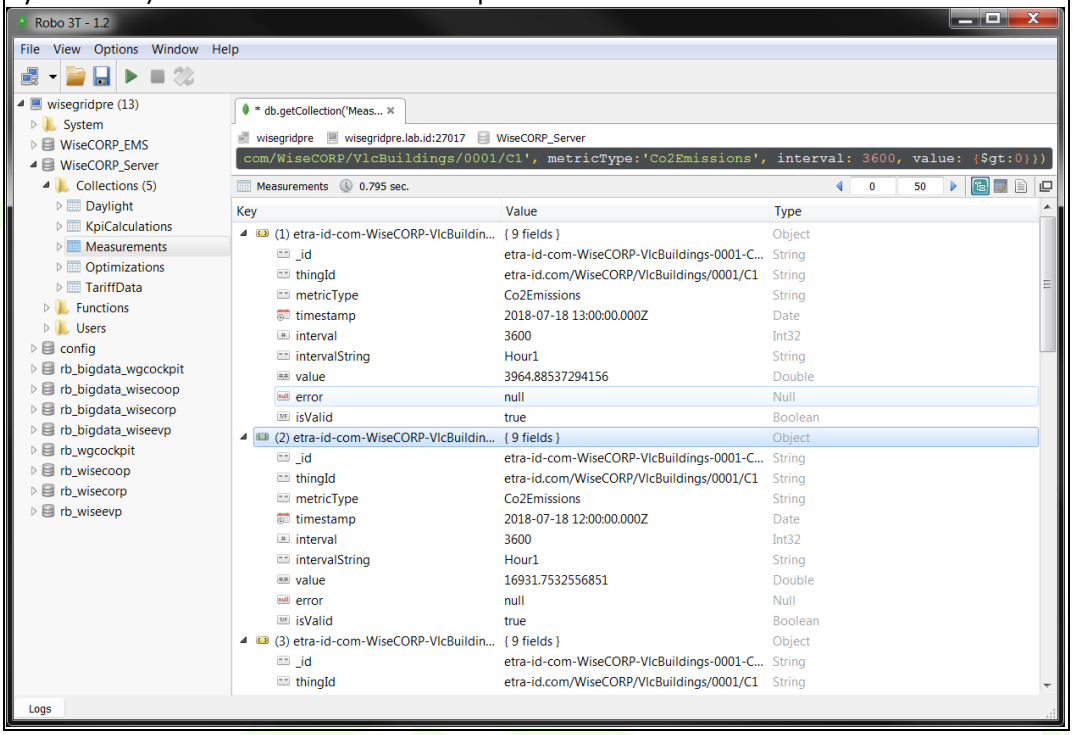
| | |
|---|---|
| |  |

| | | | |
|---|---|---|---|
| **Name** | RTM008. Store HVAC data to Long-term DB | | |
| **Module under test** | RT Monitor | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | HVAC Wrapper simulator<br>IOP platform up and running | | |
| **Features to be tested** | Data from HVAC is properly collected in the long-term database of WiseCORP (big data) | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |
| **Dependencies** | RTM005. Read HVAC data from IOP | | |
| **Steps** | The execution of this test must happen automatically upon publication of data in the IOP | | |
| **Pass criteria** | Data from the HVAC is correctly appended to the historic registry held in the long-term database | | |
| **Suspension criteria** | | | |
| **Results** | Test successful<br>The long-term database successfully stores all information sent by the HVACs monitored by WiseCORP | | |

