

| Title:  | Document Version: |
|---|-------------------|
| D12.2 RESCO services and Advanced models for smartening the distribution grid | 1.0               |

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

| Contractual Delivery Date: | Actual Delivery Date: | Deliverable Type*-Security*: |
|----------------------------|-----------------------|------------------------------|
| M18 (April 2018)           | M18 (April 2018)      | R-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: |
|--|---------------|------------------|
| Giuseppa Caruso (ENG) , Leandro Lombardo (ENG) | ENG           | WP12             |

#### Authors (organisation):

Giuseppa Caruso (ENG), Leandro Lombardo (ENG), Alberto Zambrano (ETRA), Álvaro Nofuentes (ETRA), Mihai Sanduleac (CRE), Paul Lacatus (CRE), Catalin Chimirel (CRE), Panos Karafotis (ICCS), Stela Sarri (ICCS), Foivos Palaogiannis (ICCS), Amparo Mocholi (ITE), Javier Monreal Tolmo (ITE), Jorge Sanjuán (AMP), Xavier Benavides (AMP), Benjamin Kraft (VS)

#### Abstract:

This document includes two main topics. On one hand it describes the RESCO Tool characteristics and the services provided by it to RESCO companies, as they have been developed in the T12.2. On the other hand it provides a description of some functionalities for advancing and smartening the Distribution Grid according the T12.3 activities that will be the starting points for the WG Cockpit development carried out in the WP13.

#### Keywords:

RESCO Tool, Smart grid services, RESCO services,



## Revision History

| Revision | Date       | Description  | Author (Organisation)  |
|----------|------------|--|--|
| V0.1     | 20.02.2018 | ToC  | Giuseppa Caruso, Leandro Lombardo (ENG)  |
| V0.2     | 24.02.2018 | ToC improvements   | Giuseppa Caruso, Leandro Lombardo (ENG)  |
| V0.3     | 13.03.2018 | Contribution to chapters:1,2, 3 and 4  | Leandro Lombardo (ENG), Alberto Zambrano (ETRA), Foivos Palaogiannis (ICCS), Amparo Mocholi (ITE)  |
| V0.4     | 26.03.2018 | Included draft version of Introduction and conclusion, and some improvements       | Amparo Mocholi (ITE), Mihai Sanduleac (CRE), Leandro Lombardo (ENG)  |
| V0.5     | 06.04.2018 | Added contribution to chapter 6 and some improvements to the other ones            | Foivos Palaogiannis (ICCS), Paul Lacatus (CRE), Giuseppa Caruso, Leandro Lombardo (ENG)  |
| V0.6     | 16.04.2018 | Added final contribution and provided version ready for internal review            | Giuseppa Caruso, Leandro Lombardo (ENG), Foivos Palaogiannis (ICCS), Alberto Zambrano (ETRA).  |
| V1.0     | 24.04.2018 | Peer review. Addressed comments from the peer review. Version ready for submission | Giuseppa Caruso, Leandro Lombardo (ENG), Foivos Palaogiannis (ICCS), Alberto Zambrano, Álvaro Nofuentes (ETRA), Catalin Chimirel (CRE), Amparo Mocholi (ITE), Javier Monreal Tolmo (ITE) |

## INDEX

|  |           |
|--|-----------|
| <b>EXECUTIVE SUMMARY .....</b>                           | <b>8</b>  |
| <b>1 INTRODUCTION .....</b>                              | <b>12</b> |
| 1.1 PURPOSE OF THE DOCUMENT.....                         | 12        |
| 1.2 SCOPE OF THE DOCUMENT .....                          | 12        |
| 1.3 STRUCTURE OF THE DOCUMENT .....                      | 12        |
| <b>2 RESCO TOOL BACKGROUND .....</b>                     | <b>14</b> |
| 2.1 BUSINESS FRAMEWORK OF THE RESCO TOOL.....            | 14        |
| 2.2 OVERVIEW OF RESCO TOOL FUNCTIONALITIES .....         | 15        |
| 2.2.1 Architecture Overview.....                         | 15        |
| 2.2.2 Asset Manager Module .....                         | 17        |
| 2.2.2.1 Module Description .....                         | 17        |
| 2.2.2.2 Module Data Model .....                          | 18        |
| 2.2.3 Maintenance Management Module.....                 | 20        |
| 2.2.3.1 Module Description .....                         | 20        |
| 2.2.3.2 Module Data model.....                           | 22        |
| 2.2.3.3 Workflow .....                                   | 25        |
| 2.2.4 Contract Manager Module .....                      | 25        |
| 2.2.4.1 Module Description .....                         | 25        |
| 2.2.4.2 Module Data Model .....                          | 27        |
| 2.2.5 Billing Manager Module .....                       | 29        |
| 2.2.5.1 Module Description .....                         | 29        |
| 2.2.5.2 Module Data Model and algorithmic framework..... | 31        |
| 2.2.6 Energy Metering Manager Module .....               | 32        |
| 2.2.6.1 Module Data Model and Algorithmic Framework..... | 35        |
| 2.2.7 Energy Forecasting Manager Module .....            | 36        |
| 2.2.7.1 Module Description .....                         | 36        |
| 2.2.7.2 Module Data Model and Algorithmic Framework..... | 37        |
| 2.2.8 Market Bid Manager Module .....                    | 38        |
| 2.2.8.1 Module Description .....                         | 38        |
| 2.2.8.2 Module Data Model and Algorithmic Framework..... | 39        |
| 2.2.9 Investment Decision Support Manager Module .....   | 40        |

|          |   |           |
|----------|---|-----------|
| 2.2.9.1  | Module Description .....  | 40        |
| 2.2.9.2  | Module Data Model and Algorithmic Framework.....  | 40        |
| <b>3</b> | <b>DEVELOPMENT VIEW OF RESCO TOOL.....</b>  | <b>41</b> |
| 3.1      | RESCO COMMON REPOSITORY .....   | 41        |
| 3.2      | RESCO WEB-APP.....  | 42        |
| <b>4</b> | <b>DEPLOYMENT, PROTOTYPING AND TESTING ENVIRONMENT .....</b>  | <b>43</b> |
| <b>5</b> | <b>ADVANCED MODELS FOR SMARTENING THE DISTRIBUTION GRID .....</b>                                   | <b>45</b> |
| 5.1      | SUPPORTING THE DISTRIBUTION GRID OPERATION WITH DISTRIBUTION MANAGEMENT SYSTEM (DMS) SERVICES ..... | 45        |
| 5.2      | FUNCTIONALITIES AND SERVICES OF A DMS.....  | 46        |
| 5.2.1    | Demand and Production Forecast services .....   | 46        |
| 5.2.2    | State Estimation and Load Flow .....  | 49        |
| 5.2.3    | Energy Quality .....  | 53        |
| 5.2.4    | Congestion Forecast .....   | 56        |
| 5.2.5    | Grid Fault Management.....  | 57        |
| 5.2.5.1  | Modules providing alerts.....   | 58        |
| 5.2.5.2  | Incident management modules .....   | 59        |
| 5.3      | INTEGRATION OF DMS SERVICES TO SUPPORT DISTRIBUTION GRID AND SCADA OPERATION. ....                  | 66        |
| <b>6</b> | <b>CONCLUSIONS .....</b>  | <b>67</b> |
| <b>7</b> | <b>REFERENCES AND ACRONYMS.....</b>   | <b>69</b> |
| 7.1      | REFERENCES.....   | 69        |
| 7.2      | ACRONYMS.....   | 71        |
| <b>8</b> | <b>ANNEX A.....</b>   | <b>72</b> |
| 8.1      | SWAGGER SPECIFICATION .....   | 72        |
| 8.2      | CIM METERREADING MESSAGE EXAMPLE .....  | 84        |

## LIST OF FIGURES

|   |    |
|---|----|
| Figure 1 – RESCO tool architecture .....  | 16 |
| Figure 2 – Equipment inventory .....  | 17 |
| Figure 3 – Equipment connection point .....   | 18 |
| Figure 4 – Maintenance manager.....   | 20 |
| Figure 5 – Schematic illustration of a Maintenance Management System.....               | 22 |
| Figure 6 – Screenshot of the Maintenance Module Web Interface.....                      | 23 |
| Figure 7 – Screenshot of the Maintenance Module Android App.....                        | 24 |
| Figure 8 – Customer manager .....   | 26 |
| Figure 9 – Contract manager .....   | 27 |
| Figure 10 – Billing manager .....   | 30 |
| Figure 11 – Res Asset Energy P/C .....  | 32 |
| Figure 12 – RESCO tool Smart meter data flow from SMX to DB.....                        | 33 |
| Figure 13 – RESCO tool AMI data flow from SMX to DB.....                                | 34 |
| Figure 14 – RESCO tool battery data flow from SMX to DB .....                           | 34 |
| Figure 15 – Supply point Forecast .....   | 37 |
| Figure 16 – Market Participation.....   | 39 |
| Figure 17 – Application access to Big-data Platform.....                                | 41 |
| Figure 18 – Object Mapping between Node and MongoDB managed via Mongoose [6] .....      | 42 |
| Figure 19 – GitLab login page .....   | 44 |
| Figure 20 – GitLab software repository page .....                                       | 45 |
| Figure 21 – CNEA network structure .....  | 47 |
| Figure 22 – Training workflow schema.....   | 48 |
| Figure 23 – Forecast workflow schema .....  | 49 |
| Figure 24 – Impact of DERs on the network voltage profile .....                         | 50 |
| Figure 25 – Typical SE architecture.....  | 52 |
| Figure 26 – Different kinds of data that can be fed from PMUs to the PDC.....           | 52 |
| Figure 27 – Examples of LF failure .....  | 53 |
| Figure 28 – Characteristics of voltage dips.....  | 56 |
| Figure 29 – Overview of an incident management system.....                              | 58 |
| Figure 30 – Monitored voltage in a low voltage bus.....                                 | 58 |
| Figure 31 – Voltage distribution across 58000 samples in LV .....                       | 59 |
| Figure 32 – Example of usage of flexibility to change overall demand profile [29] ..... | 60 |
| Figure 33 – Functional diagram .....  | 61 |
| Figure 34 – Fault location principle. Source: [33].....                                 | 62 |

|   |    |
|---|----|
| Figure 35 – Basic Service Restoration flowchart ..... | 64 |
| Figure 36 – Basic Genetic Algorithm flowchart .....   | 65 |
| Figure 37 – Basic BFS flowchart .....                 | 66 |

## LIST OF TABLES

|  |    |
|--|----|
| Table 1 – Assets data model .....                                      | 18 |
| Table 2 – Maintenance module .....                                     | 24 |
| Table 3 – Workflow overview of the maintenance management module ..... | 25 |
| Table 4 – Customer data model .....                                    | 27 |
| Table 5 – Contract typology data model .....                           | 28 |
| Table 6 – Customer contract data model .....                           | 29 |
| Table 7 – Billing data model .....                                     | 31 |
| Table 8 – Battery data model .....                                     | 35 |
| Table 9 – PV data model .....  | 36 |
| Table 10 – Market participation data model .....                       | 39 |
| Table 11 – Upper limits of individual harmonic voltages .....          | 54 |
| Table 12 – Classification of voltage dips .....                        | 56 |
| Table 13 – List of Acronyms .....                                      | 71 |

## EXECUTIVE SUMMARY

Power generation is moving towards more and more renewables due to environmental constraints that cannot be withheld anymore. In fact, in line with the European energy and climate policy targets of reducing the greenhouse gas (GHG) emissions, electricity will be increasingly produced from renewable energy sources (RES) that are largely intermittent and unpredictable, leading to a substantial challenge to grid stability and security of supply. For that reason too, integration of RES market has currently small share, in the Smart Grid Market. Mainly related to the development of small scale RES projects it has witnessed a growth the last years due to the economic viability of RES, mainly the solar and the wind.

Such RES distribution is challenging the distribution grid and therefore many improvements are going to be in place and these are mostly related to distribution grid management, additional DMS functionalities, storage facilities and RESCOs (Renewable Energy Service Companies) evolution. New concepts like the RESCO business model, which are yet immature in most EU countries, could be capable to assist the penetration of RES in the distribution system and increase the investments. In that panorama, the main purpose of RESCOs, is to provide RES services to final consumers that do not own nor have the proper capabilities to maintain the necessary equipment, being involved over the entire lifetime of the installed assets. RESCOs operate on a business model in which they partner with investors to own the resources (e.g. solar panels, domestic wind turbines, and batteries), using long term capital investments, generating power, selling it and collecting revenues from their customers. Usually, RESCOs collect and centralize information from small and medium size RES equipment that have no visibility to the DSO individually, but which together may have a considerable impact in the stability of the network. Moreover, the RESCO model has deep environmental positive impact by encouraging and increasing the share of RES integration into the network. WG RESCO covers most of the processes that have been identified as necessary in such organizations like the monitoring, the management, the maintenance, etc. Currently there is no specialised ICT tool in the European market to cover RESCO activities and such organizations require specific ICT tools to enhance their operation.

The purpose of this document is to describe the WG RESCO Tool characteristics and the services provided by it to RESCO companies, as they have been developed in the T12.2 "RESCO services" as well as to provide a description of the "Advanced models for Smartening the Distribution Grid" according the T12.3 activities which will be taken in account in the WP13 "WiseGRID Cockpit" implementation.

In that direction, the RESCO tool aim to provide a set of services to support the RESCO companies. As depicted in the WG RESCO tool architecture (Figure 1), this tool is based on three fundamental layers – Data capture/provider, Back-end, Front-end – and on a Big Data platform to store data. Each component of the architecture contributes to the proper functioning of all the features offered by the WG RESCO tool thanks to a continuous interaction.

- **Data capture/provider** layer is the part of the architecture that includes all those services created to collect data from different sources such as smart meter, sensors, weather provider, asset maintenance management services, energy forecasting services and all those other services that if necessary can provide data useful for the proper functioning of the WG RESCO tool.
- **Back-end** layer is represented by a component of the WiseGRID project called WG IOP (WiseGRID interoperable platform), which assures the exchange of data and information between different tools and services within the project.
- **Front-end** layer is represented by the RESCO tool web application. In this phase of the project the tool is divided into eight modules, each of which has a precise role in the complete operation of the tool itself. Currently the front-end modules are :
  - **Asset manager** which implements all the functionalities that allow the RESCO to manage all the asset present inside its portfolio.
  - **Maintenance manager** which enables the RESCO to follow the equipment status, to



manage scheduled maintenance actions as well as to monitoring and manage specific incidents.

- **Contract manager** which enables the RESCO to manage the customer's portfolio by means of a set of functionalities for register new customers to the system, manage and cease it as well as to register and manage a contract between the company and a specific customer present at system.
- **Billing manager** allows the RESCO operator to be able to easily determine, based on the type of contract with the final customer, the amount to be billed to its customers. It take also in account the energy produced and consumed for each supply point.
- **Energy metering manager** enables to meter and gather into the corresponding databases all the necessary data from the "Data capture/provider layer" (from smart meters, AML, batteries etc.).
- **Energy surplus forecasting manager** enables to have the energy production and consumption forecasting for the day ahead in order to be able to estimate the availability of energy surplus to be traded on the wholesales energy market.
- **Market bid manager**, support the RESCO user to define the appropriate bid in term of energy price and quantity to be launched into the Wholesale energy market. It take in account both the real-time energy monitoring and the energy estimation (forecasting) for the day ahead.
- **Investment decision support manager** supports the RESCO on the decisions to be taken about its investments. In particular, this module enables to decide, supported by the estimations based on historical data. The RESCO by means of it should be easier to identify the best typology of equipment to be installed in a specific point or the more appropriate contract to be proposed to the customers.

The RESCO tool requires for long term data storage access to a centralised, cloud based database. The database will be used for storing a database containing the data related to the RESCO application.

The cloud based database management system that will be used in WiseGRID project is a No SQL database management system. The NoSQL databases are used mainly in Big Data systems. The Big Data systems are systems used for storing and processing huge amount of data and they has to be easily scalable in order to be able to store and process continuously increasing amount of data.

For WiseGRID project and also for the WG RESCO tool, the implementation of Big Data platform will use an open source database management system named MongoDB that is a leading NoSQL database.

Currently, as is also shown in the Figure 1, the WG RESCO tool store its data in two different data base provided by the Big Data Platform.

- The first one the "Local DB" for the Customer data, Contract data, Asset data, Configuration data and real time meter observation coming from RES source and stored by the RT Monitor tool.
- The second one the "Long term DB" to store historical data like meter observation coming from RES source and all the data processed by the WG RESCO tool that needs to be stored for future elaborations.

Two different approaches were adopted to read and write data in these two databases:

- Direct access to the Local DB and Long Term DB through specific java script of the RESCO tools;
- Access through external module that by the WiseGRID Interoperable Platform (WG IOP) is able to share data read from the Long Term DB and store in it.

The web-app of the RESCO tool allows the logged-in user to benefit from the different functions that each module defines. In order to harness its architecture it has been developed using the MEAN.JS full-stack JavaScript solution, this choice has been made to build fast, robust, and maintainable production web applications using MongoDB, Express, AngularJS, and adopting Node.js as engine. The web-app is exposed

on the network thanks to Nginx reverse proxy.

Finally, with regards the hardware and software requirements for the deployment of the full application, the follow requirements have been envisaged:

- For hardware requirements, a 4-core (or superior) CPU at 2GHz, 8GB of RAM, 100 GB hard disk and a 64-bit CentOS 6.6 (core linux intsys-trentour 2.6.32-431.29.2.el6.x86\_64) or higher is the minimum of requirement.
- As software requirement the Node Version Manager (NVM) [1] is required into the server that hosts the RESCO tool software with the aim to facilitate the multiple active node.js versions.

There are two ways of deployment the services in local servers: either plain installation in the server machine (if this fulfils the minimum of hardware & software requirements) or setting a Virtual Machine for managing separately the different applications. This approach enables the deployment of the RESCO tool either as a host service or as an application running in a cloud service provider (with the appropriate customizations). The aforementioned deployment analysis defines the list of prerequisites for the deployment of the RESCO tool at the different pilot sites (Flanders and Terni).

The other main topic considered inside this deliverable is the description of advanced models for Smartening the Distribution Grid.

An advanced Distribution Management System is a software platform which incorporates a full set of functionalities that support the management and the operational optimization of the distribution system and allows the personnel to effectively monitor and control it while improving safety, reliability, asset protection and quality of service. It includes monitoring, analysis, control, optimization, training and planning tools that all function on a common representation of the entire electric distribution system. WiseGRID deliverable D12.1 "Analysis of Grid Management technologies for the distribution grid" already reported and described a full list of functionalities that a common DMS usually incorporates. Moreover, one of the primary objectives of the WiseGRID project is to provide "a set of solutions and technologies which increase the smartness, stability and security" of the consumer centric electricity grid. In order to achieve this goal one of the aspects that were identified and need to be addressed is "Smartening the Distribution Grid". This means that a set of functionalities that enhance grid operation and assist DSOs in the monitoring and management of the system need to be designed and developed. Consequently inside the WP12 and specifically the T12.3 "Advanced models for Smartening the Distribution Grid" topics related the enhancement of grid operation have been dealt with, that will be more in detail considered inside the WP13 activities and specifically in the development of the WG Cockpit tool.

The core functionalities that have been identified from the WiseGRID project as fundamental for the tool (WG Cockpit) that will provide the DMS services and that will be developed in the WP13 are, briefly, the following:

- **Demand and Production Forecast.** The demand and production forecast provider is a wrapper of the forecast algorithms for an easy integration with the applications of the WiseGRID ecosystem. The forecasting module developed in the WiseGRID project could be seen as a function that forecasts the next desired values taking into consideration historical data and additional data such as a calendar, exogenous variables (weather) and some client parameters.
- **Load Flow/State Estimation.** The large penetration of Distributed Energy Resources (DERs), consisting of Distributed Generation (DG) units, storage technologies and demand responsive loads has completely modified the network voltage profile. Consequently, it is very important to provide new ways for the load Flow and the State estimation in order to assure the energy grid stability.
- **Energy Quality Monitoring.** Energy Quality has emerged as a very important issue in modern electrical power systems, especially nowadays when the increasing level of renewable power penetration and electric vehicles connection introduces several negative impacts. Smart grids are required

to deliver power at a high-quality level to consumers, as insufficient power quality performance may have detrimental financial effect on both consumers and utilities in distribution systems. The term energy quality indicates the deviation of both voltage and current from their ideal waveforms. An ideal waveform is considered to be sinusoidal, with fixed frequency and amplitude, any deviation from which is considered a disturbance.

- **Congestion Forecast.** This module of the application will deal with grid congestions chances evaluation based on forecast for generation and consumption. It will evaluate various grid operating conditions looking to identify cases where congestions could occur with specific accent on the probability of RES production according forecast report
- **Grid Fault Management.** This is a generic framework for incident processing, proposed in order to be demonstrated in the project to assist the work of the DSOs. It should increase the awareness on the status of the grid. The modules, inside that framework, subscribe to the flow of data coming from field devices and/or other modules in order to analyze those in real-time to trigger alerts when an unexpected behavior is detected.