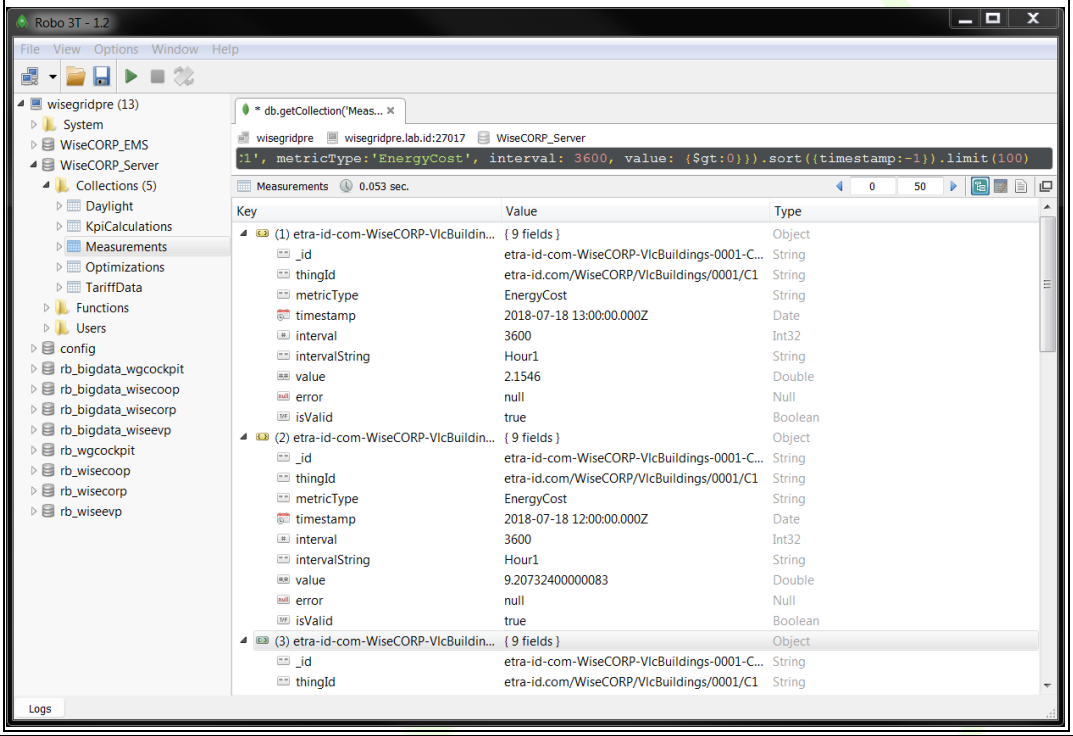
| | |
|---|---|
| |  |

### 4.2.2 KPI engine tests

| | | | |
|---|---|---|---|
| **Name** | KPI001. Associated $CO_2$ emissions | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running<br>RT monitor storing smart meter readings in long-term database<br>Spark server up and running<br>Energy mix provider module up and running (ENTSOE provider) | | |
| **Features to be tested** | This module must crosscheck the energy mix information with the individual energy demand readouts in order to compute the equivalent $CO_2$ emissions | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |
| **Dependencies** | RTM001. Read smart meter data from IOP<br>RTM007. Store smart meter data to Long-term DB | | |
| **Steps** | 1. Execute Spark job for $CO_2$ emissions calculation | | |
| **Pass criteria** | - The long-term database contains 3 collections with the quarterly, hourly and daily aggregations<br>- Each collection contains documents that contain the equivalent $CO_2$ emissions of | | |

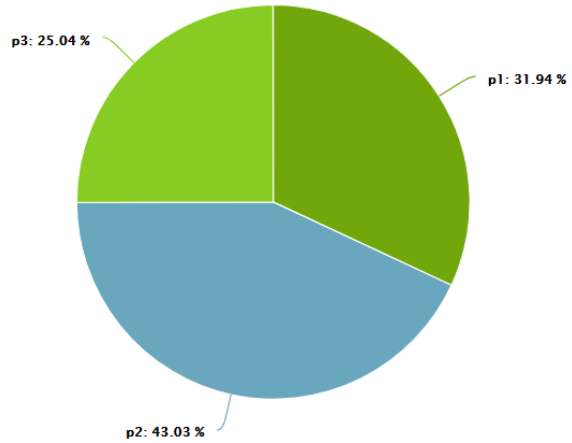
| | |
|---|---|
| | the demand they represent |
| **Suspension criteria** | |
| **Results** | Test successful<br>WiseCORP crosschecks the measured demand with the national energy mix (as reported by ENTSO-E) in order to calculate the equivalent CO2 emissions of the demand<br>  |

| | | | |
|---|---|---|---|
| **Name** | KPI002. Associated economic costs | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running<br>RT monitor storing smart meter readings in long-term database<br>Spark server up and running<br>Tariff provider module up and running | | |
| **Features to be tested** | This module must cross-check the energy price for the contracted tariff with the individual energy demand readouts in order to compute the associated economic costs | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |

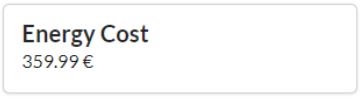
| | |
|---|---|
| **Dependencies** | RTM001. Read smart meter data from IOP<br>RTM007. Store smart meter data to Long-term DB |
| **Steps** | 1. Execute Spark job for economic costs calculation |
| **Pass criteria** | - The long-term database contains 3 collections with the quarterly, hourly and daily aggregations<br>- Each collection contains documents that contain the associated economic costs of the demand they represent |
| **Suspension criteria** | |
| **Results** | Test successful<br>WiseCORP cross-checks the measured demand with the applicable tariff in order to calculate the costs associated to the demand<br>  |



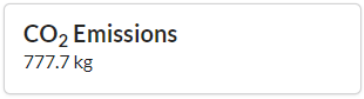
| | | | |
|---|---|---|---|
| **Name** | KPI003. Distribution of demand per tariff period | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running<br>RT monitor storing smart meter readings in long-term database<br>Spark server up and running<br>Tariff provider module up and running | | |

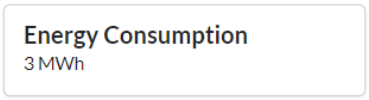
| | |
|---|---|
| **Features to be tested** | This module must crosscheck the energy demand of the building with the periods of the contracted tariff, giving an overview of the distribution of the demand over a period of time (e.g. monthly) |
| **Features not to be tested** | |
| **Preparation** | |
| **Dependencies** | RTM001. Read smart meter data from IOP<br>RTM007. Store smart meter data to Long-term DB |
| **Steps** | 1. Open the WiseCORP UI<br>2. Access the building indicators site |
| **Pass criteria** | The distribution of demand per tariff period for the selected month is presented |
| **Suspension criteria** | |
| **Results** | Test successful<br>The UI displays the required information, calculated over the data of a complete month and a building<br>Energy consumption by tariff period<br>  |

| | | | |
|---|---|---|---|
| **Name** | KPI004. Calculation of indicators per building. Monthly economic costs | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running<br>RT monitor storing smart meter readings in long-term database | | |
| **Features to be tested** | This module analyses historic information available in the long-term | | |

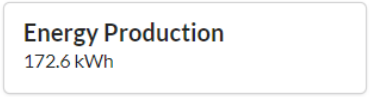
| | |
|---|---|
| | database to provide insights on the economic costs faced by the measured facilities |
| **Features not to be tested** | |
| **Preparation** | |
| **Dependencies** | KPI002. Associated economic costs |
| **Steps** | 1. Open the WiseCORP UI<br>2. Access the building indicators site |
| **Pass criteria** | The UI represents the total monthly economic cost per building |
| **Suspension criteria** | |
| **Results** | Test successful<br>The UI displays the required information, calculated over the data of a complete month and a building<br>  |

| | | | |
|---|---|---|---|
| **Name** | KPI005. Calculation of indicators per building. $CO_2$ emissions | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running<br>RT monitor storing smart meter readings in long-term database | | |
| **Features to be tested** | This module analyses historic information available in the long-term database to provide insights on the equivalent $CO_2$ emissions produced by the measured facilities | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |
| **Dependencies** | KPI001. Associated $CO_2$ emissions | | |
| **Steps** | 1. Open the WiseCORP UI<br>2. Access the building indicators site | | |
| **Pass criteria** | The UI represents the total monthly equivalent $CO_2$ emissions per building | | |
| **Suspension criteria** | | | |
| **Results** | Test successful<br>The UI displays the required information, calculated over the data of a | | |

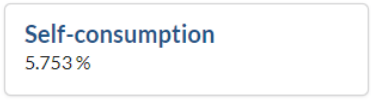
| | |
|---|---|
| | complete month and a building<br>  |

| | | | |
|---|---|---|---|
| **Name** | KPI006. Calculation of indicators per building. Total demand | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running<br>RT monitor storing smart meter readings in long-term database | | |
| **Features to be tested** | This module analyses historic information available in the long-term database to provide insights on the total energy demand of the measured facilities | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |
| **Dependencies** | RTM001. Read smart meter data from IOP<br>RTM007. Store smart meter data to Long-term DB | | |
| **Steps** | 1. Open the WiseCORP UI<br>2. Access the building indicators site | | |
| **Pass criteria** | The UI represents the total monthly demand per building | | |
| **Suspension criteria** | | | |
| **Results** | Test successful<br>The UI displays the required information, calculated over the data of a complete month and a building<br>  | | |

| | | | |
|---|---|---|---|
| **Name** | KPI007. Calculation of indicators per building. Total production | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |

| | |
|---|---|
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running<br>RT monitor storing smart meter readings in long-term database |
| **Features to be tested** | This module analyses historic information available in the long-term database to provide insights on the total energy production of the measured facilities |
| **Features not to be tested** | |
| **Preparation** | |
| **Dependencies** | RTM001. Read smart meter data from IOP<br>RTM007. Store smart meter data to Long-term DB |
| **Steps** | 1. Open the WiseCORP UI<br>2. Access the building indicators site |
| **Pass criteria** | The UI represents the total monthly production per building |
| **Suspension criteria** | |
| **Results** | Test successful<br>The UI displays the required information, calculated over the data of a complete month and a building<br>  |

| | | | |
|---|---|---|---|
| **Name** | KPI008. Calculation of indicators per building. Self-consumption ratio | | |
| **Module under test** | KPI engine | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | SMX running and sending information to IOP<br>IOP platform up and running<br>RT monitor storing smart meter readings in long-term database | | |
| **Features to be tested** | This module analyses historic information available in the long-term database to provide insights on the total self-consumption and the energy production surplus of the measured facilities | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |
| **Dependencies** | RTM001. Read smart meter data from IOP<br>RTM007. Store smart meter data to Long-term DB | | |
| **Steps** | 1. Open the WiseCORP UI<br>2. Access the building indicators site | | |
| **Pass criteria** | The UI represents the self-consumption ration for the selected month | | |

| | |
|---|---|
| | per building<br>The UI represents the total production surplus for the selected month per building |
| **Suspension criteria** | |
| **Results** | Test successful<br>The UI displays the required information, calculated over the data of a complete month and a building<br>  |

### 4.2.3 Forecast modules tests

| | | | |
|---|---|---|---|
| **Name** | FOR001. Demand/production forecasting training | | |
| **Module under test** | WiseCORP forecast module | **Resp.** | ITE |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | WiseCORP forecast module up and running<br>Historical data available in long-term DB | | |
| **Features to be tested** | WiseCORP forecast module is trained | | |
| **Features not to be tested** | | | |
| **Preparation** | | | |
| **Dependencies** | RTM002. Read sensor data from IOP | | |
| **Steps** | Perform WiseCORP forecast training | | |
| **Pass criteria** | Training MAPE below pre-defined threshold | | |
| **Suspension criteria** | | | |
| **Results** | Test successful<br>WiseCORP forecast model trained | | |

| | | | |
|---|---|---|---|
| **Name** | FOR002. Demand/Production forecasting | | |
| **Module under test** | WiseCORP forecast module | **Resp.** | ITE |

| | |
|---|---|
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector |
| **Test environment** | WiseCORP forecast module up and running<br>Historical data available in long-term DB |
| **Features to be tested** | WiseCORP forecast module performs demand/production forecasting training |
| **Features not to be tested** | |
| **Preparation** | Train WiseCORP demand/production forecast module |
| **Dependencies** | FOR001. Demand/production forecasting training<br>RTM002. Read sensor data from IOP |
| **Steps** | WiseCORP forecast module |
| **Pass criteria** | Prediction MAPE below pre-defined threshold |
| **Suspension criteria** | |
| **Results** | Test successful<br>24 hours hourly load and production prediction |

| | | | |
|---|---|---|---|
| **Name** | FOR003. Request message parsing test of WiseCORP forecast module | | |
| **Module under test** | WiseCORP forecast module | **Resp.** | ITE |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | Development RabbitMQ environment<br>WiseCORP forecast module up and running<br>Historical data available in long-term DB | | |
| **Features to be tested** | Performance of WiseCORP forecast module, at parsing forecast queries. | | |
| **Features not to be tested** | | | |
| **Preparation** | Enable RabbitMQ queues, and run WiseCORP forecast module | | |
| **Dependencies** | | | |
| **Steps** | 1. Receipt of request<br>2. Request parsing<br>3. DB request according to the requested data<br>4. Treatment of the retrieved data | | |
| **Pass criteria** | - The forecast module is able to decode the queries properly<br>- The forecast module is able to retrieve information from the long-term DB with the parsed information | | |
| **Suspension criteria** | | | |