These functionalities are the fundamental modules that will realise the advanced services of the Distribution Management System by means also a set of input data, generally, classified at the following categories:

- Real time or near real time data.
- Historical and offline data.
- Spatial data.
- Other kind of data (e.g.: weather forecasting).

## 1 INTRODUCTION

### 1.1 PURPOSE OF THE DOCUMENT

The purpose of the document is to describe the RESCO Tool characteristics and the services provided by it to RESCOs, as they have been developed in the T12.2 as well as to provide a description of the Advanced models for Smartening the Distribution Grid according the T12.3 activities which will be taken in account in the WP13 implementation. According to the Use Cases that were defined for the tool and available in the deliverable D2.1 [2], several functionalities are studied and their application methodologies are described, from a technical point of view, along with the way to implement them in the different RESCO Tool's modules. This deliverable also explains how the RESCO Tool can improve the grid and support grid operators to better manage and control the network, in a scenario of increasing share of distributed renewables and other energy resources.

On the other hand, the platform is described from the user's point of view, presenting the user interfaces; as well as with the Tool modules, technologies and services applied in the platform design and programming are also described.

In addition, this document describes in detail the required algorithm for be able to provide information and optimal solutions for acting on the grid to foresee and solve some issues, as congestion situations. Alternative methods for calculating several factors, as Energy Quality or grid status, are explained and the best way to calculate them is chosen and described with further details.

### 1.2 SCOPE OF THE DOCUMENT

This document aims at describing and presenting the main results of the activities carried out in the "T12.2 RESCO services" and "T12.3 Advanced models for Smartening the Distribution Grid". Consequently on one hand, the document has the aim of showing and explaining the application architecture, each single module and its functionalities as well as the technologies used for the development from the type of code written to the environment of continuous integration and versioning. On the other hand it aims at providing a description of functionalities and methodologies designed in the T12.3 in order to improve the grid and support grid operators to better manage and control the network, in an increasing share of distributed renewables. This latter represents also the starting point for the implementation carry out in the WP13 related to the WG Cockpit.

### 1.3 STRUCTURE OF THE DOCUMENT

In order to give the reader enough background information useful for the current document's best comprehension:

- Section 2 RESCO TOOL BACKGROUND provides an overview of the business environment and the motivation for tools like the WG RESCO tool. This chapter also provides an overview of the WG RESCO tool architecture and its main modules are explained and located inside the tools architecture.
- Section 3 DEVELOPMENT VIEW OF RESCO TOOL describes in more detail the RESCO tool repository and the Web APP solution.
- Section 4 DEPLOYMENT, PROTOTYPING AND TESTING ENVIRONMENT summarizes technical specifications regarding the system implementation and deployment.
- Section 5 ADVANCED MODELS FOR SMARTENING THE DISTRIBUTION GRID present the different functionalities to be included and considered for smartening the grid.

- Section 6 CONCLUSIONS provides the final conclusions and the next steps of the work.



## 2 RESCO TOOL BACKGROUND

### 2.1 BUSINESS FRAMEWORK OF THE RESCO TOOL

Power generation is moving towards more and more renewables due to environmental constraints that cannot be withheld anymore. In fact, in line with the European energy and climate policy targets of reducing the greenhouse gas (GHG) emissions, electricity will be increasingly produced from renewable energy sources (RES) that are largely intermittent and unpredictable, leading to a substantial challenge to grid stability and security of supply. For that reason too, integration of RES market has currently small share, in the Smart Grid Market. Mainly related to the development of small scale RES projects it has witnessed a growth the last years due to the economic viability of RES, mainly the solar and the wind.

Such RES distribution is challenging the distribution grid and therefore many improvements are going to be in place and these are mostly related to distribution grid management, additional DMS functionalities, storage facilities and RESCO evolution. New concepts like the RESCO business model, which are yet immature in most EU countries, could be capable to assist the penetration of RES in the distribution system and increase the investments. In that panorama, as described in D12.1, the main purpose of a RESCO is to provide RES services to end-users who do not own or do not wish to own the RES infrastructure (e.g. PVs, inverters, appropriate electrical installation) but who are willing to grant the land (e.g. rooftop, non-exploitable area etc.) to RESCO with some form of exchange. The RESCO takes over the deployment and maintenance of the RES equipment and, as an exchange, exploits the produced energy from the infrastructure. Three different potential scenarios related to RESCO business model, involving three different types of relationships with the customers, were recognized. According to the first scenario, the RESCO pays a fee to the customer for using the premises and trades all the produced energy. At the second scenario, part of the produced energy is used by the customer and the RESCO exploits the rest. And, finally, at the last scenario, RESCO only deploys and maintains the infrastructure, while the produced energy is fully exploited by the customer. Usually, RESCOs collect and centralize information from small and medium size RES equipment that have no visibility to the DSO individually, but which together may have a considerable impact in the stability of the network. Moreover, the RESCO model has deep environmental positive impact by encouraging and increasing the share of RES integration into the network.

WG RESCO covers most of the processes that have been identified as necessary in such an organization like the monitoring, the management, the maintenance etc. Currently there is no specialised ICT tool in the European market to cover RESCO activities but such organizations require specific ICT tools to enhance their operation.

For that reasons, in order to support the RESCOs, WiseGRID project is developing an innovative technology tool for managing processes to deal with customer contracts (e.g. registration, billing, ceasing), installed asset portfolio (e.g. monitoring, maintenance), energy (e.g. self-consumption contracts) and economic flows (e.g. investments per installation/client).

WiseGRID RESCO Tool was designed and developed based on the previously described perspective and understanding for the business model of a RESCO. Fundamentally, this tool needs to provide all the necessary functionalities that support and facilitate the activities of this business actor. For that purpose, WiseGRID RESCO Tool needs to comply with the following:

#### 1. Performs all the necessary functionalities required for the business operations.

The fundamental business processes, in order support the defined scenarios, were described in D12.1 and are the following:

- Manage contracts with customers

- Manage portfolio of installed assets;
- Manage energy flows
- Manage economic flows
- RES production forecast
- Market participation
- Provision of services to DSO.

In order for the tool to be able to implement all the previous processes, the appropriate software modules need to be designed and developed.

### 2. Supports the Interoperability with external tools

RESCO business develops and operates in the new “smart” energy environment. In this new environment, different business actors and models operate using different tools the interoperability of which is often necessary. Tools developed in WiseGRID project are expected to serve this role. According to the defined functionality and use cases of each of the developed tools, WG RESCO Tool needs to exchange data mainly with WiseHOME tool and WiseCORP Tool. To achieve this exchange of data, the tool needs to be able to exchange data with the interoperable platform of the project, WG IOP.

### 3. Incorporates Functional User Interface

The user interface is fundamental for this tool as it aims to be used at a business environment, not necessarily by technical experts. It needs to gather all the necessary data for the administration, monitoring and maintenance purposes in a coherent and comprehensive way, and provide useful information for the marketing and investment decisions.

The design and development of the WG RESCO Tool was realized according to these three aspects. This will be obvious at the following paragraphs at which the tool is thoroughly described and explained.

## **2.2 OVERVIEW OF RESCO TOOL FUNCTIONALITIES**

### **2.2.1 Architecture Overview**

The RESCO tool architecture, as shown in Figure 1, is based on three fundamentals layer, Data capture/provider, Back-end, Front-end and on a Big Data platform to store data. Each component of the architecture contributes to the proper functioning of all the features offered by the RESCO tool thanks to a continuous interaction. Each component will be described below, and the role of each layer will be explained for the complete operation of each function of the tool.



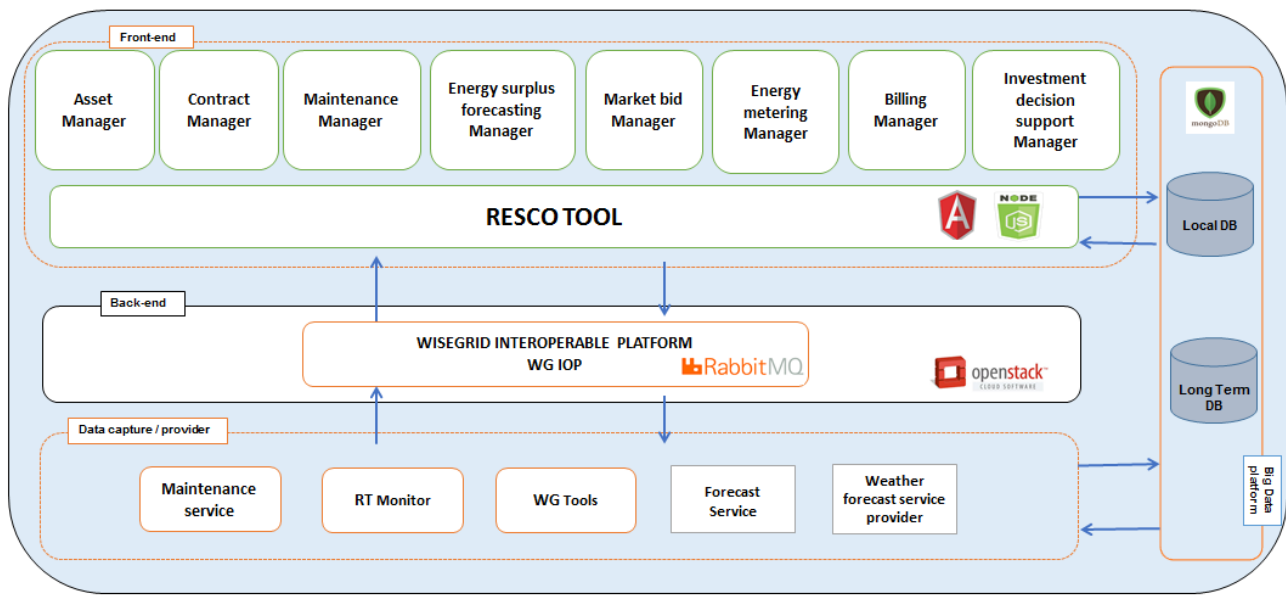


Figure 1 – RESCO tool architecture

**Data capture/provider** layer is the part of the architecture that includes all those services created to collect data from different sources such as smart meter, sensors, weather provider, asset maintenance management services, energy forecasting services and all those other services that if necessary can provide data useful for the proper functioning of the WG RESCO tool. The services in the Data collection/provider layer share the acquired data with each other services and with the WG RESCO tool in two ways. The first one through the interaction with the Back-end layer that is a message-oriented middleware that allows the exchange of messages through MQTT and AMQP protocols, the second one by saving data collected within the Big Data Platform in two different data base for long-term and local data storage.

**Back-end** layer is represented by a component of the WiseGRID project called WG IOP (Wisegrid interoperable platform), the role of this component is to allow the exchange of data and information between different tools and services within the project. This component of the architecture is better described in a dedicated deliverable (D4.2 “WiseGRID interoperable Integrated Process”) but anyway here a high-level view is described. The WG IOP core is a message-oriented middleware that is RabbitMQ. RabbitMQ is an open-source software and implement the “MQ Telemetry Transport or Message Queue Telemetry Transport” (MQTT) and “Advanced Message Queuing Protocol” (AMQP). At this stage of the project the WG IOP is hosted on a public server and security is guaranteed through an authorisation system managed by means of specific openstack rules and an authentication system managed by RabbitMQ.

**Front-end** layer is represented by the RESCO tool web application. In this phase of the project the tool is divided into eight modules, each of which has a precise role in the complete operation of the tool itself. Currently the modules are modules: Asset manager, Contract manager, Maintenance manager, Energy surplus forecasting manager, Market bid manager, Energy metering manager, Billing manager, Investment decision support manager. These modules will be better described in dedicated sections of this document. The RESCO tool is developed mainly in Node.js and Angular.js. Each module exchanges and receives data via the WG IOP through specific queues and at the same time reads and saves data on the Big Data Platform from both the Long Term DB and the Local DB based on the functionality it offers.

**Big Data-platform** is a cloud-based clustered platform, scalable and the access protected by Access credentials. The platform is shared by the entire WiseGRID project and provides the RESCO tool with two different



data bases based on MongoDB. The two data bases called Local DB and Long-term DB respectively contain data pertaining to the tool itself and the observations received from meters / sensors and forecasts.

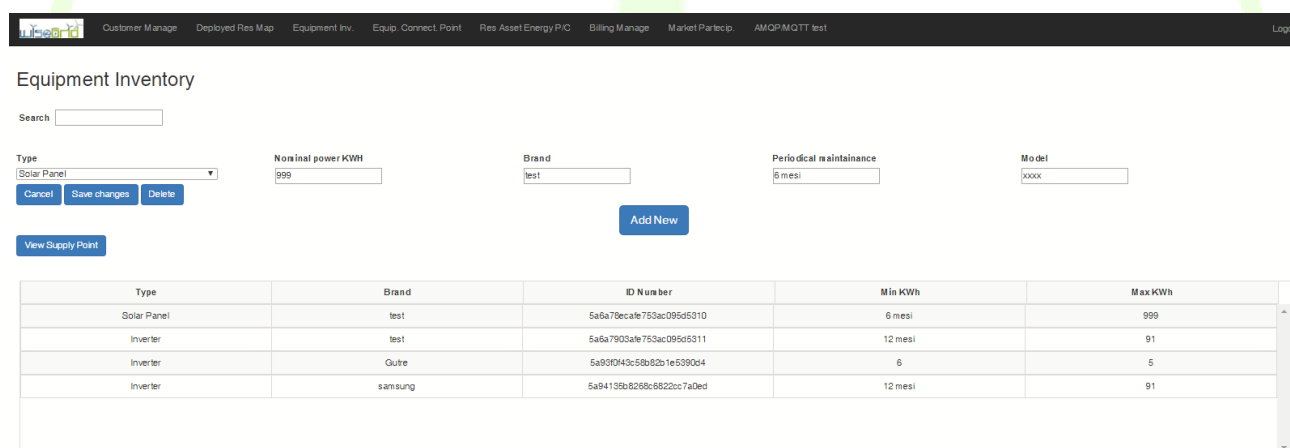
## 2.2.2 Asset Manager Module

### 2.2.2.1 Module Description

The asset manager module implements all the functionalities that allow the RESCO companies to manage all the asset to its disposal. To begin with the assets registration to the system to continue with its maintenance status during the operational time into a supply point identified by a specific contract typology.

The Asset Manager module functionalities are identified by different functionalities of the RESCO tool as follow:

- **Equipment inventory:** by this functionality the RESCO operator can add a new Equipment to the system by filling several attribute fields like the typology of the equipment (e.g. Photovoltaic Panel – Wind turbine – CHP), the asset brand, the asset model and so on with other relevant attribute. This equipment can be modified by the operator, deleted (if not linked to a contract yet) and searched by a specific functionalities able to filter the data present into the summary list without specify the column to filter (the system automatically search the word to search between the available one into the columns values).



| Type        | Brand   | ID Number                | Min KWh | Max KWh |
|-------------|---------|--------------------------|---------|---------|
| Solar Panel | test    | 5a8a78ecafe753ac095d5310 | 6 mesi  | 999     |
| Inverter    | test    | 5a8a7903afe753ac095d5311 | 12 mesi | 91      |
| Inverter    | Gutre   | 5a93f043c58b82b7e539004  | 6       | 5       |
| Inverter    | samsung | 5a94198b6268c6822cc7a0ed | 12 mesi | 91      |

Figure 2 – Equipment inventory

- **Equipment connection point:** this functionality allow the RESCO to immediately have a view about the supply points, its basic information and the equipment installed. By this functionality it is also possible to know the status of each connection point by a dedicated column in the Supply point summary list, which represents it by using three kind of “smile icons” accordingly to the equipment status – representing whether equipment is ok, or presents a problem which is or not under verification. From this functionalities the RESCO user by clicking to the “smile icon” can also view the detail of the maintenance work provided by the Maintenance module (see next section) in a dedicated new web page.

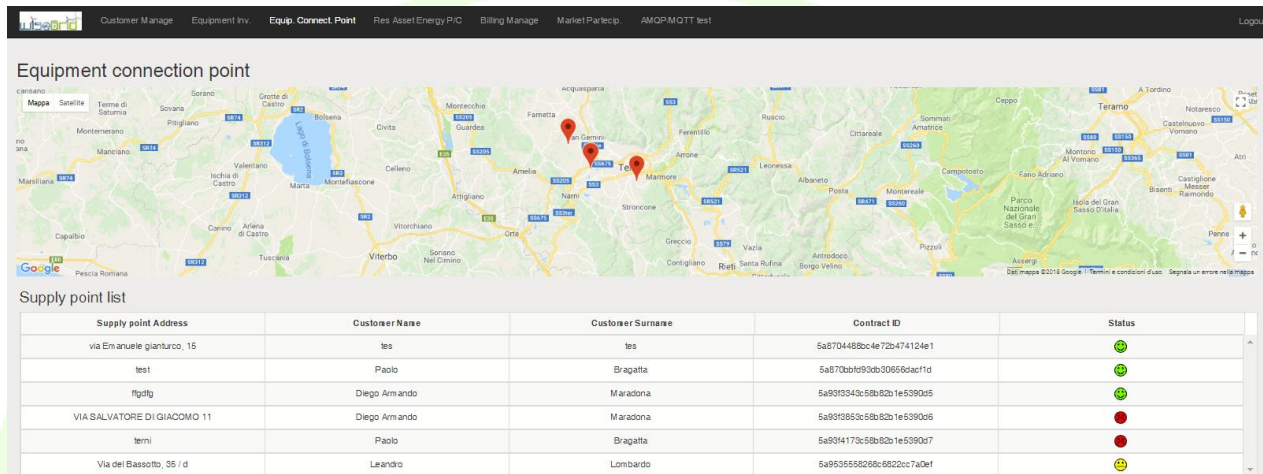


Figure 3 – Equipment connection point

### 2.2.2.2 Module Data Model

The following data model is related to the assets that could be installed into a RESCO connection point.

Not all the values listed in the data model are used in the equipment management function of the RESCO tool. However, they are also defined here because the data model of the assets is shared with other WiseGRID project tools that may need them.

Table 1 – Assets data model

| Field                      | Type   | Description  |
|----------------------------|--------|--|
| Eq_type                    | String | The type of the Equipment (An enumeration of available equipment like <ul style="list-style-type: none"> <li>• Battery</li> <li>• PV (inverter)</li> <li>• Wind turbine</li> <li>• CHP</li> </ul> .....) |
| Eq_system_ID               |        | Automatically calculated. For Batteries it should be BAT1, BAT2, BAT3,..., for PV it should be PV1, PV2, PV3   |
| Eq_brand                   | String | The equipment brand  |
| Eq_model                   | String | The equipment model  |
| Eq_maximum_charge_power    | Number | The equipment maximum charge power in kW   |
| Eq_maximum_discharge_power | Number | The equipment maximum discharge power in kW  |

|                       |        |  |
|-----------------------|--------|--|
|                       |        |  |
| Eq_nominal_capacity   | Number | The nominal capacity of the battery  |
| Eq_usable_capacity    | Number | The usable capacity of the battery .....   |
| Eq_usable_cycles      | Number | Guaranteed cycles of the battery   |
| Eq_pool_assignment    | String | Pool 1, Pool 2, Pool 3   |
| Eq_allocated_capacity | Number | Allocated capacity for pool  |
| Eq_allocated_power    | Number | Allocated power for pool   |
| Eq_preferred_mode_1   | String | <p>This field allow two following choices:</p> <p>Self-consumption</p> <p>Peak shaving</p> <p>Time of use management</p> <p>Frequency regulation</p> <p>Voltage regulation</p> <p>Backup</p> |
| Eq_preferred_mode_1   | String | <p>This field allow two following choices:</p> <p>Self-consumption</p> <p>Peak shaving</p> <p>Time of use management</p> <p>Frequency regulation</p> <p>Voltage regulation</p> <p>Backup</p> |
| Eq_voltage            | String | <p>The output voltage of the equipment</p> <ul style="list-style-type: none"> <li>- 230 V AC mono phase</li> <li>- 380 V AC three phase</li> <li>- 10 kV AC</li> <li>- 20 kV AC</li> </ul>   |
| Eq_Number_Strings     | Number | This field should contain the number of strings and related inverters of the equipment.  |
| Eq_Remote_Control     | String | This field should give information if the equipment can be disconnected by a remote operator in case of contingency (e.g. over-frequency). This field can be yes/no                          |
| Eq_Maximum_Power      | Number | Maximum power that can be provided by the equipment  |
| Eq_Nominal_Power      | Number | Nominal power that can be provided by the equipment  |

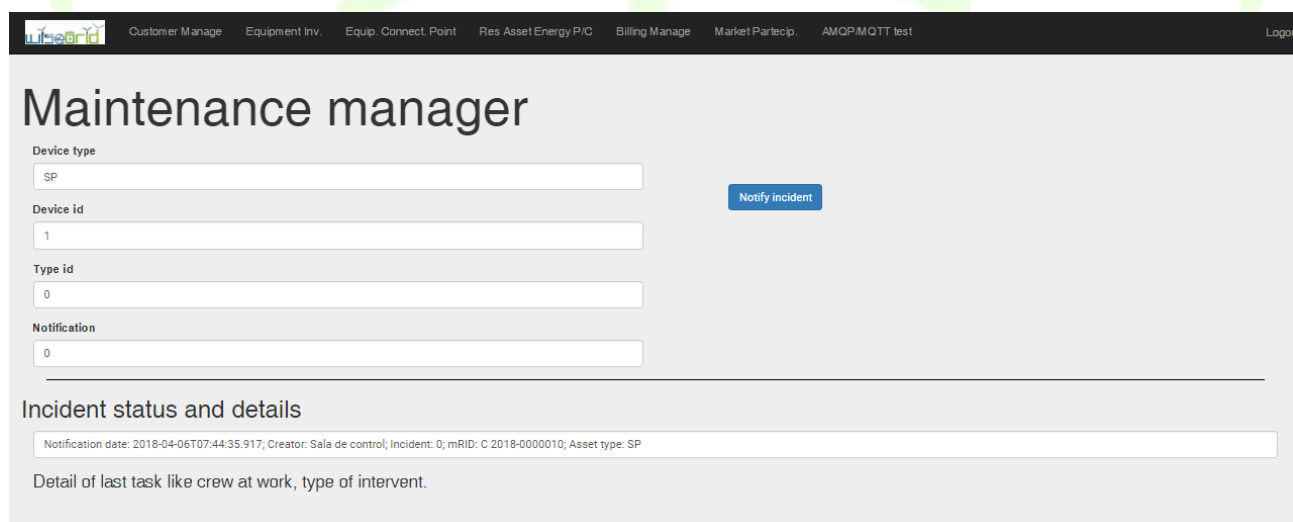
## 2.2.3 Maintenance Management Module

### 2.2.3.1 Module Description

The maintenance manager is an independent module focused on integral maintenance. The module can be hosted in the cloud (SaaS), and can be interfaced using a REST web service. It also offers web-based access to show the managed information and perform actions, and an Android App for the crew. Its versatility allows it to be integrated with different applications of the WiseGRID ecosystem, enabling an efficient system for maintenance management.

The RESCO tool use some of the maintenance module functionalities exposed by rest API inside a specific user interface as in the next figure. To use the functionalities offered by the maintenance manager module a link between this module and the RESCO tool is needed. This link is represented by a common registration of the assets and its location in both RESCO and maintenance manager module data base. This link is created by sending information related RESCO asset and its location during the creation of a contract in which all these information are presents (a specific maintenance manager API is called that return a common ID for both tools). At this point it is possible to send maintenance request from the RESCO maintenance functionalities to the maintenance manager without specify nothing else that an ID that represents the common asset identification and the typology of issue. Currently, two events can trigger this notification:

- The first one, a preventive action based on the available information about the normal maintenance work of each asset linked to a specific contract (the calculation start by analysing the maintenance period indicated in phase of asset registration and the contract start date).
- The second one: corrective maintenance, triggered by the RESCO company by maintenance user interface in case of a customer signal or automatically when the system for example does not receive for a certain time metering data from the supply point (an agent is listening to the RT monitor system an check if data are correctly stored into the long-Term Data Base). When the incident is notified to the Maintenance Manager Module the RESCO tool maintenance user interface show the incident status and related details to the RESCO company by invoking the specific API offered by the Maintenance Manager Module that are explained into the REST API (Table 2 )



The screenshot shows the 'Maintenance manager' web interface. At the top is a navigation bar with links: Customer Manage, Equipment Inv., Equip. Connect. Point, Res Asset Energy P/C, Billing Manage, Market Particip., AMQP/MQTT test, and a Login button. The main content area has a title 'Maintenance manager' and a form with the following fields:

- Device type: SP
- Device id: 1
- Type id: 0
- Notification: 0

There is a 'Notify incident' button next to the Device id field. Below the form, there is a section titled 'Incident status and details' which contains a text box with the following information: 'Notification date: 2018-04-06T07:44:35.917; Creator: Sala de control; Incident: 0; mRID: C 2018-0000010; Asset type: SP'. Below this text box is a label 'Detail of last task like crew at work, type of intervent.'

Figure 4 – Maintenance manager

#### 2.2.3.1.1 Maintenance Management Systems

An effective way for an organization to manage the day to day events and risks is by employing a

**Maintenance Management System.** A maintenance management system is an ICT system with the purpose of automating the registration and information distribution processes relative to system incidents, it includes the following functionalities:

- Human resources
- Asset inventory
- Preventive maintenance planning and management
- Corrective maintenance management

In addition, it has to include different ways to register incidents or faults occurring in the system, these may be automatically reported by control systems or manually notified by clients or operators.

The objective of these systems is to manage the maintenance in an ordered way, handling incidents through its whole life, and allowing the organization to keep track and analyse them, answering questions such as:

- When has the incident occurred?
- Who reported it?
- Who has been assigned to solve it?
- Which materials have been used to solve it?
- How long did it take?

#### **2.2.3.1.2 Structure of the maintenance management module**

The structure of a Maintenance Management System is very coupled to the structure of the organization (see Figure 5). Following a bottom-up approach, the following roles (and the corresponding modules of the Maintenance Management System) can be usually found in those systems:

- Roadside service management
  - Maintenance brigades: field-site technicians dealing with the preventive and corrective actions. They usually use a specific handheld application for receiving work orders (details on tasks they have to perform) and notifying the evolution of their work on these tasks and the materials employed to solve them
  - Crew Foremen: technicians in charge of a maintenance brigade. They supervise the work of the brigade. They usually use a specific handheld app for monitoring the work of the brigade, communicate with them and shift, reorder or reassign work orders for practical reasons
  - Maintenance manager: technician responsible of the whole roadside service management. Monitors overall activity of the brigades and is able to shift, reorder or reassign packages of related work orders among the different brigades
- Integral management
  - Operators: they control the details of the maintenance management system from a control facility. These tasks include registration of new incidents in the system (for instance, when incident is reported via phone call), planning of preventive tasks, management of inventory, management of vehicle fleet, etc. Depending on the size of the organization, operators are usually assigned to control a certain area or a certain type of operations (e.g. operators can be split in separate teams to manage the energy and gas systems in a multi-utility company)
  - Head of service: the person in charge of the operators, with access to an overview of the whole system
  - Executive manager: the system also provides executive reports of the different aspects managed by the system (i.e. amount of incidents, distribution of the incidents over the different areas of the organization, costs associated to the maintenance operations, etc.)

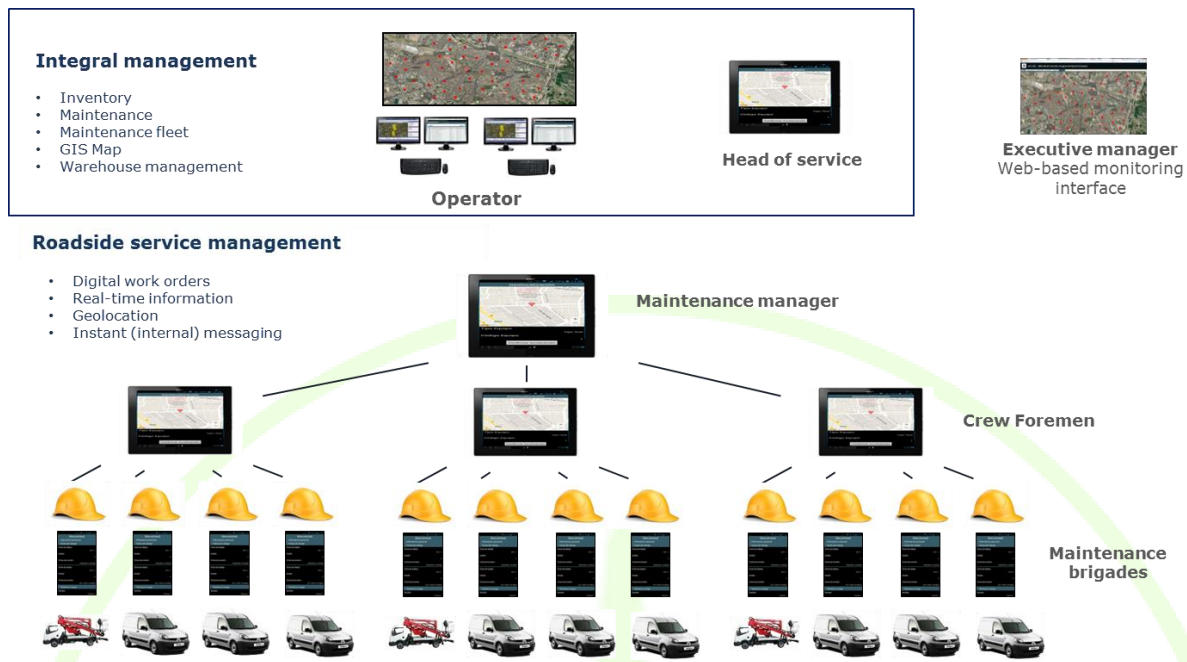


Figure 5 – Schematic illustration of a Maintenance Management System

## 2.2.3.2 Module Data model

### 2.2.3.2.1 Incidents

#### Type

An important issue of Maintenance Management Systems is the kind of incidents able to handle. These incidents or events typically are classified into two groups:

- **Corrective events:** those that refer to an active problem or fault. Given a specific fault, the organization usually has to solve it within a certain amount of time (agreed with the client)
- **Preventive events:** those that are taken in order to prevent future problems. These actions are usually defined by the organization, based on their know-how, previous experiences or legal requirements (e.g. checking the components of certain devices). These events are usually planned on a periodic basis.

#### Lifecycle

An incident goes through 4 different status during its lifecycle

- **Notified (N):** the maintenance manager has registered the incident, but it has not been attended yet
- **Registered (R):** the maintenance operator confirms the reception of the incident, but does not perform any action yet
- **Initiated (I):** the maintenance operator is working on the incident
- **Finished (F):** the maintenance operator has finished working on the incident

### 2.2.3.2.2 Tasks

A task is any action performed on an incident. These actions reflect evolution of the status, or any information provided by the crew working on the incident (free text or photographs).

### 2.2.3.2.3 Crew



The maintenance manager holds a list of personnel that can be assigned to work in the incidents. Each incident can be assigned to a single worker.

#### 2.2.3.2.4 Internal modules

The maintenance management module internally implements the following modules:

- Maintenance database: stores all data and handles the lifecycle of the incidents
- Maintenance management UI: web interface where all functionality of the module is exposed
- Android APP: allows personnel assigned to maintenance to interact with the maintenance management module, check list of assigned incidents and update the status of those
- REST API: exposes some of the functionalities of the maintenance manager module to allow integration of third party applications

#### Web interface

The web interface is the main working interface for the personnel coordinating the maintenance in the DSO. From this interface, a general overview of the status of all assets, open incidents, planning of maintenance and evolution of the works is presented. In addition, reports can be generated to gain insight into the efficiency of the works, the times usually required to solve certain type of problems, or the distribution of the incidents according to its type, for instance.

Corrective > Incidents Query

FiltersActionsReports

Filter☐Filter

State

F I N R Imp

| Show                     | PT             | Equipment | Name   | NP | Notified Incident | Invert | Notification Date | Area        | Imp. | Pri. | Remaining |
|--------------------------|----------------|-----------|--|----|-------------------|--------|-------------------|-------------|------|------|-----------|
| <input type="checkbox"/> | C 2018-0000001 | BAT0001   | Battery 0001 (Ampere)                                    | E  | NOCOMM            | F      | 4/3/2018 12:59    | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000002 | BAT0001   | Battery 0001 (Ampere)                                    | S  | NOCOMM            | N      | 4/3/2018 13:55    | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000003 | BATJ2     | Test   | S  | NOCOMM            | N      | 4/3/2018 14:37    | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000004 | BATJ4     | Test Battery   | S  | NOCOMM            | N      | 4/3/2018 16:42    | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000005 | BATJ3     | Test2  | S  | NOCOMM            | N      | 4/5/2018 10:54    | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000006 | BATJ5     | description  | S  | NOCOMM            | N      | 4/5/2018 11:05    | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000007 | BATJ10    | Description of type BAT                                  | S  | NOCOMM            | N      | 4/5/2018 11:08    | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000008 | BATJ11    | Description of type BAT                                  | S  | NOCOMM            | N      | 4/5/2018 11:09    | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000009 | SPJ4      | Test Battery2  | S  | 0                 | N      | 4/5/2018 14:24    | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000010 | SPJ1      | Test Battery   | S  | 0                 | N      | 4/6/2018 09:44    | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000011 | SPJ3      | Test Battery2  | S  | 0                 | N      | 4/12/2018 05:53   | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000012 | BATJ12    | Description of type BAT                                  | S  | NOCOMM            | N      | 4/12/2018 16:17   | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000013 | SPJ33     | Supply point name Leandro surname Lombardo idSmx BBB5979 | S  | 0                 | N      | 4/13/2018 10:58   | RESCO TERNI | 0    | ---  | ---       |
| <input type="checkbox"/> | C 2018-0000014 | SPJ34     | Supply point name Leandro surname Lombardo idSmx bbb9579 | S  | 0                 | N      | 4/13/2018 13:31   | RESCO TERNI | 0    | ---  | ---       |

1

>

<

14 items listed

Page 1 from 1

Page 1 from 1

14 items listed

Figure 6 – Screenshot of the Maintenance Module Web Interface

#### Android app

The Android App is the main working interface for the field personnel in charge of working in the incidences. This application is designed as a tool that allows maintenance technicians to have simply updated information in real time in relation to their work and communicate the progress of the work done to the rest of the organization, facilitating the exchange of information among the different agents of the system during their working day.

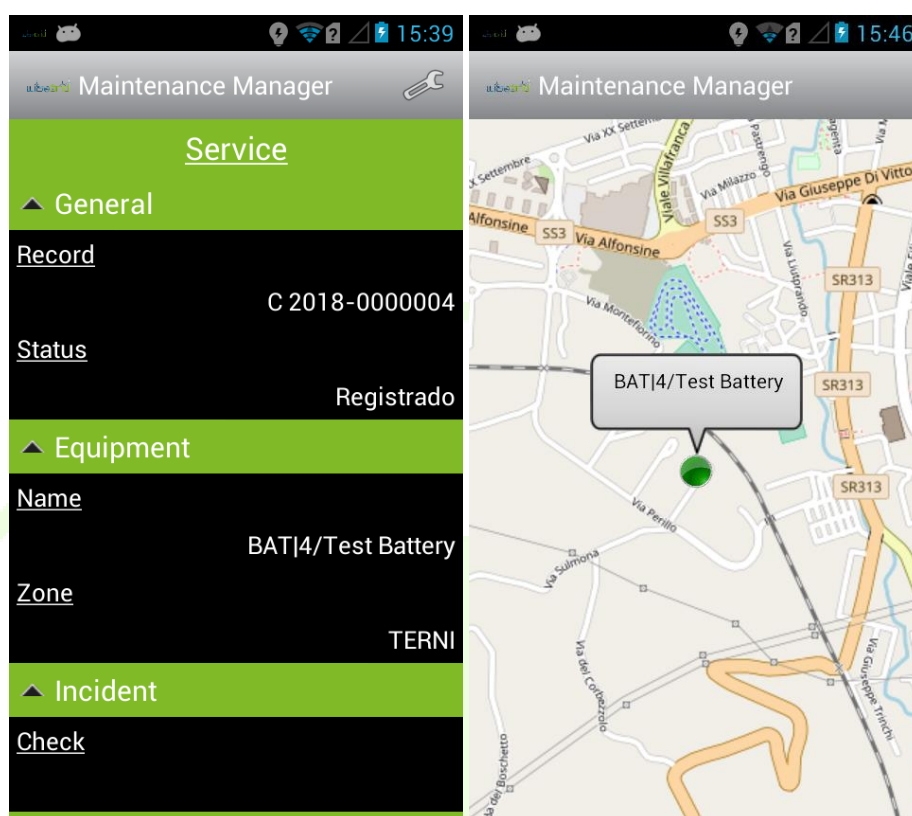


Figure 7 – Screenshot of the Maintenance Module Android App

## REST API

The REST API is documented using swagger. A summary is presented hereby.

Table 2 – Maintenance module

| REST API METHOD | PATH                     | INPUT                 | OUTPUT   |
|-----------------|--------------------------|-----------------------|--|
| GET             | /api/incident            |                       | List registered incidents                        |
| POST            | /api/incident            | New incident details  | Details of new registered incident               |
| GET             | /api/incident/{id}       | Id of incident        | Details of incident                              |
| GET             | /api/incident/{id}/tasks | Id of incident        | List of registered tasks related to the incident |
| GET             | /api/crew                |                       | List of workers                                  |
| GET             | /api/crew/{id}           | Id of worker          | Details of workers                               |
| GET             | /api/crew/{id}/incidents | Id of worker          | List of incidents assigned to worker             |
| GET             | /api/crew/{id}/tasks     | Id of worker          | List of tasks performed by worker                |
| GET             | /api/device              |                       | List of registered devices                       |
| POST            | /api/device              | Device details        | Details of new registered device                 |
| GET             | /api/device/{type}       | Type of device        | List of registered devices for the given type    |
| GET             | /api/device/{type}       | Type and id of device | Details of device                                |



|     |                  |  |                                 |
|-----|------------------|--|---------------------------------|
|     | }/{id}           |  |                                 |
| GET | /api/devicetypes |  | List of registered device types |

### 2.2.3.3 Workflow

This section describes a possible workflow for the integration of the maintenance management module within a system:

**Table 3 – Workflow overview of the maintenance management module**

| ORDER | ACTION                            | Interface   | Description   |
|-------|-----------------------------------|---|---|
| 1     | Register asset types              | Maintenance Management UI                         | The organization registers the kind of devices they maintain (e.g. PV panels, batteries)  |
| 2     | Register incident types per asset | Maintenance Management UI                         | Incidents in the maintenance management module are typified. Each asset type may be only assigned certain types of incidents  |
| 3     | Register personnel information    | Maintenance Management UI                         | The organization registers the personnel assigned to maintenance. An account for the smartphone app is created for each individual upon registration  |
| 4     | Register devices                  | Maintenance Management UI / REST API              | Actual instances of the devices under maintenance are registered in the system. An instance is identified by its asset type and its numeric id  |
| 5     | Register new incidents            | Maintenance Management UI / REST API / Mobile App | Upon detection of a problem with an asset, a new incident can be registered within the maintenance management module  |
| 6     | Assign personnel to an incident   | Maintenance Management UI                         | The maintenance manager may assign personnel to deal with the incident  |
| 7     | Work on incident                  | Maintenance Management UI / Mobile App            | Personnel assigned to the incident receives the information and performs updates on the status from the mobile app. Similar work can be performed by the maintenance manager using the web UI |
| 8     | Close incident                    | Maintenance Management UI / Mobile App            | Personnel assigned to the incident can close the incident from the mobile app when work is finished. Similar work can be performed by the maintenance manager using the web UI                |
| 9     | Monitor incidents and personnel   | Maintenance Management UI / REST API              | Maintenance manager may monitor the status of the personnel and the incidents using the web UI. Some functionalities for monitoring are also exposed to the REST API                          |

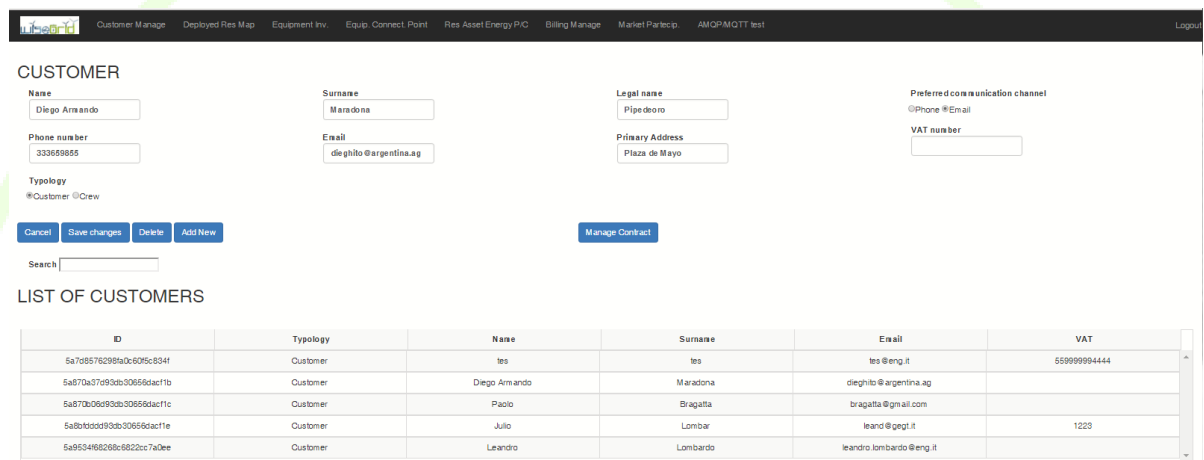
A swagger specification is available in the ANNEX A paragraph 8.1.