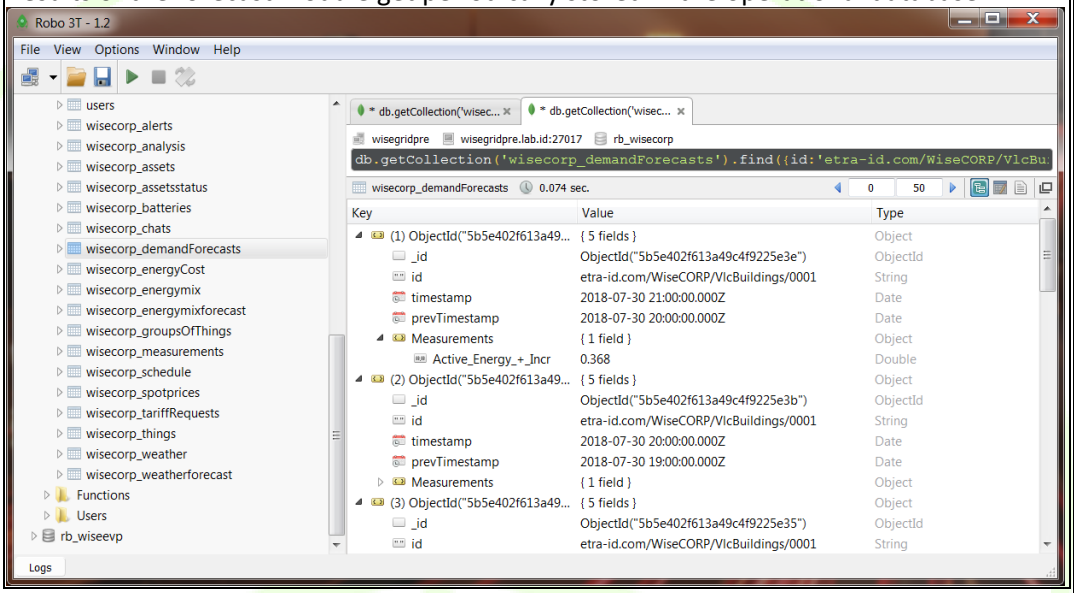
| | |
|---|---|
| **Results** | *The module is able to parse the request messages and process it to retrieve information from the long-term DB.* |

| | | | |
|---|---|---|---|
| **Name** | FOR004. Forecast response message generation test of WiseCORP forecast module | | |
| **Module under test** | WiseCORP forecast module | **Resp.** | ITE |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | RabbitMQ environment<br>WiseCORP forecast module up and running<br>Historical data available in long-term DB | | |
| **Features to be tested** | Performance of WiseCORP forecast module, at generating and submitting the forecast response. | | |
| **Features not to be tested** | | | |
| **Preparation** | Enable RabbitMQ queues<br>Run the forecast module | | |
| **Dependencies** | | | |
| **Steps** | 1. Parsing of the forecasting algorithm output<br>2. Generating forecast response message | | |
| **Pass criteria** | - The forecast module is able to analyse properly the output provided by the forecasting algorithm<br>- The forecast module is able to generate properly the forecast response message | | |
| **Suspension criteria** | | | |
| **Results** | *The module is able to analyse the information provided by the forecast algorithm, and generates the response.* | | |

| | | | |
|---|---|---|---|
| **Name** | FOR005. Forecast is periodically triggered | | |
| **Module under test** | Forecast orchestrator | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | Internal ESB up and running | | |

| | |
|---|---|
| | Historical data available in long-term DB |
| **Features to be tested** | WiseCORP periodically posts a demand and a production forecast request per bus to the corresponding queue of the internal ESB |
| **Features not to be tested** | |
| **Preparation** | Open RabbitMQ monitor, monitor demand and production forecast queues |
| **Dependencies** | |
| **Steps** | 2. Execute forecast orchestrator module |
| **Pass criteria** | - Periodically, every hour, one request per smart meter appears in the demand and production forecast queues<br>- Requests claim next 24 hours hourly prediction |
| **Suspension criteria** | |
| **Results** | Test successful<br>The following extract of logs of the Docker container wisecoop_forecastbridge_demand_1 shows that one forecast query for each asset is being posted every hour.<br>`etraid@wisegridpre:~$ docker logs -t --tail 1000 wisecorp_forecastbridge_demand_1 \| grep querying \| grep etra-id-com-WiseCORP-VlcBuildings-0001-C1`<br>`2018-07-19T11:15:06.654098746Z [etra-id-com-WiseCORP-VlcBuildings-0001-C1] querying...`<br>`2018-07-19T12:15:09.975139716Z [etra-id-com-WiseCORP-VlcBuildings-0001-C1] querying...`<br>`2018-07-19T13:15:04.758048669Z [etra-id-com-WiseCORP-VlcBuildings-0001-C1] querying...`<br>`2018-07-19T14:15:04.668450527Z [etra-id-com-WiseCORP-VlcBuildings-0001-C1] querying...` |

| | | | |
|---|---|---|---|
| **Name** | FOR006. Forecast results are saved to operational DB | | |
| **Module under test** | Forecast orchestrator | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | Internal ESB up and running<br>Historical data available in long-term DB<br>Demand and production forecast modules up and running | | |
| **Features to be tested** | WiseCORP receives the results of the forecast module, formats them following the same format used to store real-time data, and stores the in the operational database | | |
| **Features not to be tested** | | | |

| | |
|---|---|
| **Preparation** | Open operational database, query next 24 hours of demand/production forecasts |
| **Dependencies** | |
| **Steps** | 1. Execute forecast orchestrator module |
| **Pass criteria** | Periodically, every hour, next 24 hours forecast metrics get updated in the operational database |
| **Suspension criteria** | |
| **Results** | Test successful<br>Results of the forecast module get periodically stored in the operational database<br>  |

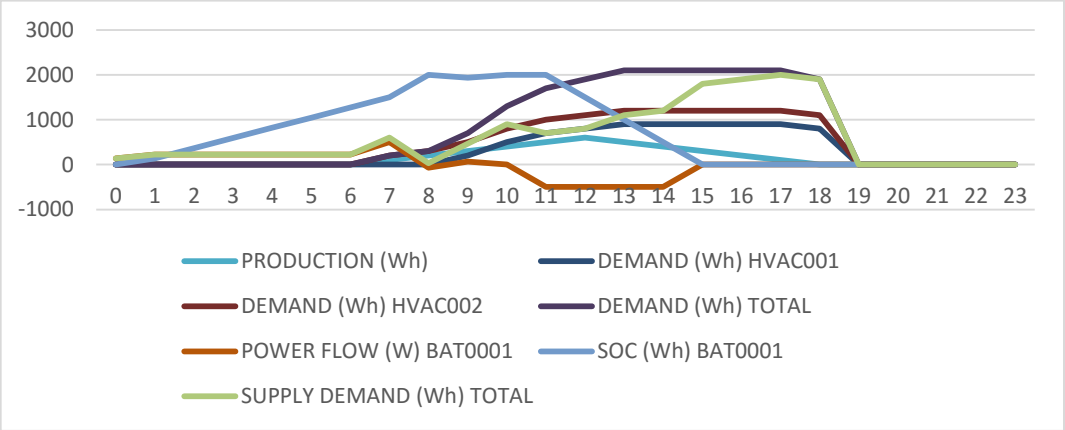


### 4.2.4 Tariff comparer tests

| | | | |
|---|---|---|---|
| **Name** | TC001. Create simulated bill for building | | |
| **Module under test** | Tariff comparer | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | WiseCORP UI up and running<br>Existing predefined set of tariffs | | |
| **Features to be tested** | WiseCORP facilitates to ESCOs the ability to simulate bills according to historical demand data and tariff definitions | | |
| **Features not to be tested** | | | |
| **Preparation** | Open WiseCORP UI | | |

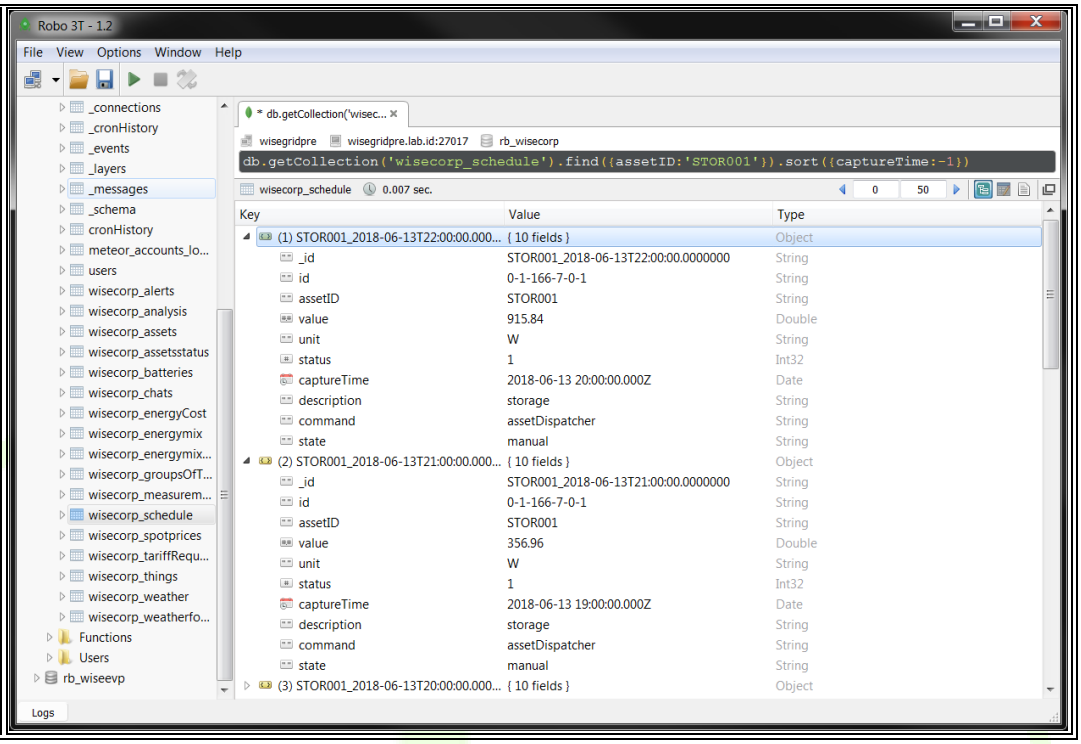
<table>
<tr><td>Dependencies</td><td></td></tr>
<tr><td>Steps</td><td>1. Go to the billing section<br>2. Select a building<br>3. Trigger bill simulation</td></tr>
<tr><td>Pass criteria</td><td>The user interface shows a summary with the comparison of results of the simulated tariffs<br>The user interface displays the details of the bill for the selected criteria</td></tr>
<tr><td>Suspension criteria</td><td></td></tr>
<tr><td>Results</td><td>Test successful<br>The tariff optimization section of the UI can be used to successfully create a simulated bill for any of the tariffs created in the lab-testing environment.<br>May 2018<br>Site: Valencia | Sort solutions by: Global cost | Solution: 11846.38 € > Ahorro Pro Luz 3.0 (Aldro Energía)<br><table><tr><th>Month</th><th>Days</th><th>Entities</th><th>Peak active power (kW)</th><th>at</th><th>Total consumption (kWh)</th><th>Total cost (€)</th></tr><tr><td>May 2018</td><td>31</td><td>5</td><td>315.75</td><td>23/05/2018 11:45</td><td>76699</td><td>4751.66</td></tr></table><table><tr><th>Plan name</th><th>Company</th><th>Tariff type</th><th>Green</th><th>Fixed prices</th><th>Discount</th></tr><tr><td>Ahorro Pro Luz 3.0</td><td>Aldro Energía</td><td>3.0A</td><td>Yes</td><td>Yes</td><td>Yes</td></tr></table><table><tr><th>Concept</th><th>Price</th><th>Value</th><th></th><th>Total</th></tr><tr><td>Contracted power (P1)</td><td>0.116746 €/kW/day</td><td>315 kW</td><td></td><td>1140.02 €</td></tr><tr><td>Contracted power (P2)</td><td>0.072233 €/kW/day</td><td>286 kW</td><td></td><td>640.42 €</td></tr><tr><td>Contracted power (P3)</td><td>0.049977 €/kW/day</td><td>185 kW</td><td></td><td>286.62 €</td></tr><tr><td></td><td></td><td></td><td>Total contracted power</td><td>2067.06 €</td></tr><tr><td>Consumed energy (P1)</td><td>0.140545 €/kWh</td><td>20717 kWh</td><td></td><td>2911.67 €</td></tr><tr><td>Consumed energy (P2)</td><td>0.119795 €/kWh</td><td>39179 kWh</td><td></td><td>4693.45 €</td></tr><tr><td>Consumed energy (P3)</td><td>0.08652 €/kWh</td><td>16803 kWh</td><td></td><td>1453.8 €</td></tr><tr><td></td><td></td><td></td><td>Total consumed energy</td><td>9058.91 €</td></tr><tr><td>Energy consumption discount</td><td>20 %</td><td>9058.91 €</td><td></td><td>-1811.78 €</td></tr><tr><td></td><td></td><td></td><td>Total discounts</td><td>-1811.78 €</td></tr><tr><td>Energy taxes</td><td>5.11269632 %</td><td>9314.19 €</td><td></td><td>476.21 €</td></tr><tr><td>VAT</td><td>21 %</td><td>9790.4 €</td><td></td><td>2055.98 €</td></tr><tr><td></td><td></td><td></td><td>Global total</td><td>11846.38 €</td></tr></table></td></tr>
</table>