## 2.2.4 Contract Manager Module

### 2.2.4.1 Module Description

The most basic process of the RESCO organisation is handling the portfolio of customers. This mainly implies a customer management and related contract management. The Contract Manager Module allows the RESCO to manage these aspects by implementing some specific functionalities as described below.

- **Customer management:** By this functionalities the RESCO company user can register a new customer to the system, manage it and cease it. The typical attributes that identify a customer into the RESCO tool are name, surname, address, birth date, preferred communication channel, vat number and email. Each of these field have a specific role into the tool like to know the address to send a bill or a written communication, the email address to send communication and other information like contract variation etc. and so on with the others information.



**CUSTOMER**

Name:  Surname:  Legal name:  Preferred communication channel: ☐ Phone ☐ Email

Phone number:  Email:  Primary Address:  VAT number:

Typology: ☒ Customer ☐ Crew

Search:

**LIST OF CUSTOMERS**

| ID                       | Typology | Name          | Surname  | Email                   | VAT          |
|--------------------------|----------|---------------|----------|-------------------------|--------------|
| 5a70b576299a0c605c834f   | Customer | Ies           | Ies      | ies@eng.it              | 559999994444 |
| 5a870a37d99db30656dac1fb | Customer | Diego Armando | Maradona | dieghito@argentina.ag   |              |
| 5a870b06d99db30656dac1fc | Customer | Piaolo        | Braglia  | braglia@gmail.com       |              |
| 5a80b0d999db30656dac1fe  | Customer | Julio         | Lombar   | leand@pegi.it           | 1223         |
| 5a953488268c8822cc7aDee  | Customer | Leandro       | Lombardo | leandro.lombardo@eng.it |              |

Figure 8 – Customer manager

- **Manage Contract:** By this functionalities the RESCO company user can register, manage a contract between the company and a specific customer present at system. A customer may have several contracts, each of them linked to a particular dwelling (supply point). Contracts may include information such as contract start and end date, contract supply point location (address, latitude, longitude), references of installed assets and contract type. This functionalities is reachable by a specific button in the “Customer manage” functionalities. The RESCO company user can define three different typology of contract, an important remark is that the type of contract has implications on the point where the equipment shall be installed, and where the energy measurements need to be taken and to calculate the bill. The following definition better explain these contract typology.

**Contract Type 1:** under this contract type, the RESCO trades the whole production of the assets installed in customer premises. It is therefore necessary to measure this production directly from the connection point of the assets – i.e. measurements cannot be affected anyway by the customer’s consumption. In order to cover this assumption and simplify the billing process, it would be necessary that the RESCO contracts a different supply point with the DSO, where all the assets and the corresponding smart meters are connected.

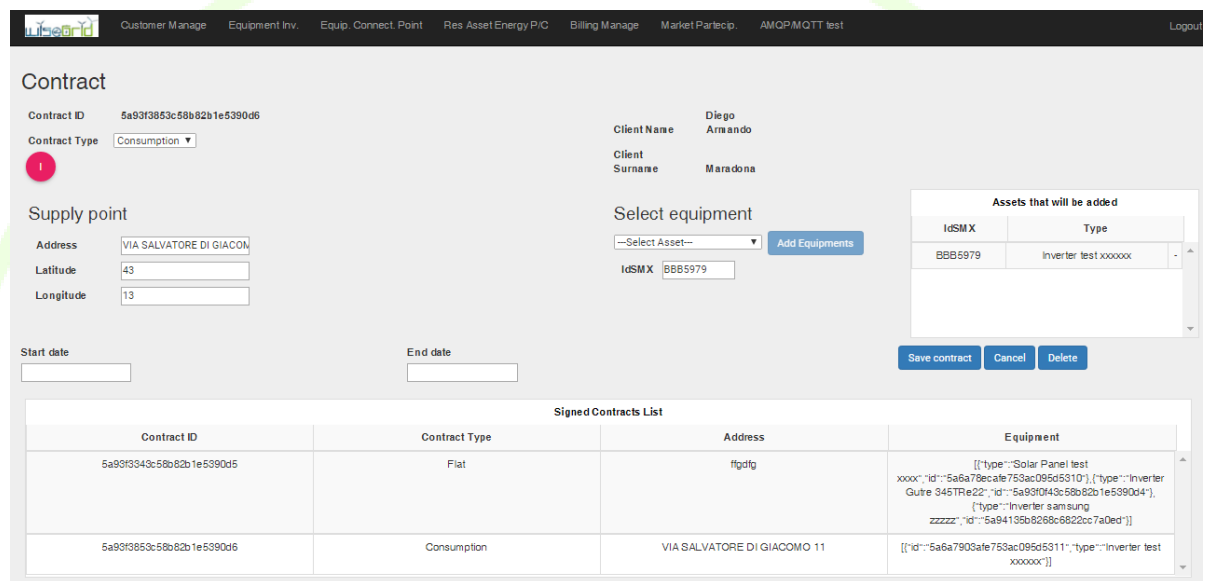
**Contract Type 2:** under this contract type, the customer is allowed to self-consume energy and the RESCO trades the production surplus. Two kind of measurements are therefore needed:

- Production surplus: can be obtained directly from the information provided by the smart meter located at the supply point of the customer. This information is usually retrieved by

the DSO, and provided to the customer. Any energy produced at the supply point is property of the RESCO.

- Self-consumed energy: if the contract stipulates a price for the self-consumed energy, it will be necessary to install an additional smart meter to retrieve the whole production data. Self-consumption can be measured by comparing this data with the metering information at the supply point.

**Contract type 3:** under this scenario, the contract basically includes a fee for the renting of the assets, and no metering information is needed.



The screenshot shows the 'Contract' management interface. It includes fields for Contract ID, Client Name, Client Surname, Supply point Address, Latitude, Longitude, Start date, and End date. There is a 'Select equipment' section with a dropdown and an 'Add Equipments' button. A table titled 'Assets that will be added' shows equipment details. Below the form is a 'Signed Contracts List' table.

| Contract ID              | Contract Type | Address                     | Equipment   |
|--------------------------|---------------|-----------------------------|---|
| 5a93f3853c58b82b1e5390d5 | Flat          | flgdlg                      | [[{"type":"Solar Panel test xxxxx","id":"5a6a78ecafe753ac095d5310"}, {"type":"Inverter Gutre 345TRe22","id":"5a93f043c58b82b1e5390d4"}, {"type":"Inverter sam sung zzzzz","id":"5a94135b8268c6822c7a0ed"}]] |
| 5a93f3853c58b82b1e5390d6 | Consumption   | VIA SALVATORE DI GIACOMO 11 | [[{"id":"5a6a7903afe753ac095d5311","type":"Inverter test xxxxxx"}]]   |

Figure 9 – Contract manager

## 2.2.4.2 Module Data Model

In this paragraph the “customer” and “contract typology” as well as the interaction between “customer” and “Contract” data models are presented.

### 2.2.4.2.1 CUSTOMER

Such collection contains all the information of a RESCO customer. In addition to personal information like Name, Surname and Birthdate, there are also information relating to VAT number, email address and telephone number and on the preferred channel of communication to allow the system to send communications, bills and so on to individual customers by ordinary mail, e-mail or via phone.

Table 4 – Customer data model

| Field   | Type   | Description      |
|---------|--------|------------------|
| Name    | String | The user name    |
| Surname | String | The user surname |

|                                  |        |   |
|----------------------------------|--------|---|
| Birthdate                        | String | Birthdate of the customer for identification  |
| legal_name                       | String | The user/company legal name   |
| Customer_ID                      | String | Automatically calculated  |
| phone_number                     | String | The principal user phone number   |
| Email                            | String | The customer email (Could be used to identify the customer into other tools eg. WISE-HOME)                |
| Vat_number                       | String | The user VAT number (Could be used to identify the customer into other tools eg. WISE-CORP)               |
| Primary_address                  | Object | <pre>{ "country": String,   "city": String,   "address": String }</pre> The address to legal notification |
| pre-ferred_communication_channel | String | This field allows two choice <ul style="list-style-type: none"> <li>• Phone</li> <li>• Email</li> </ul>   |

#### 2.2.4.2.2 CONTRACT TYPOLOGY.

Such collection contains only the definition of the available contract typology that could be signed by RESCO manager and final Customer, currently three typology are available as defined into the module description section.

1. RESCO pays a fee to end-users using their premises (e.g. for installing PVs on their roof), installs and maintains the RES assets and markets all produced energy;
2. RESCO provides to customers the supply of energy coming from RES owned by the RESCO (i.e. allowing self-consumption) and markets the production surplus;
3. RESCO provides to customers the installation of RES equipment (e.g. PV panels) which are owned and maintained by the RESCO but fully exploited by the end customers (renting business model).

**Table 5 – Contract typology data model**

| Field       | Type   | Description   |
|-------------|--------|---|
| Type        | String | The contract type (Currently 3 kind of contract are identified) |
| Description | String | The contract description  |

#### 2.2.4.2.3 CUSTOMER\_CONTRACT

Such collection contains all the information of a RESCO contract signed between a RESCO and a RESCO customer. All the information presents into a contract are fundamental to the whole WG RESCO tool. For example the contract type is needed to the billing module to calculate the customer bill, the start contract date is needed for the maintenance module to triggers the date of the “Planned maintenance” and also the asset linked to the contract to give information about the assets to maintain. The supply point location identify a contract in a specific place and give information to the maintenance crew about the site where an

operation is to be carried out, the same information is needed to the Forecasting service to ask Weather forecast needed to calculate the Energy forecast.

**Table 6 – Customer contract data model**

| Field                    | Type                | Description   |
|--------------------------|---------------------|---|
| Contract_id              | String              | The id of the contract (Automatically calculated, sequence)   |
| Contract_type            | String              | The contract type (The same of the contract_definition model)   |
| customer_id              | String              | The id of the customer linked to the contract   |
| contract_start_date      | Date                | The start date of the contract  |
| Contract_end_date        | Date                | The end date of the contract  |
| Supply_point_location    | Object              | <p>Eg: (GeoJson)</p> <pre>{   "type": "Feature",   "geometry": {     "type": "Point",     "coordinates": [125.6, 10.1]   },   "properties": {     "name": "Dinagat Islands"   } }</pre> <p>The address of the supply point for the specific contract</p> <p>The supply point latitude</p> <p>The supply point longitude</p>                     |
| asset_linked_to_contract | Array[Ref -> Asset] | <p>A list of equipment linked to the contract (After the element Equipment_type and Equipment_id)</p> <pre>[   {     "type":String,    (From the Asset definition)     "id" : String    (MeterID (mRID) of the real asset installed)     "installation_date" : Date (The installation date to determine the periodical maintenance)   } ]</pre> |

## 2.2.5 Billing Manager Module

### 2.2.5.1 Module Description

The billing module allows the RESCO to be able to easily determine, based on the type of contract with the final customer the relevant bill to be charged to it. As described into the Customer contract module section the relationship between a RESCO and a Customer can be regulated by three different types of contract, the first is that the RESCO pays a price for the rental of the area where it installs its assets and trades the produced energy, the second one the customer is allowed to self-consume energy and the RESCO trades the production surplus and the last one the RESCO rents and maintains a RES installation directly at the customer's location and asks for payment in return., the amount to be billed in terms of the amount to be collected for the energy consumed and to be paid for the energy produced.

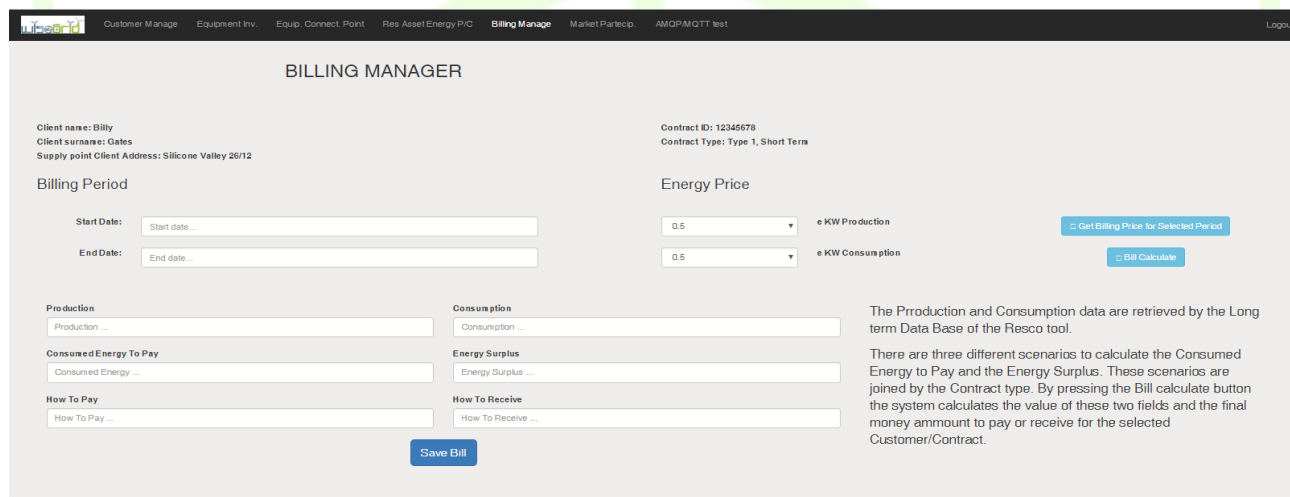
The Billing Manager Module solve these functionalities by implementing this function:

The billing module is responsible for determining the correct invoice with the customer based on the scenarios described above. As explained above, the first and third scenarios provide for a fixed price established in the contract which did not require any calculation, for the second scenario, some calculations are needed and are explained below in the 2.2.5.2.2 Module Algorithmic Framework .

The Billing manager functionality is available by the corresponding Billing Manage menu or by the Contract manager functionalities by a specific button. The function have a static part in which information about the Customer and the contract that has been signed with the company are displayed and a dynamic part in which the RESCO can choice the billing period, the price of the energy (divided into price for energy produced and price for energy consumed) both manually based on contractual agreements or dynamically by acquiring the energy price by a “Tariff provider” services available by WG IOP.

When these first information are filled into the respectively fields, the functionalities is ready to calculate the bill by pressing the “Bill calculate” button (see next 2.2.5.2.2 Module Algorithmic Framework). Information about the energy production and consumption and the amount to pay or to receive to or from the customer is showed in the specific fields of the functionalities user interface.

By clicking on the “Save bill” button, the bill is stored into the long term DB for future analysis and in the meantime a message with bill information is sent to the WiseCORP and WiseHOME tools via WG IOP for visualization of the end-users.



The screenshot shows the 'BILLING MANAGER' web interface. At the top, there is a navigation bar with links: Customer Manage, Equipment Inv., Equip. Connect. Point, Res. Asset Energy P/C, **Billing Manager**, Market Particip., and AMQP/MQTT test. The main content area is titled 'BILLING MANAGER' and contains the following sections:

- Client Information:** Client name: Billy, Client surname: Gates, Supply point Client Address: Silicone Valley 26/12. Contract ID: 12345678, Contract Type: Type 1, Short Term.
- Billing Period:** Start Date: [text input], End Date: [text input].
- Energy Price:** e KW Production: 0.5, e KW Consumption: 0.5. Buttons: 'Get Billing Price for Selected Period' and 'Bill Calculate'.
- Production:** Production ... [text input].
- Consumption:** Consumption ... [text input].
- Consumed Energy To Pay:** Consumed Energy ... [text input].
- Energy Surplus:** Energy Surplus ... [text input].
- How To Pay:** How To Pay ... [text input].
- How To Receive:** How To Receive ... [text input].
- Buttons:** 'Save Bill' and 'Bill Calculate'.

On the right side, there is explanatory text: 'The Production and Consumption data are retrieved by the Long term Data Base of the Resco tool.' and 'There are three different scenarios to calculate the Consumed Energy to Pay and the Energy Surplus. These scenarios are joined by the Contract type. By pressing the Bill calculate button the system calculates the value of these two fields and the final money ammount to pay or receive for the selected Customer/Contract.'

Figure 10 – Billing manager

## 2.2.5.2 Module Data Model and algorithmic framework

### 2.2.5.2.1 BILLING

Such collection contains the billing data of a RESCO supply point.

**Table 7 – Billing data model**

| Field        | Type      | Description  |
|--------------|-----------|--|
| Contract_id  | String    | The contract id that represents the supply point and the TYPE of contract to determine the billing   |
| Start_date   | Timestamp | The start date of the billing period   |
| End_date     | Timestamp | The end date of the billing period   |
| Energy_price | Number    | The price of a kWh of energy for a certain period of time  |
| Consumption  | Number    | The real consumption (depend by the contract type it could be the difference between the consumption – production.....)  |
| Production   | Number    | The real production of a supply point (depend by the contract if the production is > than the consumption it could be used to calculate the amount to pay to the customer) |

### 2.2.5.2.2 Module Algorithmic Framework

The billing calculation algorithm is based on the type of contract. At the time of writing for one of the contract type a specific calculation is performed, it is the case for the contract in which the customer must pay for the surplus of energy consumed respect the energy produced or receive a payment for the surplus of energy fed into the grid if it consumes less energy than it produces.

The input variable are:

1. Billing period (start date and end date)
2. Energy price (as contract agreements or by Tariff provider service)

The output variable are:

1. Energy production value
2. Energy consumption value
3. Consumed energy to pay
4. Energy surplus that can be trade
5. Price to pay in bill
6. Price to receive for energy surplus not consumed

The system reads the value of energy consumption and energy production for the selected period of time for the specific contract directly from the Long-term DB in which they are stored. Sum of each measured value over that time period, if the energy production sum is higher than energy consumption sum, the “Energy surplus” is valued with this difference, otherwise the "Consumed energy to Pay" field will be valued.

If the “Energy surplus” field is  $> 0$  then the algorithm calculates the price to be paid to the customer by multiplying this value by the producer price, otherwise, if “Consumed energy to pay” is  $> 0$ , the algorithm calculates the price of the bill by multiplying this value by the consumer price.

### 2.2.5.2.3 Module interaction with other tools via WG IOP

The billing manager functionalities is able to transmit information related the billing of a specific Supply point for a certain contract typology via WG IOP to other tools like WiseHOME and WiseCORP, where those will be presented to the customers. The functionalities collect into a specific queue named “resco\_billing” a message/payload that contains billing information. The billing information are sent via WG IOP every time a billing is calculated or every time a WiseGrid tool send a specific request for it. The AMQP protocol is the one adopted between the WG IOP available protocols for its versatility. The structure of the message is explained into the deliverable D4.2 at the WG RESCO tool message structure section.

## 2.2.6 Energy Metering Manager Module

The integration of the energy metering devices with the application is performed by using the Real-time monitoring module, the generic module developed within the project in order to read data flows from the WG IOP and integrate the data into the corresponding databases of the application.

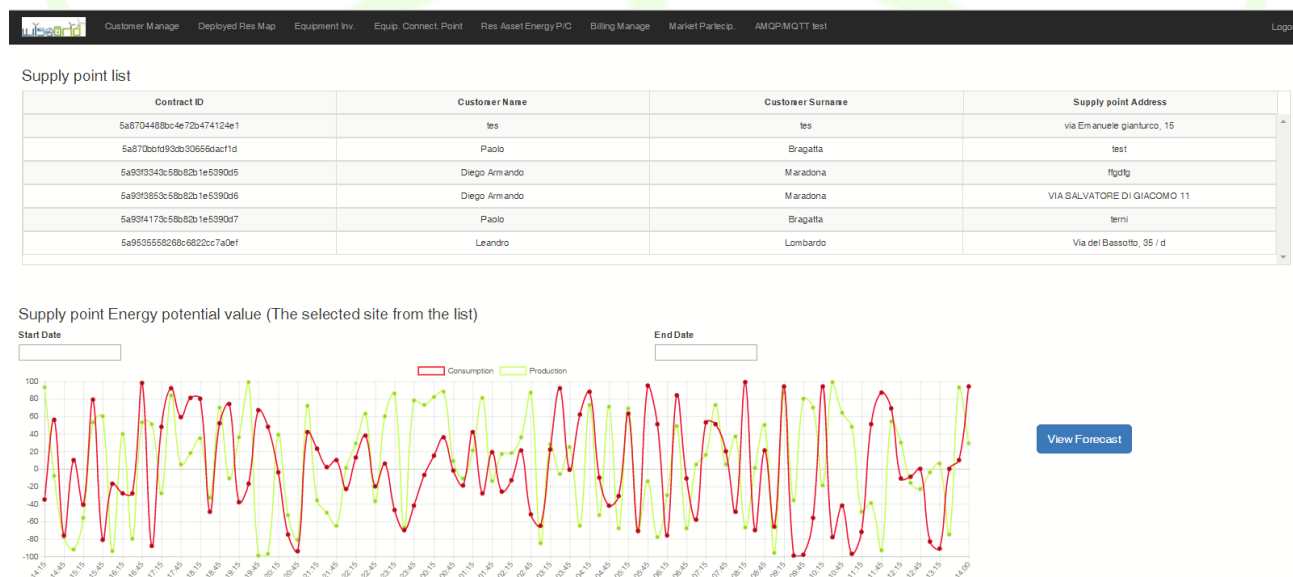


Figure 11 – Res Asset Energy P/C



The WiseGRID RESCO application reads data mainly from 2 different kind of sources.