### 4.2.5 Energy usage optimizer

| | | | |
|---|---|---|---|
| **Name** | EUO001. Unit testing | | |
| **Module under test** | Energy Usage Optimizer | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | MATLAB<br>Predefined set of input curves (prices, energy demand)<br>Model of 2 HVACs and 1 battery | | |
| **Features to be tested** | The energy usage optimizer must calculate the optimum 24-hours long schedule for the given assets, considering usage calendar and energy price. | | |
| **Features not to be tested** | | | |
| **Preparation** | Prepare input files for algorithm | | |
| **Dependencies** | | | |
| **Steps** | 1. Run algorithm on MATLAB with prepared input files | | |
| **Pass criteria** | The optimum energy usage distribution for the simulated 24 hours is produced | | |
| **Suspension criteria** | | | |
| **Results** | Test successful<br>As an example, the results of the optimum schedule for a battery located in a building with 2 HVACs, PV production and subject a three-period tariff are presented (subject to economic cost minimization).<br>  | | |

| | |
|---|---|
| |  |

| | | | |
|---|---|---|---|
| **Name** | EUO002. Produce day-ahead optimum schedule for assets | | |
| **Module under test** | Energy Usage Optimizer | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | WiseCORP lab-testing environment<br>Operational database contains necessary inputs, including the list of assets, usage calendar and dynamic prices from the DR framework module | | |
| **Features to be tested** | Upon completion of the execution of the energy usage optimizer module, results are stored in the operational database of WiseCORP | | |
| **Features not to be tested** | | | |
| **Preparation** | Prepare lab-testing platform to mimic the context of unit testing | | |
| **Dependencies** | EUO002. Read dynamic price | | |
| **Steps** | 1. Executed energy usage optimizer module | | |
| **Pass criteria** | After algorithm execution completes, the operational database of WiseCORP contains the 24-hour schedule for the configured assets. | | |
| **Suspension criteria** | | | |
| **Results** | Test successful<br>The results of the Energy Usage Optimizer are successfully transferred to the operational database, therefore available to other modules of the tool | | |




### 4.2.6 DR framework tests

| | | | |
|---|---|---|---|
| **Name** | DRF001. Estimate occupant thermal/visual comfort profile | | |
| **Module under test** | Profiling | **Resp.** | HYP |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | WiseCORP up and running, temperature/ humidity/ luminance sensors reporting ambient conditions, HVAC operational setpoint monitoring capabilities | | |
| **Features to be tested** | Generation of the comfort profile for the individual occupants regarding thermal and visual comfort | | |
| **Features not to be tested** | Optional | | |
| **Preparation** | WiseCORP up and running, sensor/ actuator network working properly | | |
| **Dependencies** | | | |
| **Steps** | 1. Listen to internal WiseCORP ESB to collect sensor readings about ambient conditions, HVAC operational setpoint<br>2. Data cleaning, filtering and normalization<br>3. Launch of machine learning algorithm to update comfort profile (conditional to data validity/value) | | |