### Smart meters

The main data flow, including the customers' demand and the RES assets production handled by the RESCO, is provided by smart meters. Within the project, two actual sources realizing this data flows are considered:

#### SMXs (Smart Meter eXtension) / SLAMs (Smart Low-cost Advanced Meter)

SMXs and SLAMs are a result of the H2020 project NobelGRID [3] that are also incorporated to the WiseGRID project. Both devices offer smart metering features with the capability to install third party applications in order to offer extra functionalities on top of the smart meter readings and interface with other devices (e.g. batteries, PV inverters, domestic assets...). In the WiseGRID project, those devices will be mainly used to obtain high rate energy readouts that are sent to the IOP to be processed by one or many applications

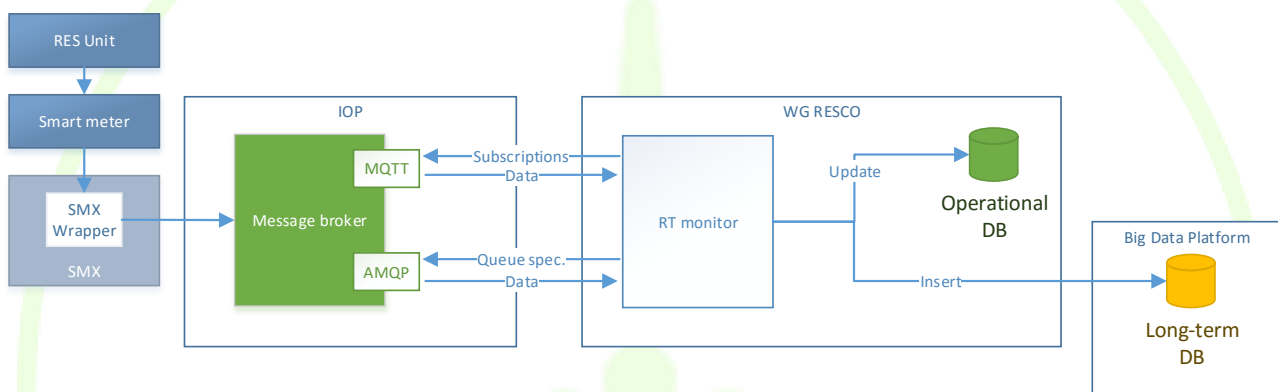


Figure 12 – RESCO tool Smart meter data flow from SMX to DB

SMX produces data in the form of DLMS over MQTT (OBIS codes). This format is convenient for transmission (small size), but is not comfortable to handle (data related to the same timestamp is sent using different messages).

DLMS to CIM translator performs three operations:

- Bundles all readouts referred to the same timestamp within a single message
- Formats the message to the CIM MeterReading common data model (encoded using JSON)
- Emits the result back to MQTT under a different topic

An example of a CIM MeterReading message is available in the ANNEX A paragraph 8.2.

### AMI (Advanced Metering Infrastructure) systems

Some of the pilot sites of the project have already deployed smart meters to a significant portion of their customers, which are being read by AMI systems. The project will therefore take advantage of the existence of such systems to demonstrate how the information about the energy demand and injection of the prosumers collected by those can be published to the WiseGRID IOP and incorporated to the ecosystem of applications. The approach is very similar to the one described for the data provided by Unbundled Smart Meter.

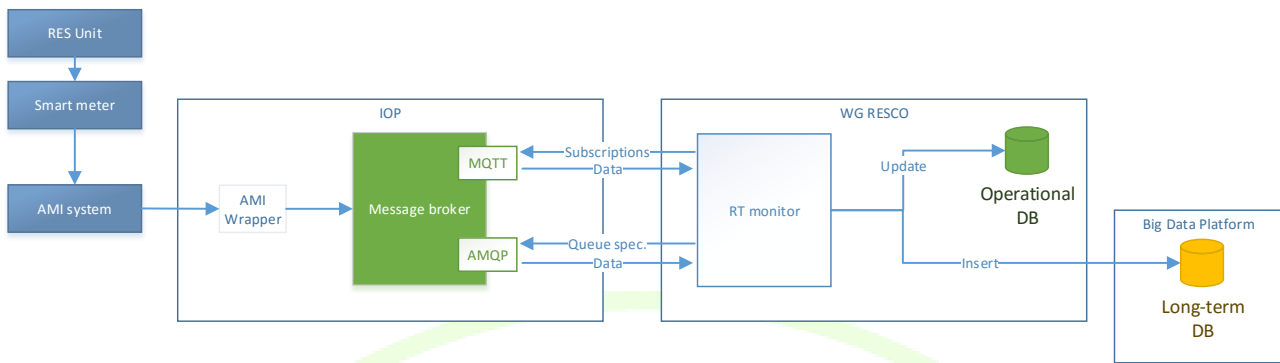


Figure 13 – RESCO tool AMI data flow from SMX to DB

## Batteries

The second main potential flow of data for the RESCO tool could come from domestic batteries. Within the WiseGRID project, batteries from VARTA and AMPERE ENERGY will be demonstrated in domestic environments. These batteries are capable to communicate directly with the WiseGRID IOP, offering their capabilities to the different applications of the WiseGRID ecosystem.

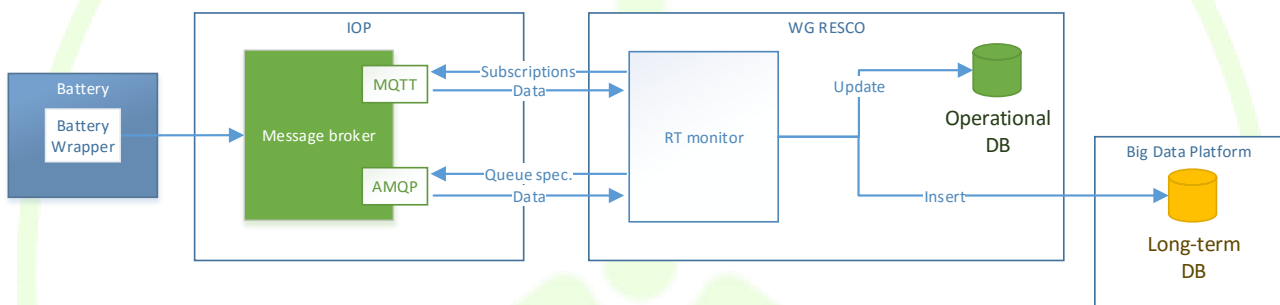


Figure 14 – RESCO tool battery data flow from SMX to DB

## Real-time monitor

As depicted in the previous diagrams, *RT monitor* is a generic module which implements the necessary logic for the data ingestion of the RESCO tool, by interfacing with the WG IOP Broker on one side and with the MongoDB databases on the other. Specifically, two main features are implemented within this module:

- It is able to subscribe to a MQTT topic on the WG IOP Broker and keep the current status of each device in a MongoDB database.
- It is able to track changes in a collection of MongoDB (operational) database, pushing a history of the changes into a different (long-term) database.

These two features are combined in order to fit the architecture of the RESCO tool, by feeding both the operational and the long-term (Big Data) databases of the application:

- Operational DB keeps one document per device (battery or smart meter data metering RES assets), stating last known status
- Long-term DB keeps a history of changes per device, each change hold by a document

## 2.2.6.1 Module Data Model and Algorithmic Framework

### BATTERY DATA MODEL

Such collection contains the measurement collected by the RT Monitor from the battery.

**Table 8 – Battery data model**

| Field                 | Type   | Description   |
|-----------------------|--------|---|
| id                    | String | The battery ID  |
| MeterActivePower      | String | At the grid connection point of the household as measured by the battery controller                                       |
| MeterReactivePower    | Number | At the grid connection point of the household as measured by the battery controller                                       |
| InverterActivePower   | Number | (AC) active power of the battery inverter   |
| InverterReactivePower | Number | (RP) reactive power of the battery inverter   |
| InverterPVPower       | Number | of the PV inverter included in a battery system, if it is such a “hybrid” system  |
| ExternalPV            | Number | External power of a stand-alone PV inverter, which is somehow measured by or in communication with the battery controller |
| InverterBatteryPower  | Number | DC power of the battery inverter  |
| BatterySOC            | Number | Usable energy presently in the battery divided by actually usable energy capacity   |
| BatterySOH            | Number | Actually usable energy capacity divided by rated capacity   |
| BatteryVoltage        | Number | The voltage of the battery  |
| MeterGridVoltage      | Number | Meter voltage at the grid connection point of the household as measured by the battery controller                         |
| MeterGridFrequency    | Number | Meter frequency at the grid connection point of the household as measured by the battery controller                       |
| Temperatures          | Number | maximum temperature of the battery cells  |
| CosPhi                | Number | Power factor  |
| ChargeDisp            | Number | Actual maximum available active charge power (AC) of the battery  |
| DischargeDisp         | Number | Actual maximum available active discharge power (AC) of the battery   |
| Status                | Number | 0: disconnected<br>1: connected<br>2: charge<br>3: discharge<br>4: standby<br>5: error                                    |

|                   |           |  |
|-------------------|-----------|--|
|                   |           | 6: busy<br>7: islanding                |
| Alarms            | Array     | Error number                           |
| InverterPVVoltage | Number    | The inverter PV voltage value          |
| WorkingMode       | Number    | The typology of available working mode |
| Load              | Number    | The load parameter value               |
| Operation         | String    | The operation value ("add")            |
| updatedAtTime     | Timestamp | The timestamp of the measurement       |

### PV DATA MODEL

Such collection contains the measurement collected by the RT Monitor from the PV.

**Table 9 – PV data model**

| Field            | Type      | Description   |
|------------------|-----------|---|
| mrID             | String    | The PV ID   |
| Operation        | String    | At the grid connection point of the household as measured by the battery controller |
| ReadingType      | String    | The CIM code for a specific reading typology  |
| timeStamp        | Timestamp | The time of the measurement   |
| value            | Number    | The value of the observation  |
| ReadingQualities | String    | A description of the reading qualities  |

#### 2.2.6.1.1 Module interaction with other tools via WG IOP

The metering manager functionalities is able to transmit information related the metering of a specific Supply point both for near real time data and historical data via WG IOP to other tools like WiseHOME and WiseCORP. The functionalities collect into a specific queue named "resco\_metering" a message/payload that contains metering production and consumption information of a certain supply point for a certain period of time. The billing information are sent via WG IOP every time a WiseGrid tool send a specific request for it. The AMQP protocol is the one adopted between the WG IOP available protocols for its versatility. The structure of the message is explained into the deliverable D4.2 at the WG RESCO tool message structure section.

## 2.2.7 Energy Forecasting Manager Module

### 2.2.7.1 Module Description

The Energy Forecasting Manager is reachable by clicking on the "View Forecast" button present into the

Energy metering manager (see Figure 11). At the same time two requests of Energy forecasting for the day ahead for a specific supply point is sends to the Demand and Production Forecast services (see 5.2.1 chapter) via the WG IOP system. The first one request is about Production forecast and the second one about Consumption forecast, the Forecasting service returns via the same system two AMQP queues with messages that contains Production and Consumption forecast data for the next 24 hours. This data is appropriately read and presented in a graph as shown in the next image (see Figure 1Figure 15). It is possible to know the value of the production forecast by navigating the graph curves by clicking on which the values in Watt or Kilowatt are displayed. It is also possible to hide one of the two curves by clicking on the legend at the top of the graph to allow a better reading of the data. At the time of writing, the functionality shows only the forecast for a single supply point, then the functionality will be expanded with an additional graph with the data of energy forecast for all supply points managed by the RESCO.

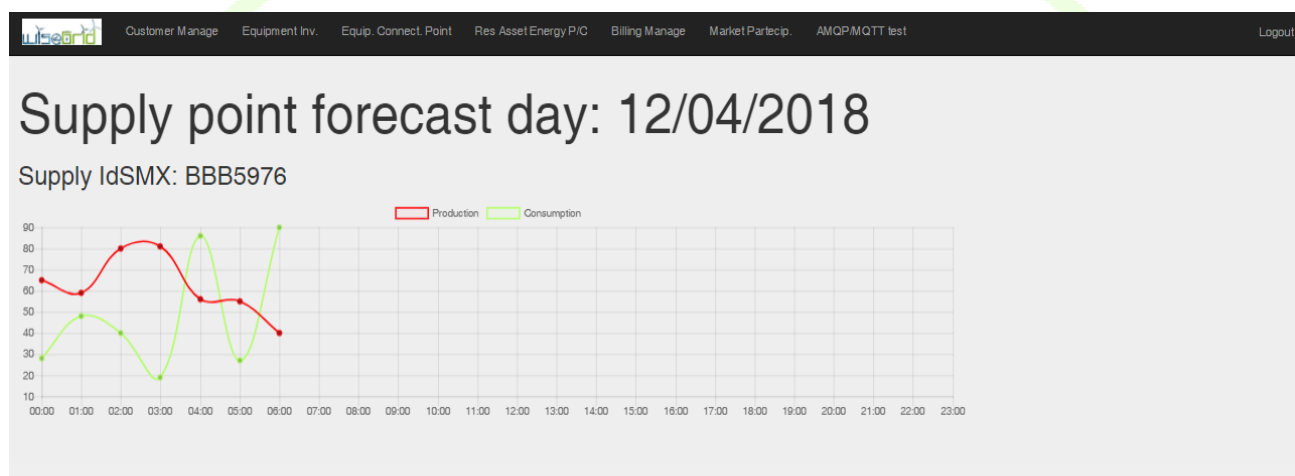


Figure 15 – Supply point Forecast

### 2.2.7.2 Module Data Model and Algorithmic Framework

The forecasting module will ask for new forecasting profiles by means of the forecasting service described in section the deliverable D4.2 [4]. Based on this service input and output variables required to the forecasting service are describe next.

#### 2.2.7.2.1 INPUTS VARIABLES

For each forecast to be generated, this module will take into consideration next inputs from the client:

- The client identifier. One client identifier is associated to an aggregation thus the same client application can request a forecast under different client identifiers.
- The period of time between two consecutive forecasting values.
- The total window horizon for the forecast output.

The algorithm has also a high number of internal parameters that will be automatically optimized to obtain the best prediction, such as: number of past days for training, number of past hours for prediction and variables related with internal supervised learning and optimization algorithms.

Furthermore in the Long Term database should be available information about historical data and calendars described below. In addition, it is mandatory to have next information in the long term data base:

- The aggregation of “ Supply Point ID” “(concerning consumers and productions) associated to the same client identifier.

- In the case of productions, the energy resource type of each CUPS, the possible renewable resource considered in the scope of the project are photovoltaic and wind power. Thus, a production forecast request can consider a mix of different renewable resources.

Finally, production forecast service request weather forecast to the service developed in the scope of the project. Specifically, the detailed information required from the weather forecast to the production forecast service is detailed below.

Historical data will correspond with the historical information of the load/renewable generation profile for each supply point.

The information that will be included in this input will be:

- Time stamp (type: timestamp). Recorded day and time concerning the historical information.
- Value (type: int16). Recorded power value at the specific timestamp in watts [W].

Calendar input will be a two-dimensional matrix with the detail of working days, holidays and weekends for a whole year. Therefore, as many calendars as regions will be necessary.

Information contained in this matrix will be:

- Day (type: day [dd-mm-yyyy]). Day of the year
- Type of day (type: int8). It could have the values (1) working days; (2) holyday

The weather forecasts will need, as an input, information concerning the maximum, minimum and average temperature of a day for the next days to be included in the forecasting window. Thus, the information contained in this matrix will be:

- Day and hour (type: day [dd-mm-yyyy hh:mm]). Day of the year and hour.
- Wind speed (type: int8). Meters per second.
- Wind direction: SSO, NNO, etc.
- Average temperature forecasted in Celsius degrees (type: int8)
- Maximum temperature forecasted in Celsius degrees (type: int8)
- Minimum temperature forecasted in Celsius degrees (type: int8)
- Solar radiation (type: int8). W/m<sup>2</sup>

#### 2.2.7.2.2 OUTPUT VARIABLE

The demand forecast module will have only one variable which corresponds to a bi-dimensional matrix where the first column contains the time stamp and the second column the value of the forecast in the specified unit:

- Time stamp (type: timestamp). Day and time of each forecast value.
- Value (type: int16). Forecasted value at the specific timestamp.
- Unit (type: string). Unit regarding values, normally W or kW.

### 2.2.8 Market Bid Manager Module

#### 2.2.8.1 Module Description

Related with the previous point, tools used by RESCOs shall facilitate the management of all information required by them to eventually sell energy in the Energy market. This information mainly includes production estimations and real-time monitoring, assistance to calculation of proper offered prices, and

management of accepted bids and imposed limits to the generation.

The Market Participation functionalities reachable by the RESCO tool main menu, show to the RESCO company a graph with information related the Energy forecast production and consumption for the day ahead aggregated for all the supply point that the RESCO manages. The aggregated energy surplus value is indicated by the “RES energy surplus Offered” field in the lower side of the functionalities, this value represents the energy that for the day ahead could be sell.

The below screen serves as proof of concept which provides the required information needed by the RESCO to eventually participate to the Energy Market.



Figure 16 – Market Participation

## 2.2.8.2 Module Data Model and Algorithmic Framework

### 2.2.8.2.1 MARKET\_PARTICIPATION

Such collection contains the data of the market participation for a RESCO supply point.

Table 10 – Market participation data model

| Field       | Type   | Description                      |
|-------------|--------|----------------------------------|
| kWh_offered | Number | The kWh offered to the market    |
| Stard_date  | Date   | The start date of the Kw offered |
| End_date    | Date   | The end date of the Kw offered   |
| Price       | Number | The price in Euro for one kWh    |



## 2.2.9 Investment Decision Support Manager Module

### 2.2.9.1 Module Description

Part of a RESCO business is based on the gain from the sale of the energy produced by its renewable energy plants. The RESCO RES installation is provided directly from the RESCO to a customer's premises and the RESCO must certainly pay close attention before sign a contract with the customer to be sure that this can give an economic advantage in the short and long term especially when the contract is based on the sale of the electricity produced and not only on the payment of a fixed fee. Certainly RESCO company by sign a contract with a customer will have to bear the managing costs of the various assets that is proposed to install at a specific supply point and will have to guarantee the assets maintenance, whether scheduled or not. This obligation involves some fixed costs like asset installation and programmed asset maintenance and unanticipated costs (due to plant problems) which must be carefully considered in order to avoid a contract resulting in a negative rather than a positive investment for the company. In this regard, the investment support module is a PoC (Proof of Concept) with the aims to provide RESCO company with a series of information that can help it make a decision on whether or not the investment.

### 2.2.9.2 Module Data Model and Algorithmic Framework

For each support investment report to be generated, this module should take into consideration next inputs from the RESCO tool modules and external providers to determine the economic investment return from a hypothetical supply point RES installation.

Input variables:

- Installation cost of the plant and relative pre work costs of the places where it will take place to make the location suitable to a specific type of plant (not all supply points may have ideal structural conditions for certain types of plant).
- Different labour costs from country to country due to different salaries of skilled technicians.
- Facility to access to places for maintenance to avoid additional maintenance costs (Easy access, difficult access).
- Possibility of different tax deductions different from country to country and from region to region.
- Atmospheric condition of the different installation points that may make one type of asset or another prefer to be installed. (wind turbine suction zones, sunny zones for photovoltaic panels, etc...).
- Historical data acquired over the time from existing plants to understand if the amount of energy produced and consumed into a specific geographical point is convenient or not for one type of asset or another.
- Examination of past invoices by customer type and contract type and comparison of profit margins.
- Sales prices of energy from renewable sources for the specific country.

All this information are directly available from the RESCO tool by analyzing for example stored data of the Billing module to deduce the real profit of a specific supply point. By analyzing forecasting electricity production and consumption historical data provided by the Forecast module and the corresponding real data of production and consumption provided by the Metering manager module. It is also possible analyze of the average time to repair faults through the Maintenance module historical data to try to understand the



maintainability of particular supply points and if present the information of related costs invoiced by the maintenance company. All these parameters should be evaluated by the algorithm by scoring each of the parameters on a fixed scale so as to set a range within which their sum determines whether an investment can be considered profitable or not for the RESCO.