| | |
|---|---|
| **Pass criteria** | Availability of comfort profile per building thermal zone or occupant (depending on feasibility/granularity of control) |
| **Suspension criteria** | Non-availability of data regarding ambient conditions or HVAC setpoint |
| **Results** | Test successful |

| | | | |
|---|---|---|---|
| **Name** | DRF002. Calculate human-centric demand flexibility of building | | |
| **Module under test** | Flexibility Estimation | **Resp.** | HYP |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | WiseCORP up and running, temperature/ humidity/ luminance sensors reporting ambient conditions, energy (sub)metering data, HVAC operational setpoint monitoring capabilities | | |
| **Features to be tested** | Estimation of demand flexibility time series | | |
| **Features not to be tested** | | | |
| **Preparation** | 1. Association of (sub)metering information with HVAC/lighting system setpoints to calculate the impact of devices on energy consumption<br>2. Set up stream of data from sensors, energy meters and device setpoints by capturing the messages on the WiseCOOP internal ESB | | |
| **Dependencies** | EUO003. Produce day-ahead optimum schedule for assets<br>DRF008. Estimate occupant thermal/visual comfort profile | | |
| **Steps** | 1. Create energy models of HVAC system and lighting system<br>2. Obtain day-ahead building asset operational setpoint schedule<br>3. Estimate available flexibility based on comfort profiles and asset operational schedule<br>4. Generate demand flexibility forecast | | |
| **Pass criteria** | Continuous generation and dispatch of demand flexibility forecasts per time interval | | |
| **Suspension criteria** | | | |
| **Results** | Test successful | | |

| | |
|---|---|
| **Name** | DRF003. Receive request to activate demand flexibility |

| | | | |
|---|---|---|---|
| **Module under test** | Control Optimization | **Resp.** | HYP |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | WiseCORP up and running, WiseCOOP up and running, IOP up and running | | |
| **Features to be tested** | Reception of the appropriate message from WiseCOOP specifying the detailed break-down of demand flexibility per device to be activated | | |
| **Features not to be tested** | | | |
| **Preparation** | Listen to the appropriate queue of the IOP for messages from WiseCOOP | | |
| **Dependencies** | | | |
| **Steps** | 1. Instantiate listener to wait for messages in the correct queue of the IOP<br>2. Upon reception of message, validate its structure and contents | | |
| **Pass criteria** | The message – an example of which can found in Table 19 – has been correctly received and validated. | | |
| **Suspension criteria** | | | |
| **Results** | Test successful | | |

| | | | |
|---|---|---|---|
| **Name** | DRF004. Asset schedule modification | | |
| **Module under test** | DR framework | **Resp.** | HYP |
| **Module requirement** | HL-UC 7_PUC_1_Dynamic management of demand side assets in tertiary sector | | |
| **Test environment** | WiseCORP up and running | | |
| **Features to be tested** | Estimation of optimal setpoint per device and dispatch to the “asset dispatcher” component that sends the setpoints to the device wrappers. | | |
| **Features not to be tested** | | | |
| **Preparation** | WiseCOOP has sent a request for modification of the demand of specific assets. | | |
| **Dependencies** | | | |
| **Steps** | 1. Decode the incoming message from WiseCOOP<br>2. Identify the target demand modification per asset<br>3. Translate demand modification to target setpoint<br>4. Send target setpoint per device to the WiseCORP asset | | |

| | |
|---|---|
| | dispatched component by placing them in the appropriate queue of the WiseCORP-internal ESB. |
| **Pass criteria** | The setpoints that will induce the required demand modifications per device are communicated to the asset dispatcher component. |
| **Suspension criteria** | |
| **Results** | Test successful |

### 4.2.7 Asset dispatcher

| | | | |
|---|---|---|---|
| **Name** | AD001. Load schedule from operational database | | |
| **Module under test** | Asset dispatcher | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | WiseCORP lab-testing environment<br>Operational database contains the schedule for each controllable asset configured in the building | | |
| **Features to be tested** | The asset dispatcher module can read from operational database all necessary information about the schedule of the controllable assets | | |
| **Features not to be tested** | | | |
| **Preparation** | Prepare lab-testing platform to mimic the context of unit testing | | |
| **Dependencies** | EUO003. Produce day-ahead optimum schedule for assets | | |
| **Steps** | 1. Executed asset dispatcher | | |
| **Pass criteria** | Asset dispatcher has access to the operational database and is able to retrieve the current schedules for each controllable asset | | |
| **Suspension criteria** | | | |
| **Results** | Test successful<br>The asset dispatcher accesses the information contained in the schedule collection of the operational database successfully | | |

| | | | |
|---|---|---|---|
| **Name** | AD002. Read current asset status from operational database | | |
| **Module under test** | Asset dispatcher | **Resp.** | ETRA |
| **Module** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and | | |

| | |
|---|---|
| **requirement** | active participation into energy market |
| **Test environment** | WiseCORP lab-testing environment<br>Operational database contains the current status (setpoint) of each one of the controllable assets |
| **Features to be tested** | The asset dispatcher module can read from operational database all necessary information about the current setpoint executed by the controllable assets |
| **Features not to be tested** | |
| **Preparation** | Prepare lab-testing platform to mimic the context of unit testing |
| **Dependencies** | RTM003. Read CHP data from IOP<br>RTM004. Read battery data from IOP<br>RTM005. Read HVAC data from IOP |
| **Steps** | 1. Executed asset dispatcher |
| **Pass criteria** | Asset dispatcher has access to the operational database and is able to retrieve the current status for each controllable asset |
| **Suspension criteria** | |
| **Results** | Test successful<br>The asset dispatcher accesses the information contained in the current status collection of the operational database successfully |