### 3 DEVELOPMENT VIEW OF RESCO TOOL

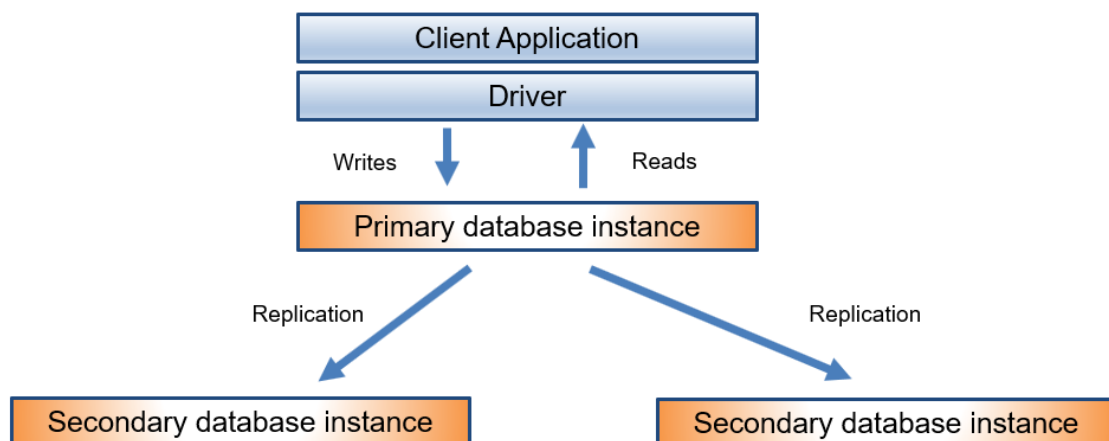
This chapter provides an overview about the common repository used for the long term data storage and some details about the development of the WG RESCO tool web application.

#### 3.1 RESCO COMMON REPOSITORY

The WG RESCO tool requires for long term data storage access to a centralised, cloud-based database platform. The platform will be used for storing a database named “wgresco” containing the data related to the WG RESCO application.

The cloud based database management system that will be used in WiseGRID project is a NoSQL database management system. NoSQL sometimes translated as “Not Only SQL” define a complementary database management system to relational databases (RDBMS) that came with multiple advantages for an application like WG RESCO part of WiseGRID ecosystem. More details about the Big Data Platform and the used technologies are available in the deliverable D5.1 [5].

The RESCO application will directly interact with the application Server /Router Cluster computers through an appropriate driver



**Figure 17 – Application access to Big-data Platform**

The RESCO application will create a database named “wgresco” on the Bigdata Platform to store and retrieve the data from the platform. Access will only be possible by a user/password pair.

The “wgresco” database does not present issues approach related to personal data. Detailed analysis and evaluation of personal data exposure are presented in deliverable D3.2 within the DPIA process.

Currently the WG RESCO tool stores its data in two different data base provided by the Big Data Platform.

The first one the “Local DB” for the Customer data, Contract data, Asset data, Configuration data and real time meter observation coming from RES source and stored by the RT Monitor tool.

The second one the “Long term DB” to store historical data like meter observation coming from RES source and all the data processed by the RESCO tool that needs to be stored for future elaborations.

Two different approaches were adopted to read and write data in these two databases:

- Direct access to the Local DB and Long Term DB through specific java script of the RESCO tools;
- Access through external module that by the Wisegrid Interoperable Platform (WG IOP) is able to share data readed from the Long Term DB and store in it.

Both approaches are written in java script and use Mongoose that is an Object Data Modeling (ODM) library for MongoDB and Node.js. It manages relationships between data, provides schema validation, and is used to translate between objects in code and the representation of those objects in MongoDB.

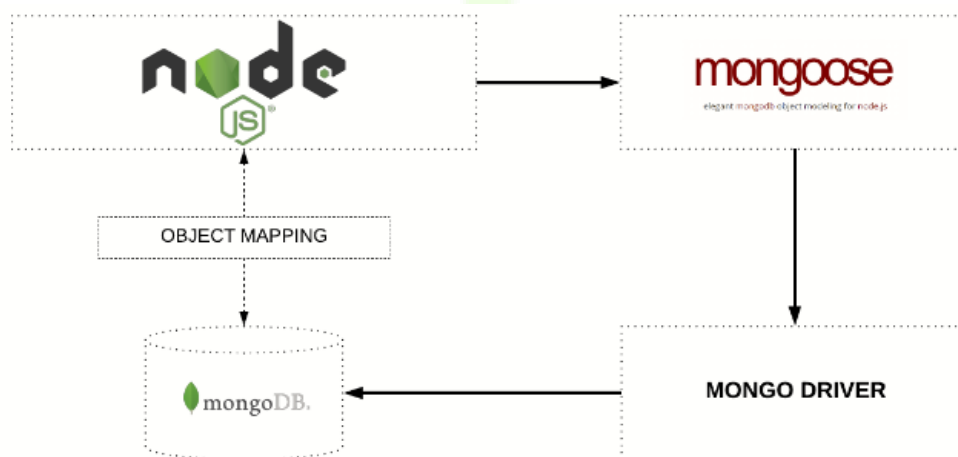


Figure 18 – Object Mapping between Node and MongoDB managed via Mongoose [6]

### 3.2 RESCO WEB-APP

The web-app of the RESCO tool allows the logged-in user to benefit from the different functions that each module defines. In order to harness its architecture it has been developed using the MEAN.JS full-stack JavaScript solution, this choice has been made to build fast, robust, and maintainable production web applications using MongoDB, Express, AngularJS, and adopting Node.js as engine. The web-app is exposed on the network thanks to Nginx reverse proxy.

#### Mean [7]

MEAN.JS is a full-stack JavaScript solution that helps you build fast, robust, and maintainable production web applications using MongoDB, Express, AngularJS, and Node.js.

#### Express [8]

Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications. It provide a myriad of HTTP utility methods and middleware at your disposal, creating a robust API is quick and easy and a thin layer of fundamental web application features, without obscuring Node.js features.

### Angular [9]

AngularJS is a structural framework for dynamic web apps. AngularJS is the MVC framework through which the WG RESCO tool web application side has been developed. Thanks to the fact that with AngularJS applications are defined with modules that can depend from one to the others, very powerful templates directly in RESCO tool HTML part have been defined with new attributes or tags and expressions, encapsulating the behavior of the RESCO tool in presenter. Moreover, thanks to the use of dependency injection provided by AngularJS, the test JavaScript code was structured very easily. Finally, AngularJS allowed to use HTML as template language as well as to extend HTML's syntax to express RESCO tool application's components clearly and succinctly. Angular's data binding and dependency injection provided the development team with the necessary features to make the implementation faster and easier.

### Node.js [10]

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world.

### Ngix [11]

NGINX is known for its high performance, stability, rich feature set, simple configuration, and low resource consumption.

NGINX is one of a handful of servers written to address the C10K problem. Unlike traditional servers, NGINX doesn't rely on threads to handle requests. Instead it uses a much more scalable event-driven (asynchronous) architecture. This architecture uses small, but more importantly, predictable amounts of memory under load. Even if you don't expect to handle thousands of simultaneous requests, you can still benefit from NGINX's high-performance and small memory footprint. NGINX scales in all directions: from the smallest VPS all the way up to large clusters of servers

### Bootstrap CSS framework [12]

Bootstrap is a free and open-source front-end web framework for designing websites and web applications. It contains HTML- and CSS-based design templates for typography: forms, buttons, navigation and other interface components, as well as optional JavaScript extensions.

Bootstrap is modular and consists essentially of a series of *Less stylesheets*, that implement the various components of the toolkit. A stylesheet called "Bootstrap less" includes the components stylesheets. Developers can adapt the Bootstrap file itself, selecting the components they wish to use in their projects. The Less stylesheet language allowed the development team to use variables, functions and operators, nested selectors, as well as mixing for the implementation of the RESCO tool Web-app.

## 4 DEPLOYMENT, PROTOTYPING AND TESTING ENVIRONMENT

Along with the development view of the system component special focus is delivered on the deployment view of RESCO tool. The minimum hardware and software requirements for the deployment of the full application are presented.

For hardware requirements, An 4-core (or superior) CPU at 2GHz, 8GB of RAM, 100 GB hard disk and a 64-bit CentOS 6.6 (core linux intsys-trentour 2.6.32-431.29.2.el6.x86\_64) or higher is the minimum of requirement.

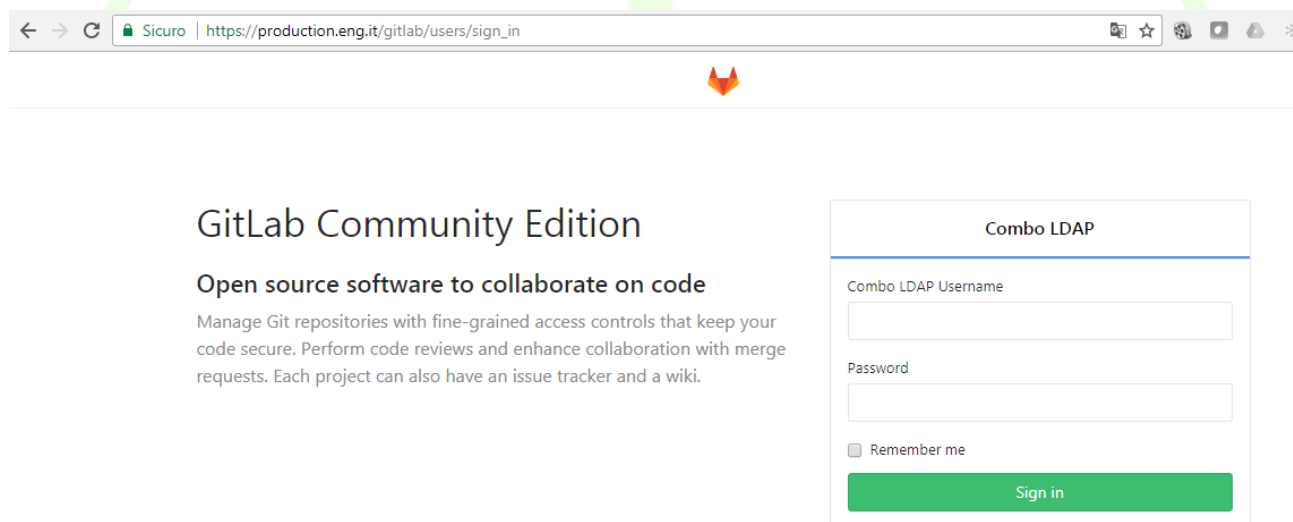
As software requirement the Node Version Manager (NVM) [1] is required into the server that hosts the RESCO tool software with the aim to facilitate the multiple active node.js versions. The data storage is performed on the MongoDB [13] database for which there are two instances, one for the Local DB and one for the Long Term DB, these two instances are made available by the WiseGRID project Big Data platform.

There are 2 ways of deployment the services in local servers: either plain installation in the server machine (if this fulfils the minimum of hardware & software requirements) or setting a Virtual Machine for managing separately the different applications. This approach enables us to deploy the RESCO tool either as a host service or as an application running in a cloud service provider (with the appropriate customizations).

The aforementioned deployment analysis defines the list of prerequisites for the demonstration of the RESCO tool at the different pilot sites (though as presented above we are fully flexible at any possible deployment). The current deployment of the RESCO tool is available in a cloud server, managed by the ENGINEERING DATA CENTER of Pont-Saint-Martin (AO) as the leader of this task. The web-ui is currently accessible at the following url: <http://wisegrid.esf.eng.it/resco/login>. This URL is managed by Nginx reverse proxy . The WEB-UI access is protected by username and password authentication.

The software developed and its versioning is saved on a Git repository offered by Engineering to all project partners.

It is possible to get the last version of the WG RESCO tool linking on <https://production.eng.it/gitlab/WiseGrid/rescotools> by accessing with user provided user credentials.



**Figure 19 – GitLab login page**

Next image show the repository tenant that hosts the source code of the RESCO tool and its related source directory main structure.

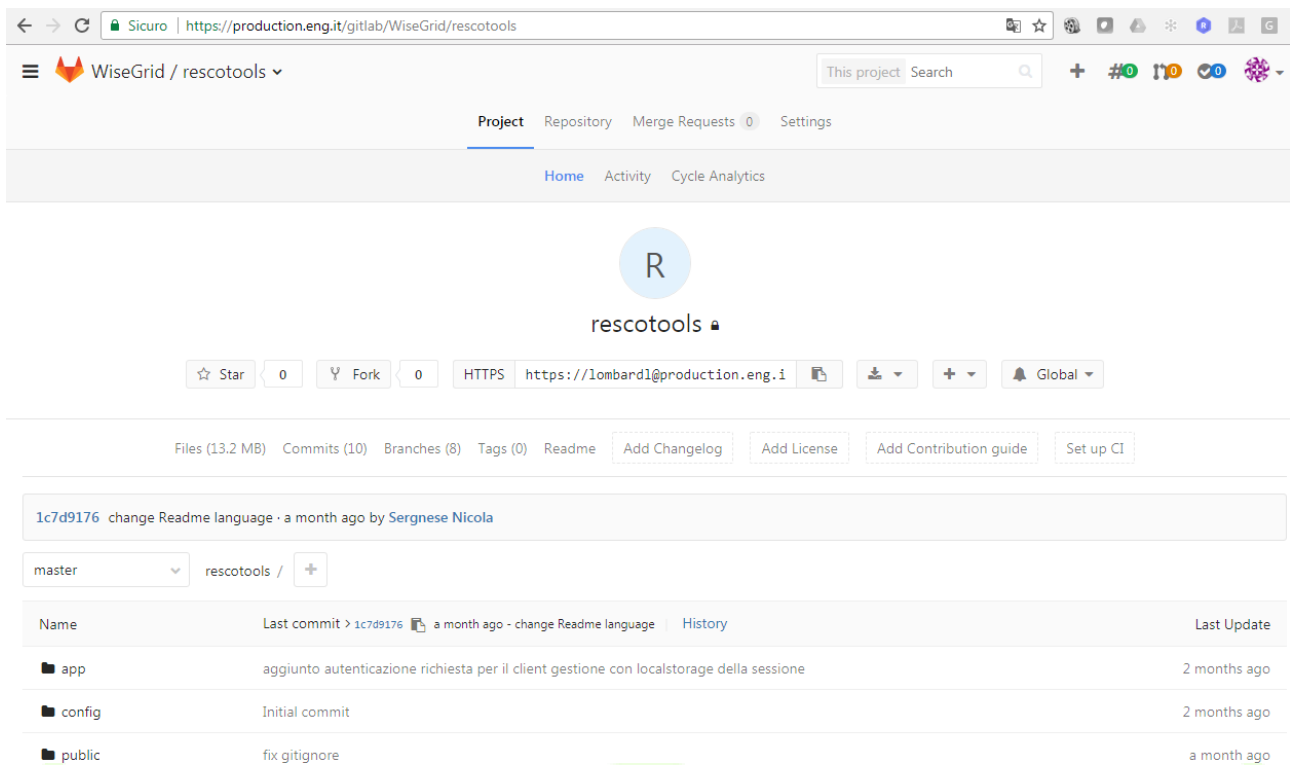


Figure 20 – GitLab software repository page

## 5 ADVANCED MODELS FOR SMARTENING THE DISTRIBUTION GRID

### 5.1 SUPPORTING THE DISTRIBUTION GRID OPERATION WITH DISTRIBUTION MANAGEMENT SYSTEM (DMS) SERVICES

An advanced Distribution Management System is a software platform which incorporates a full set of functionalities that support the management and the operational optimization of the distribution system and allows the personnel to effectively monitor and control it while improving safety, reliability, asset protection and quality of service. It includes monitoring, analysis, control, optimization, training and planning tools that all function on a common representation of the entire electric distribution system. WiseGRID deliverable D12.1 reported and described a full list of functionalities that a common DMS usually incorporates. However, the exact tools that a DMS system has usually depends on the requirements and the specific needs of the DSO that acquires it.

One of the basic objectives of the WiseGRID project is to provide “a set of solutions and technologies which increase the smartness, stability and security” of the consumer centric electricity grid. In order to achieve this goal one of the aspects that were identified and need to be addressed is “Smartening the Distribution Grid”. This necessarily means that a set of functionalities that enhance grid operation and assist DSOs in the monitoring and management of the system need to be designed and developed. The core functionalities that have been identified from the WiseGRID project as fundamental for the tool that will provide the DMS

services, which is the WG Cockpit, and which will be developed in the context of WP13 are, briefly, the following:

- Demand and Production Forecast.
- Energy Quality Monitoring.
- Load Flow/State Estimation.
- Congestion Forecast.
- Outage Management – Fault Location, Isolation and Restoration

By using, analysing and evaluating the results of these core functionalities the grid operator will be assisted in the operation, maintenance and planning of the distribution system. Also, this set of functionalities can provide the basis for the development of a large number of higher-lever applications that can extend the capabilities of the tool and provide more user-specific results. The presented functionalities are extensively described at the following sections.

At the following section the technical description of the selected functionalities is presented in detail. However, more information regarding the implementation and integration of them, WiseGRID Cockpit, that is the DSO support information tool designed and developed in WiseGRID, can be found in deliverable D13.1 [14].

## 5.2 FUNCTIONALITIES AND SERVICES OF A DMS

This section presents the functionality and theoretical background of the services to be considered and implemented in the WP13.

### 5.2.1 Demand and Production Forecast services

The main objective of this service is the forecasting of the electrical energy consumptions and generations requested by the RESCO tool.

The load and generation forecast is a very valuable outcome for RESCO to optimize the energy management of their clients. Furthermore, this information is important for the distribution system operator (DSO), as it can be used to predict the status of the system and avoid faults or other problems.

RESCO will gather conveniently load demand and production data of its clients individually or into different groups. Then, one forecasting model shall be built for each group in order to provide aggregated load demand and production forecasting.

For building the forecasting models, a minimum of data variables would be required, including not only historical load demand/production data, but also variables related with daily and seasonal variability as well as endogenous variables related such as climate variables

The demand and production forecast provider is a wrapper of the forecast algorithms for an easy integration with the applications of the WiseGRID ecosystem, including WiseGRID Cockpit.

The forecasting module developed in the WiseGRID project could be seen as a function that forecasts the next desired values taking into consideration historical data and additional data such as a calendar (with working days, holidays and weekends), exogenous variables (weather) and some client parameters.

Concerning the forecasting algorithm integrated in the forecasting service, it has been implemented under CNEA (Cascaded Neuro-Evolutionary Algorithm) forecasting model premises that are based on neural networks algorithms. The CNEA forecasting model consists on some SVM (Support Vector Machine) in cascade with an optimal chose of input models, number of neurones and evolutive training process (see Figure 21). The iterative CNEA algorithm uses a SVM to forecast the power of the first hour (h+1) this result is used as an input of the next SVM to forecast the power for the next hour (h+2) and the algorithm repeat this pro-

cess until the last forecasting slot is achieved. Thus, it is required 24 SVM executed in cascade for doing a forecasting of 24 values [15].

The proper way to obtain a good forecasting result in a short term power forecasting is to use past days with similar power profiles. Therefore, if the historical information available in the long term data base is not enough the error of the forecasting result will increase. To reduce this error, in the first step of the CNEA model, it is chosen the more influencer variables from the data base in an optimal way.

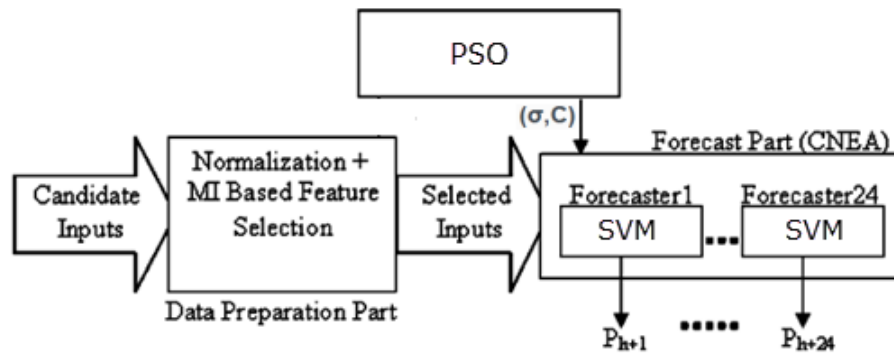


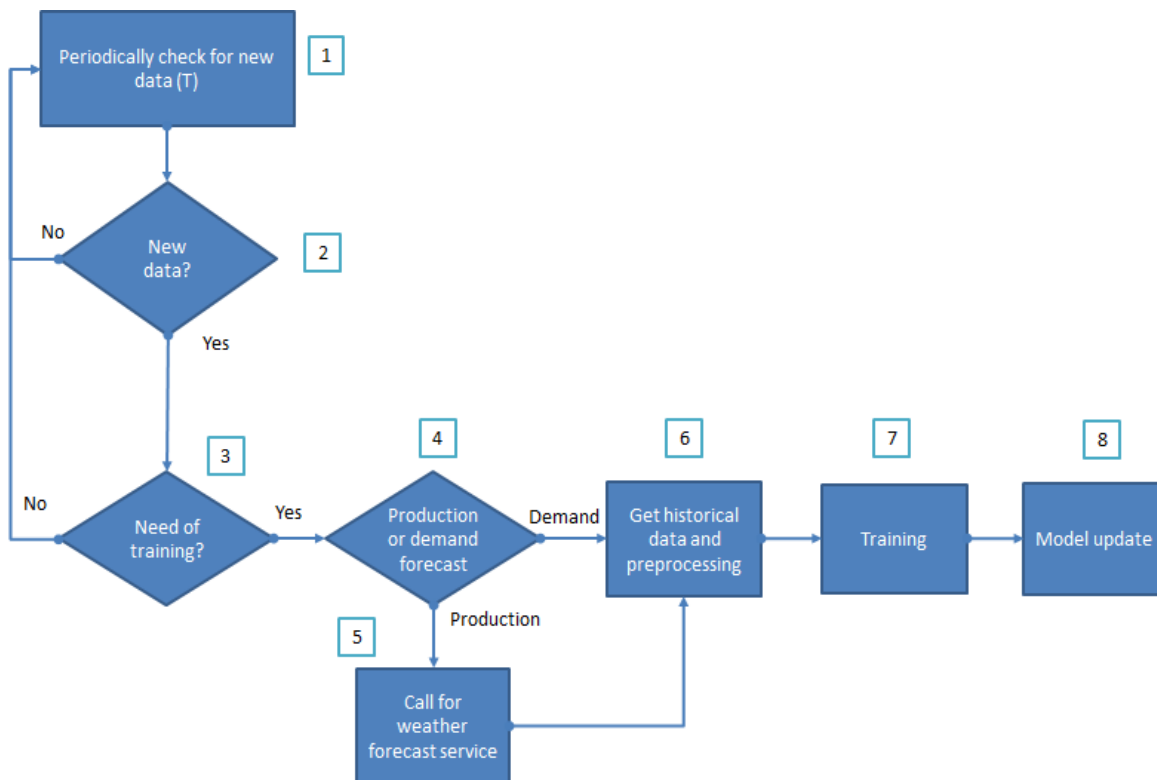
Figure 21 – CNEA network structure

The forecasting algorithms will have two different workflows: one for training the models and the other one to attend the forecasting requests.

The workflow for training the forecasting models runs once a day (at the end of the day) and follows next steps:

1. This service works periodically, checking for new historical data for each client.
2. If new data is available on Long Term DB next step will be reached, otherwise return to step 1.
3. Not always is needed to train forecasting models. Depending on the restrictions could be not necessary re-train the model.
4. If the request is about demand, it would be only necessary historical data, at least initially. If we talk about production it could be needed a weather forecast.
5. If the model to be trained is about production, a call for the weather historical and forecast data will be needed. The client location should be able into the database to perform these calls.
6. Historical demand / production data will be obtained through a Long Term DB call.
7. With all collected data the training of the model is performed.
8. Finally, old models are deleted and the new one remains to perform new forecasts.





**Figure 22 – Training workflow schema**

Finally, the workflow to attend the forecast requests follows next steps:

1. A client asks for a forecast by invoking a RPC function and adding some input parameters.
2. The identifier of that client is checked to know if exists.
3. The input parameters are checked to validate coherence.
4. It is checked if a model is trained and historical data is available.
5. If the request is about demand, it would be only necessary historical data. If we talk about production it could be needed a weather forecast.
6. If the request is about production, a call for the weather historical and forecast data will be needed. The client location should be able into the database to perform these calls.
7. Historical demand / production data will be obtained through a Long Term DB call.
8. Forecast is performed and returned to the client.

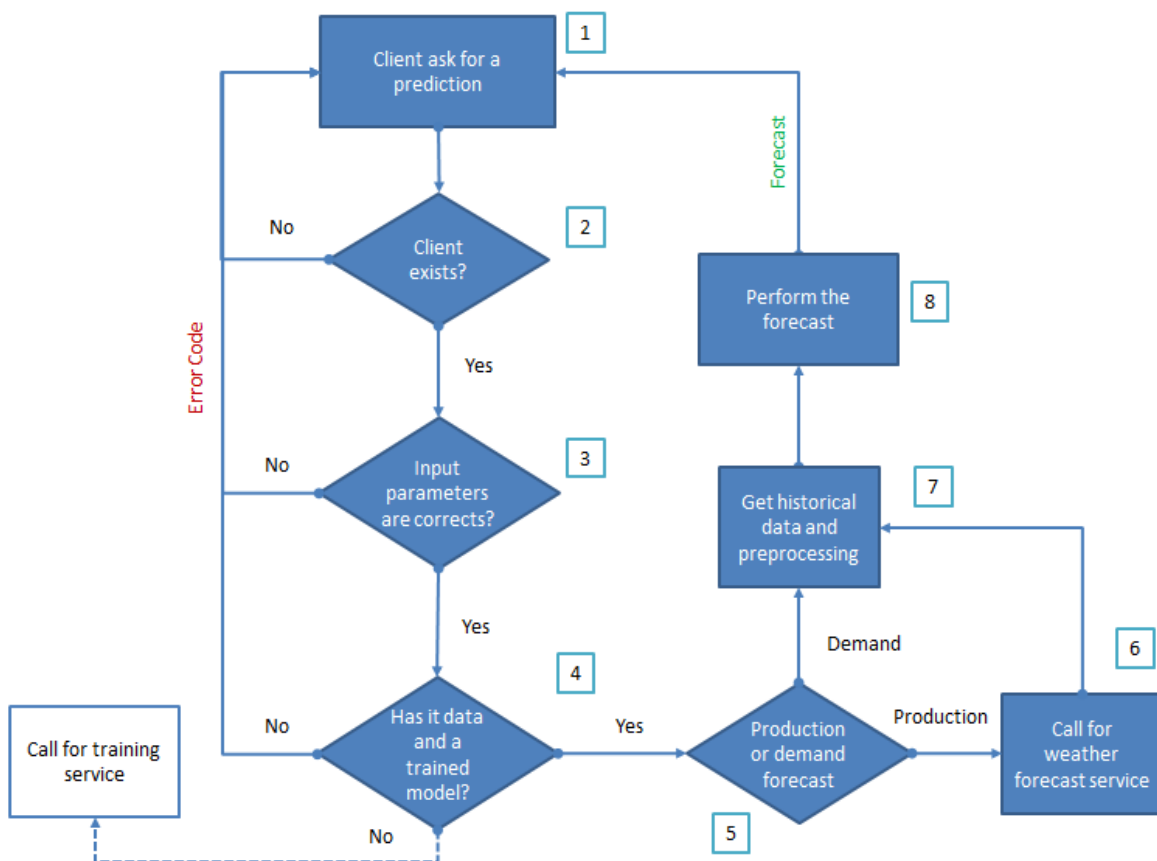


Figure 23 – Forecast workflow schema

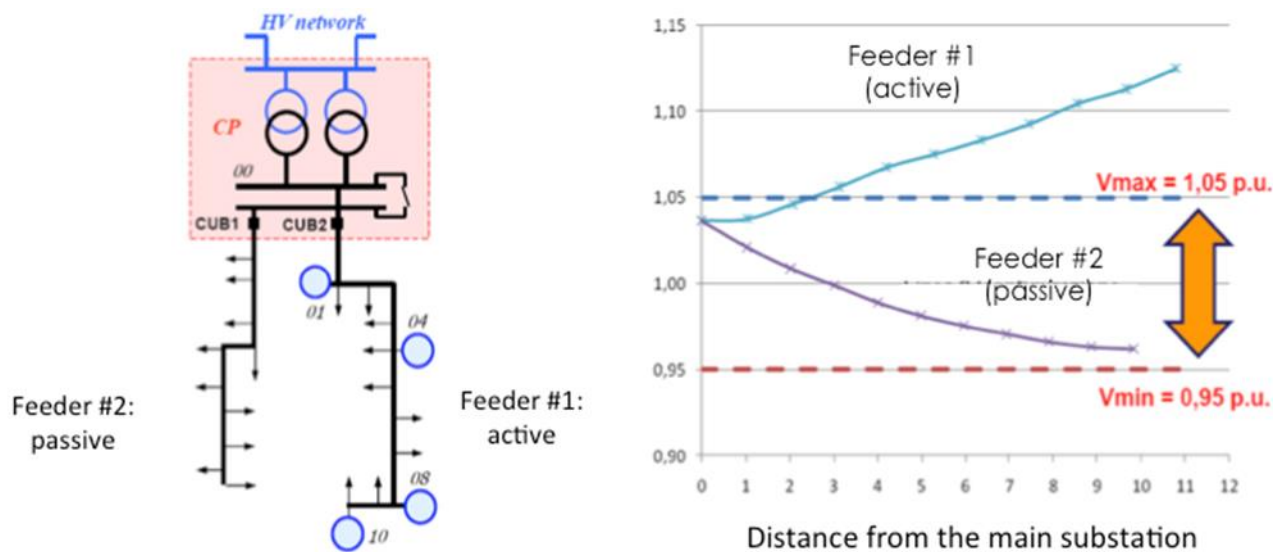
### 5.2.2 State Estimation and Load Flow

During the last years, the transition from passive to the so-called Active Distribution Networks (ADNs) has been accompanied by the following changes [16]:

Increased electricity customers participation

- Integration of Renewable Energy Sources (RES)
- Necessity to handle network congestion
- Development of Information and Communication Technologies (ICT)
- Investment on network renewal/modernization (ageing assets)

The large penetration of Distributed Energy Resources (DERs), consisting of Distributed Generation (DG) units, storage technologies and demand responsive loads has completely modified the network voltage profile. ADNs are characterized by local decentralized production (closer to the loads) and bidirectional power flows. This may result in increased voltage levels, which - in some cases - may be outside the allowed limits as imposed by international standards and guidelines, as it is shown in Figure 24. Consequently, it is important to know the state of the system and, whenever it is needed, perform voltage/power control in order to bring voltage levels back to the desired range.



**Figure 24 – Impact of DERs on the network voltage profile**

In power systems, the static state of the grid is typically defined as the phase-to-ground voltage phasors (i.e. voltage magnitude and phase) at all network all network buses [17], [18], [19]. By knowing the voltage phasors everywhere and for a given network topology, it is possible to calculate the injected currents, the branch currents and the active and reactive power at all network buses; therefore, it is possible to infer the system operating conditions by using as minimum information a) the voltage phasors and b) the network topology that is expressed by the network admittance matrix.

### **Three Phase Power Flow**

Power-flow or load-flow analysis is the mathematical tool that allows to determine the, previously described, static states of the grid and it has been traditionally used for grid operation, grid planning and other grid-related studies. The power flow study is essentially an analysis on the system's capability to adequately supply the connected load. Until recently, the power flow tools were mainly, if not exclusively, used for the transmission systems and, thus, the algorithms were developed for the solution of the mathematical problems based on the modeling of these systems.

The power flow analysis of the distribution network is similar to that of an interconnected transmission system, however there are some significant differences due to the different characteristics of the two types of systems. More specifically, the loading of a distribution feeder is inherently unbalanced due to a large number of unequal single-phase loads and the non-symmetrical conductor spacing of three-phase underground and overhead line segments. The conventional power flow programs used for transmission systems assume a perfectly balanced system and, as a result, the single-phase equivalent can be used. Furthermore, these algorithms do not show good convergence properties for distribution systems due to the radial structure and the high R/X ratio of the latter. As a result, the conventional power flow analysis tools do not provide the actual behavior of the network. A three-phase power flow analysis is necessary to identify the issues related to distribution grids and to obtain the actual behavior of the system.

### **State Estimation**

The need to perform state estimation (SE) is driven by the fact that the true system state is not known. The only available piece of information is the measured system state. As a consequence, it is crucial to use all available measurements in order to get an approximation that is as close as possible to the true system

state. The approximation consists in the estimated system state [20]. The latter can be obtained by making use of well-known or more sophisticated SE algorithms, as it is described in more detail in the D13.1.

Traditional state estimators have been designed for transmission systems and have been based on 1-phase equivalent circuits (symmetric). In transmission systems, SE has been used in applications such as situation awareness, detection of inter-area oscillations, protection, stability assessment and "N-1" security assessment [21], [22], [23].

However, the characteristics of ADNs, such as radial topology, high R/X ratios, unbalanced lines and loads, lack of real-time measurements, larger harmonic distortion, higher values of measurements noise, and increased penetration of volatile unpredictable RESs, have made it necessary to design 3-phase state estimators, which shall be characterized by:

- high estimation accuracy (i.e. small values of estimation error, which is defined as the difference between the measured and estimated system state)
- high measurement reporting rates and estimation refresh rates and
- low time latency.

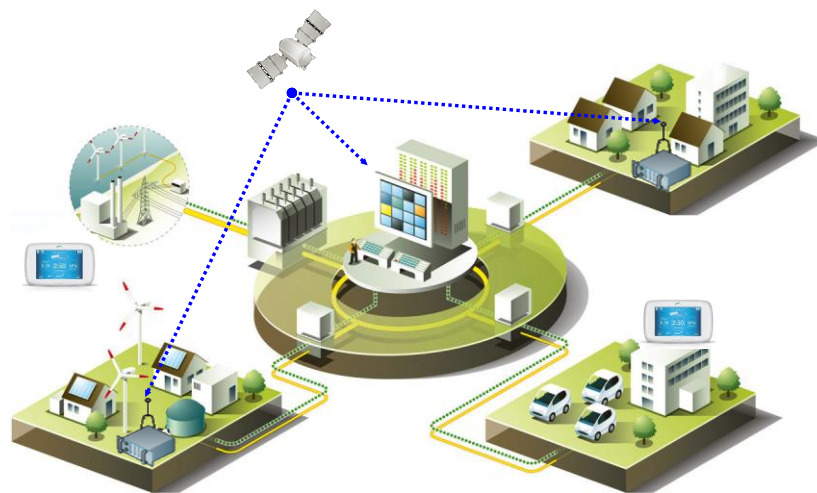
State estimators deployed in distribution networks target the following applications:

- RT control (voltage, line congestions) ) (e.g. [24], [25]),
- protection and fault detection, and
- network monitoring (e.g. [26]).

Typical refresh rates of existing estimators are in the order of 15 minutes, since they are dictated by the reporting rates of conventional measurement devices, such as Remote Terminal Units (RTUs). This order of refresh rates makes it impossible to track the high ADNs dynamics. New emerging measurement technologies, such as Phasor Measurement Units (PMUs) allow to acquire measurements every 20-100 ms and this has a very significant impact on the SE refresh rate as well, which can shrink to some ms [27]. The time latency is defined as the total time from the time-stamping of the measurements and their acquisition up to moment that the system state is calculated [20], [28].

Figure 25 shows a typical SE architecture, where measurements are obtained by RTUs and PMUs. PMUs are GPS-synchronized whereas RTUs are typically not GPS-synchronized. Measurements from RTUs are gathered in the Supervisory Control And Data Acquisition (SCADA) system, whereas measurements from PMUs are gathered at another system, which is called Phasor Data Concentrator (PDC). The PDC should be suitably coupled with conventional SCADA systems, in case a hybrid measurement model is used. It is also important to guarantee a good communication between the two different systems in order to enable the efficient exchange of information between them. The data that are collected by the PDC can be either sent to other PDCs or be used in order to feed other applications. Many PDCs that belong to different utilities can be connected to a common central PDC (the so-called "super-PDC"), whose role is to aggregate data across the different utilities and provide an interconnection-wide snapshot of the acquired measurements.

Figure 26 shows the different kinds of data that can be fed to the PDC by PMUs. The so-called "data-frame" consists of a set of synchrophasor, frequency, and Rate of Change of Frequency (ROCOF) measurements that corresponds to the same time-stamp. Other data may optionally include the calculated active and reactive power etc. Each PMU provides its own data frame and/or other data to the common PDC.



Involved(components/technologies:)

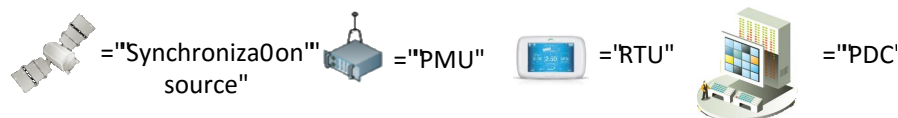


Figure 25 – Typical SE architecture

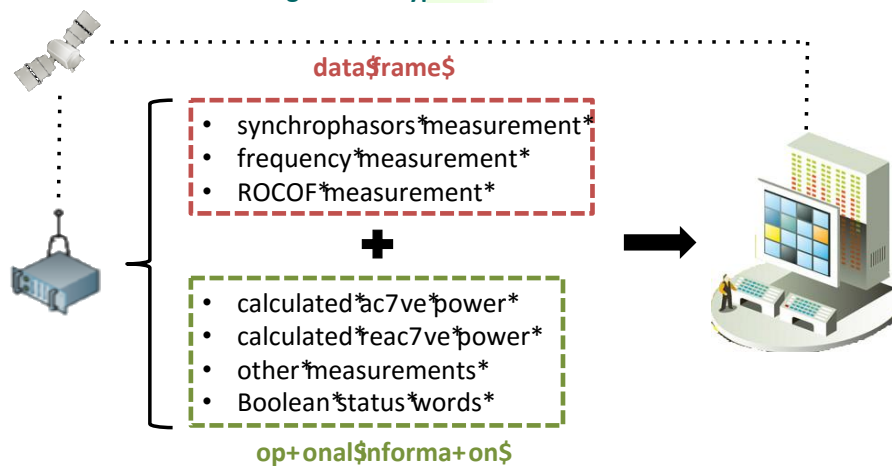


Figure 26 – Different kinds of data that can be fed from PMUs to the PDC

### State estimation vs. Load flow

Load flow (LF) analysis has been used for many years to calculate the system state of electrical networks. It would be important to mention the differences between LF and SE, in order to understand the advantages of the latter over the former.

In LF analysis, the input data are:

- the injected active and reactive power (at load buses)
- the injected active power and the voltage magnitude (at generation buses).
- the voltage phasor at the slack bus (reference bus).

However, if any of the above-mentioned data are missing or are not available, then the solution of the LF problem may be tricky. In particular, as shown in Figure 27 – Examples of LF failure, LF produces a wrong result in case

- A. Some measurements are missing (LF cannot be computed)

- B. Some measurements are inaccurate (LF gives a wrong solution)
- C. Some measurements are wrong (LF gives a wrong solution)

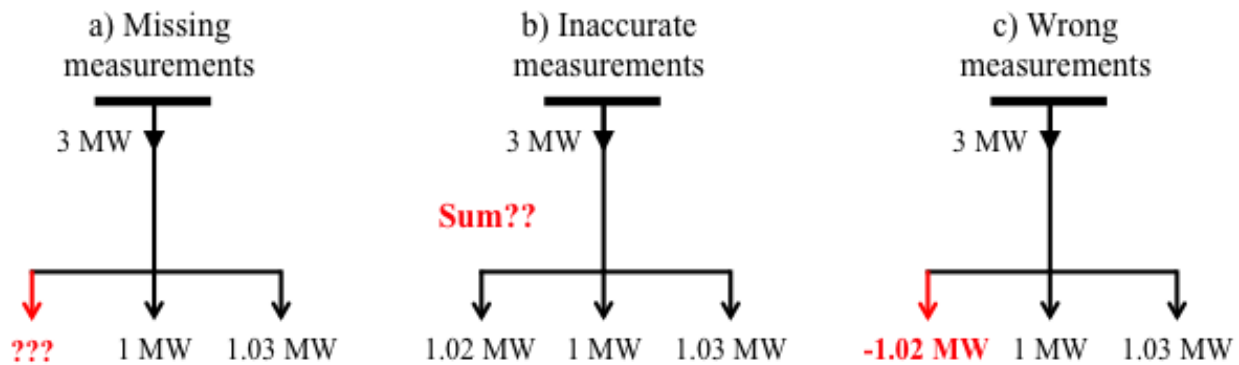


Figure 27 – Examples of LF failure

In order to deal with the above-mentioned problems, the LF analysis has been suitably combined with estimation theory and statistical analysis; the outcome is SE. In particular, SE compared to LF:

- takes into account the fact that measurements contain noise (measurement error);
- uses all types of measurements (e.g. voltage and current magnitude and phase, injected and branch currents and powers etc.) and incorporates all different information concerning the network topology;
- uses novel methods to detect, identify and replace bad data;
- uses extra measurements (the so-called “redundant measurements” to increase estimation accuracy and
- solves an optimization problem in order to calculate the best estimate of the system state, which shall be accurate and reliable.