| | | | |
|---|---|---|---|
| **Name** | AD003. Detect deviation from schedule | | |
| **Module under test** | Asset dispatcher | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | WiseCORP lab-testing environment | | |
| **Features to be tested** | Given a point in time when current setpoint and scheduled setpoint differs for a controllable asset, the asset dispatcher must be able to detect the incoherence | | |
| **Features not to be tested** | | | |
| **Preparation** | Prepare lab-testing platform to mimic the context of unit testing<br>Use HVAC simulator to set current setpoint to value that differs from the scheduled one | | |
| **Dependencies** | AD001. Load schedule from operational database<br>AD002. Read current asset status from operational database | | |
| **Steps** | 1. Executed asset dispatcher | | |
| **Pass criteria** | Asset dispatcher detects the deviation in the schedule and logs this detection to the log file | | |

| | |
|---|---|
| **Suspension criteria** | |
| **Results** | Test successful<br>The asset dispatcher successfully compares the necessary information to decide whether a new setpoint must be dispatched |

| | | | |
|---|---|---|---|
| **Name** | AD004. Trigger asset setpoint | | |
| **Module under test** | Asset dispatcher | **Resp.** | ETRA |
| **Module requirement** | HL-UC 7_PUC_2_Dynamic aggregation of distributed energy assets and active participation into energy market | | |
| **Test environment** | WiseCORP lab-testing environment | | |
| **Features to be tested** | Given a point in time when current setpoint and scheduled setpoint differs for a controllable asset, the detection of this incoherence must result in the publication of a command to the controllable asset to set the appropriate setpoint | | |
| **Features not to be tested** | | | |
| **Preparation** | Prepare lab-testing platform to mimic the context of unit testing<br>Use HVAC simulator to set current setpoint to value that differs from the scheduled one | | |
| **Dependencies** | AD003. Detect deviation from schedule | | |
| **Steps** | 1. Executed asset dispatcher<br>2. Connect to IOP and monitor commands to controllable assets (MQTT) | | |
| **Pass criteria** | Asset dispatcher publishes a command targeting the controlled asset in which the deviation was detected | | |
| **Suspension criteria** | | | |
| **Results** | Test successful<br>The following extract of the logs of the assets dispatcher shows the command dispatched via MQTT upon detection of a deviation from the current status of an HVAC assets and its schedule<br>`etraid@wisegridpre:~$ docker logs --tail 0 -f wisecorp_asset_dispatcher_1`<br>`>> [HVAC001/SHIC01/0-1-160-7-0-1] { "_id" : "0-1-160-7-0-1", "assetID" : "HVAC001", "value" : "25", "unit" : "grdC", "status" : 1, "captureTime" : ISODate("2018-07-19T15:16:03.864Z"), "description" : "modbus", "mode" : "cooling", "command" : "auto", "state" : "manual" }` | | |

# 5 CONCLUSIONS AND NEXT STEPS

The main conclusion of the work presented in this deliverable is that the methodology followed during the implementation and lab-testing phase was optimal for both tools. The standardization of a process and its explanation to the partners involved in this phase, allowed to avoid any misunderstanding and to follow the same steps so the final result is a coherent and homogeneous work.
All the tests and activities performed within this deliverable have been successful even if, in some cases, it has been necessary to refine the implementation of the modules and repeat the tests, which allowed the involved partners to better understand the singularities of each module.

Although it has not been possible to make all the test that the partners would like to perform for this tool, the main ones and some complementary ones have been done. For Task 14.2 “WiseGRID integrated ecosystem Lab-Testing” more tests will be performed in order to prove the integration of the different tools together. During the deployment and demonstration phases, as all the tools will be integrated in real-life conditions and the consortium will have better knowledge of the particularities of each Pilot Site, the partners will be able to collect some feedback and continuing refining the tools and perform some more tests in order to develop the tools and optimally adapt them for the different Pilot Sites.

# 6 REFERENCES AND ACRONYMS

## 6.1 REFERENCES


[1] Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Software_test_documentation. [Accessed 14 July 2016].

[2] The International Software Testing Standard, "ISO/IEC/IEEE 29119 Software Testing," 10 September 2014. [Online]. Available: http://www.softwaretestingstandard.org/. [Accessed 14 July 2016].

[3] G. Arnold, D. Craciun, W. Heckmann and N. Schäfer , "Utility-scale PV installations and their challenges in grid-code compliance testing," *storage & grids: Technical Briefing,* pp. 74-78, February 2015.

[4] "D7.1 WiseCOOP and WiseCORP Apps Design".

[5] [Online]. Available: https://bl.ocks.org/rpgove/0060ff3b656618e9136b.

[6] "D10.2 WiseGRID Flexibility-based DR Optimization Framework Specifications".


## 6.2 ACRONYMS

**Table 27 – Acronyms list**

| Acronyms List | |
|---|---|
| AMI | Advanced Metering Infrastructure |
| AMQP | Advanced Message Queuing Protocol |
| CHP | Combined Heat Power |
| DB | Data Base |
| DR | Demand Response |
| DSO | Distribution System Operator |
| ENTSOE | European Network of Transmission System Operators for Electricity |
| ESB | Enterprise Service Bus |
| ESCO | Energy Service COmpany |
| GUI | Graphical User Interface |
| HL-UC | High Level Use Case |
| HTTP | Hypertext Transfer Protocol |
| HVAC | Heating, Ventilation and Air Conditioning |
| IOP | InterOperable Platform (one of the WiseGRID products) |
| KPI | Key Performance Indicator |

| | |
|---|---|
| MQTT | Message Queue Telemetry Transport |
| RES | Renewable Energy Source |
| RPC | Remote Procedure Call |
| RT | Real-Time |
| SMX | Smart Meter eXtension |
| UI | User Interface |
| USEF | Universal Smart Energy Framework |