### 5.2.3 Energy Quality

Energy Quality (a term also known as Power Quality) has emerged as a very important issue in modern electrical power systems, especially nowadays when the increasing level of renewable power penetration and electric vehicles connection introduces several negative impacts. Smart grids are required to deliver power at a high-quality level to consumers, as insufficient power quality performance may have detrimental financial effect on both consumers and utilities in distribution systems.

The term energy quality indicates the deviation of both voltage and current from their ideal waveforms. An ideal waveform is considered to be sinusoidal, with fixed frequency and amplitude, any deviation from which is considered a disturbance. These disturbances may cause increased system losses, electrical equipment overheating, insulation breakdown or even power interruptions. With the increased use of non-linear loads such as power electronic switching-based equipment, and computer loads, as well as due to time varying single phase and three phase loads, the deformation of voltage and current from their ideal waveform has increased to a large extent.

Typical power quality phenomena in power distribution systems are referred to harmonics, unbalance and voltage dips. Thus, the quantification of the deformations caused by these phenomena, using appropriate indices, as well as the satisfaction of acceptable limits (according to the European Standard EN 50160), has



become a very important issue in modern power systems. Therefore, many power quality indices, e.g., Total Harmonic Distortion of Voltage and Voltage Unbalance Factor, have been established worldwide, offering methods to compress raw information, usually multidimensional in nature, into a single value.

### Harmonics

The term Harmonic Voltage indicates the sinusoidal voltage with a frequency equal to an integer multiple of the fundamental frequency of the supply voltage. Harmonics of the supply voltage are caused mainly by network users' nonlinear load connected to all voltage levels of the supply network, as well as transformers, rotating machines and power electronics. The main effects of harmonics include thermal losses, load disruption, and reduction of electrical equipment's lifetime. Moreover, harmonic currents flowing through the network impedance rise to harmonic voltages. Both harmonic voltages and currents usually vary in time. There are two different ways in which harmonic voltages can be calculated:

- individually, by their relative amplitude related to the fundamental:

$$U_h = \frac{V_h}{V_1}$$

where:  $h$  is the harmonic order,

$V_1$  is the amplitude of fundamental voltage,

$V_h$  is the amplitude of  $h$  harmonic voltage, and

$U_h$  is the relative amplitude of  $h$  harmonic voltage

- globally, by the Total Harmonic Distortion Factor of Voltage, given by:

$$THD = \frac{\sqrt{\sum_{h=2}^{+\infty} V_h^2}}{V_1}$$

According to the European Standard EN 50160, under normal operating conditions, during each period of week, 95% of the 10 min mean rms. values of each individual harmonic voltage shall be less than or equal to the value given in Table 1, presented below. Table 1 includes the upper limits of individual harmonic voltages at the supply terminals for orders up to 25 given in percent of the fundamental voltage  $U_1$ , Harmonics of order higher than 25 are usually small, but largely unpredictable due to resonance effects. Moreover, the total harmonic distortion of supply voltage (including all harmonics up to the order 40) shall be less than or equal to 8%.

**Table 11 – Upper limits of individual harmonic voltages**

| Odd harmonics      |                                 |                |                                 | Even harmonics |                                 |
|--------------------|---------------------------------|----------------|---------------------------------|----------------|---------------------------------|
| Not multiples of 3 |                                 | Multiples of 3 |                                 |                |                                 |
| Order $h$          | Relative voltage<br>( $U_h$ ) % | Order $h$      | Relative voltage<br>( $U_h$ ) % | Order $h$      | Relative voltage<br>( $U_h$ ) % |
| 5                  | 6,0                             | 3              | 5,0                             | 2              | 2,0                             |



|    |     |    |     |          |     |
|----|-----|----|-----|----------|-----|
| 7  | 5,0 | 9  | 1,5 | 4        | 1,0 |
| 11 | 3,5 | 15 | 0,5 | 6 ... 24 | 0,5 |
| 13 | 3,0 | 21 | 0,5 |          |     |
| 17 | 2,0 |    |     |          |     |
| 19 | 1,5 |    |     |          |     |
| 23 | 1,5 |    |     |          |     |
| 25 | 1,5 |    |     |          |     |

These limits for both individual harmonic voltages and total harmonic distortion of voltage are the same for low and medium voltages.

### **Unbalance**

The term Voltage Unbalance is referred to the condition when the three phase voltages are of different magnitudes and/or do not have a phase shift of  $120^\circ$  with respect to each other. The degree of this inequality is usually expressed as the ratios of the negative and zero sequence components to the positive sequence component. In the European Standard EN 50160, only the negative phase sequence is considered. Voltage unbalance is mainly caused by the use of single phase loads and incurs overheating in the equipment of both power system and consumer, reducing its lifetime. There are several approximations giving reasonably accurate results for the levels of unbalance, such as:

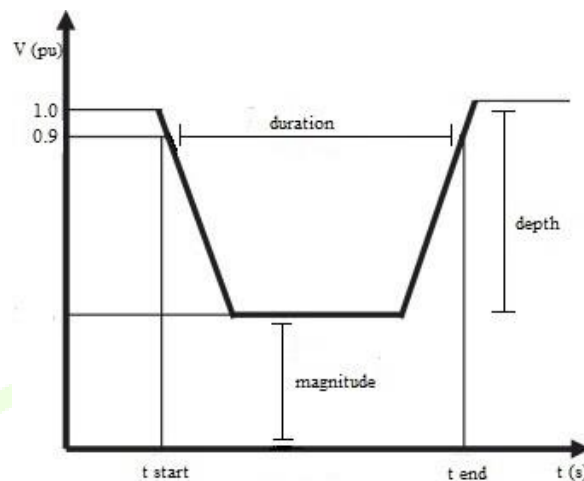
$$VUF = \sqrt{\frac{6 \cdot (U_{12}^2 + U_{23}^2 + U_{31}^2)}{(U_{12} + U_{23} + U_{31})^2}} - 2$$

where  $U_{12}$ ,  $U_{23}$ ,  $U_{31}$  are the three line-to-line voltages.

According to the European Standard EN 50160, under normal operating conditions, during each period of one week, 95% of the 10 min mean r.m.s. values of the negative phase sequence component (fundamental) of the supply voltage shall be within the range of 0% to 2% of the positive phase sequence (fundamental). In some areas with partly single phase or two phase loads, unbalances up to 3% at three phase supply terminals may occur. These limits are the same for both low and medium voltages

### **Voltage Dips**

The term supply Voltage Dip is referred to the sudden reduction of the supply voltage to a value between 90% and 1% of the declared voltage, followed by a voltage recovery after a short period of time. The duration of a voltage dip is considered to be between 10 ms and 1 min. The depth of a voltage dip is defined as the difference between the minimum r.m.s. voltage during the voltage dip and the declared voltage, as shown in Figure 28. When the supply voltage is not reduced under 90% of the declared voltage, then it is not considered as dip.



**Figure 28 – Characteristics of voltage dips.**

Voltage dips are unpredictable and quietly random events, caused generally by faults occurring in both customer's or public distribution network, but also as a result of connecting large loads, motor starting or transformer energizing. Voltage dips are the most critical power quality problem and can lead to substantial financial losses to industries due to the frequent disruptions to the industrial processes and malfunction of electrical equipment.

Under normal operating conditions, the annual expected number of voltage dips may be up to a few tens to up to one thousand. Usually, the voltage dips occurred last for less than 1 s and the voltage level is not reduced under 40%. However, occasionally, voltage dips with greater duration and depth can also occurred. Voltage dips with a retained level between 85% and 90% of the declared voltage can occur rather frequently, due to switching loads in customer's network. These statistical values are observed to be the same for both low and medium voltages.

A classification of voltage dips in respect to their duration and magnitude, commonly used, according to the European Standard EN 50160, is shown in Table 12.

**Table 12 – Classification of voltage dips**

| Voltage level<br>(%) | Duration (ms) |           |            |             |              |
|----------------------|---------------|-----------|------------|-------------|--------------|
|                      | 10<t<200      | 200<t<500 | 500<t<1000 | 1000<t<5000 | 5000<t<60000 |
| 90>u>80              |               |           |            |             |              |
| 80>u>70              |               |           |            |             |              |
| 70>u>40              |               |           |            |             |              |
| 40>u>5               |               |           |            |             |              |
| 5>u                  |               |           |            |             |              |

#### 5.2.4 Congestion Forecast

For such an application the main principle is concerning the steady state calculations for various input data.

Steady state calculations means a large mathematical model application on the grid topology, to find the voltages within nodes and active and reactive power transfer by the grid elements (lines and transformers).

The logic diagram would be as:

**A) Prepare grid data needed for the power flow** meaning to create the physical model of the grid as configured for the regimes to be evaluated.

- Retrieve from SCADA topology-related real-time **grid topology data** (topology connections, tap changers possibilities, etc) and **grid parameters data** (reactance, admittances for lines and transformers). These are data that define the grid structure and configuration for the regimes to be evaluated.
- Retrieve **nodal data**, that means to get the necessary data that shall model the specific operating electrical parameters that are voltages in the nodes and active and reactive power transferred within the grid elements (lines and transformers):
  - $P_g$  and  $U$  for all generators, including renewables, based on forecast of renewables; forecast will be based on next  $K$  hours, for instance next 24 hours. . These are based on the generation scheduled (for classical generators) and generation forecast for RES (as  $P_g$  and generating nodes voltages in module as desired for safety operation).
  - $P_c$  and  $Q_c$  for all loads, based on forecast of consumers; forecast will be based on next  $K$  hours, for instance next 24 hours. These are defined based on the forecast for consumption.
  - $U$  and  $\theta$  for the main supply point. . This is an algorithm designed node which calculates the balance necessary for an equilibrium between generation and consumption (including losses in the grid).

Identify main feeders (subjects of congestion) and compute main feeders' available capacity; These would be the possibly congested feeders and would also include the transformers as case.

**B) Compute steady state regime through power-flow algorithm** (run steady state regime for each forecasted scenario at  $N$  without  $N-1$ . That would mean at this stage we will only evaluate the grid operation conditions based on the complete grid structure (with no considered accidental outages due to unforeseen faults). For a further stage, the  $N-1$  criteria could be also developed, meaning that even some cases of outages (having a grid structure with  $-1$  element) could be considered.

### C) Assess congestions

Compare results of power flow calculations with determined available capacity for each main feeder within each forecasted scenario. Alarm capacity overload (congestions) based on agreed margins (settable)

Assess voltage levels. Alarm for over and under-voltage.

**D) Evaluate possible grid topology switching** (tap changers, capacitors, reactors, etc.) to fix alarmed congestions.

Simulate the operation such as switching devices (mitigation) for congestion fixing.

Run steady state regime for each of the previous forecasted scenario with mitigation switching.

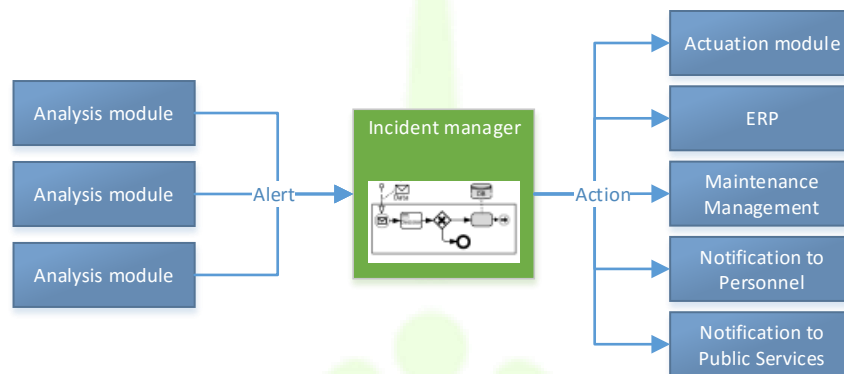
List capacity overload cases (for each main feeder within each evaluated scenario) grouped in  $N$  regime (normal grid status).

### 5.2.5 Grid Fault Management

Several modules are being investigate to help the DSO operator to increase the awareness of the status of the distribution grid and the occurrence of incidents requiring actions from their side. The relevant requirements those systems need to fulfil are, namely:

- In order to reduce the reaction time of the DSO upon the occurrence of an incident, the focus is put in systems that are able to perform real-time analysis of the different data flows initiated by the field devices (smart meters, power analyzers and sensors connected to the SCADA)
- Once an incident is detected by any of the different analysis modules, the list of the actions to be performed afterwards needs to be versatile and highly configurable. These actions may include triggering other modules responsible of applying automated solutions, register evidence of failure in a certain asset of the grid under a maintenance database, notify specific personnel or public services, publish information in a transparency portal of the DSO for public awareness, etc.
- Due to the variety of actions that may be taken and actors that may need to be informed of an incident on the grid, the alerting system will need to communicate using a variety of channels (web services for notification to automated systems, web, mobile phone messaging technologies, email, etc.)

Considering the previous requirements, a generic framework for incident processing is proposed to be demonstrated in the project to assist to the work of the DSOs.



**Figure 29 – Overview of an incident management system**

#### 5.2.5.1 Modules providing alerts

The following subsections describe particular modules that will be demonstrated by the project in order to increase the awareness on the status of the grid. Those modules subscribe to the flow of data coming from field devices and/or other modules in order to analyze those in real-time to trigger alerts when an unexpected behavior is detected.

##### 5.2.5.1.1 Threshold monitor

The purpose of the threshold monitor is to crosscheck in real-time the new voltage and power measurements coming from sensors in the grid with the corresponding acceptable limits (as specified in the topology of the grid), triggering an alert whenever any threshold is reached.



**Figure 30 – Monitored voltage in a low voltage bus**

#### 5.2.5.1.2 Outlier detector

The purpose of the outlier detector is to calculate in real time the average and standard deviation of electric parameters (voltage and frequency) on the different buses of the distribution grid. While this information itself is valuable to the DSO operator, since it provides insights of the quality of the energy supplied at different points of the grid over the time, it will also be used to implement a real-time *Control chart* with the objective of triggering early warnings to the DSO operator whenever these values deviate from their average (even without surpassing the acceptable thresholds).

A control chart is statistical process control tool to determine when a certain magnitude of a process becomes *abnormal*. Normality is statistically defined by the average and standard deviation of the samples measured, assuming those follow a normal distribution. The objective of this outlier detector is to analyse new measurements of voltage and frequency, being able to provide:

- Average values for voltage and frequency. This may be useful to identify persistent issues in certain points of the grid causing permanent deviations from nominal
- Standard deviations for voltage and frequency
- Alerts upon measurements of values out of *normality*, even if those values do not infringe the operational limits. This allows the DSO operator to identify deviations that may lead to problems (actual surpasses of the operational thresholds) in the future

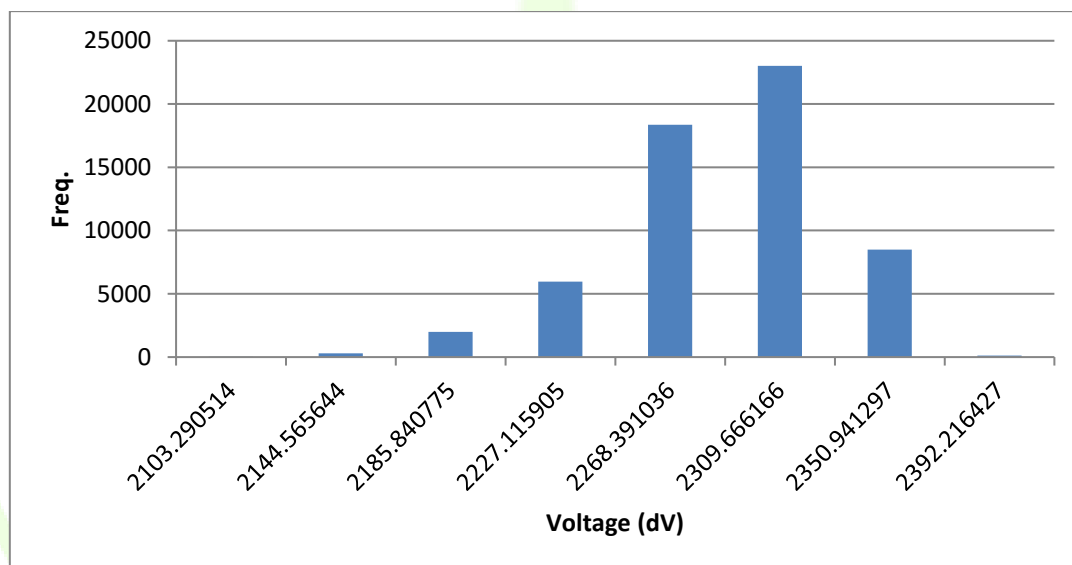


Figure 31 – Voltage distribution across 58000 samples in LV

#### 5.2.5.1.3 Congestion forecast

Already described in section 5.2.4, the purpose of this module is to evaluate the short-term future status of the grid, triggering an alert whenever a congestion issue is expected at a certain point of the grid.

#### 5.2.5.2 Incident management modules

The following subsections describe particular modules that will be demonstrated by the project in order to implement different actions upon the detection of an incidence on the grid.

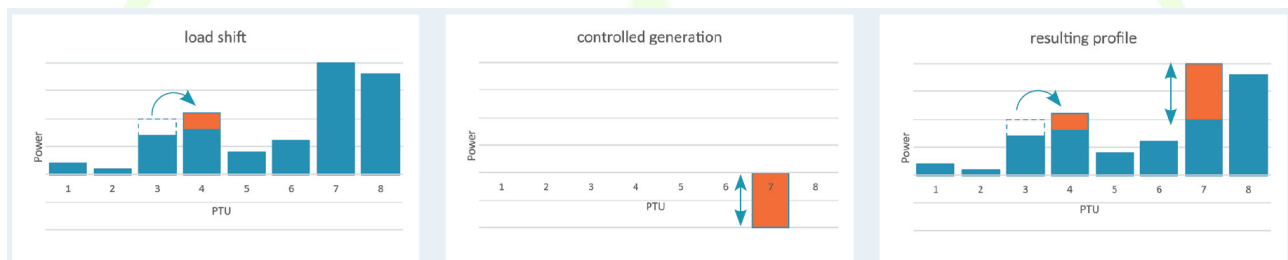
##### 5.2.5.2.1 Congestion support through Ancillary Services Market

One of the key points of the project is taking advantage of the variety of actors targeted by the different applications being developed (EV fleet managers, aggregation of prosumers, facility managers...) to

demonstrate how the use of their flexibility in the use of energy can be beneficial for assisting the DSO operator when facing congestion problems.

Within this context, the concept of Ancillary Services Market is put in place, managing the interactions of the different actors with flexibility with the DSO. One of the possible actions upon forecast of a congestion issue in the near future is to send a request to the Ancillary Services Market with the details of the problem (the main data being the period in time and surpass of power flow estimated over the acceptable threshold). Different actors participating in the market will be free to post offers for delivering the required flexibility, and DSO will analyze the received offers and accept the most convenient ones that solve the problem. The actual implementation of the management of the flexibility that is offered in this market is closely related to the business model and the typology of assets aggregated by each one of those participating actors.

The behavior of the Ancillary Services Market to be demonstrated in the project is based on the USEF specification [29]. Further details on the specifications of the market to be implemented and the algorithms behind the generation of the flexibility offers and the selection of flexibility requests can be found on D4.2 [4] and the design deliverables of the corresponding participant tools [30] [31] [32].



**Figure 32 – Example of usage of flexibility to change overall demand profile [29]**

#### 5.2.5.2.2 Assisted maintenance

Maintenance Management Systems are systems which can assist an organization to optimally manage the resources in charge of the maintenance of the assets (human resources, inventory, corrective and preventive maintenance). Upon detection of a problem in a device under control of the DSO, the maintenance manager can be automatically informed of the problem, thus making the proper personnel aware of the fail and assist to schedule the actions to be taken to solve it.

In section 2.2.3.1.1 it is possible to find further information of this maintenance capabilities.

#### 5.2.5.2.3 Fault Location, Isolation and Service Restoration

This module comprises two stages:

- At the first stage, the submodule named “Fault location” provides the following functionalities:
  - a) Verifies the occurrence of faults (functionality F1) and
  - b) provides the faulted line segments in the network (functionality F2) by combining the status information of the directional fault pass indicators (DFPIs), located in some points of the network, with the fault distance information coming from the fault distance relays, located generally at the substation.
- At the second stage, the submodule named “Service Restoration”, suggests an optimal reconfiguration of the topology of the network necessary to
  - a) isolate the fault (functionality F3) and
  - b) restore the service in the affected area (functionality F4).

Specifically, the algorithm provides the switching scheme indicating the recommended status of switching points or switching elements of the MV network for optimal operation according to predefined restoration criteria. It also provides the values obtained with respect to these optimization criteria, and the sequence of switching operations required to re-configure the network from the current (faulty) state to the optimized one.

This second submodule may also be used as an optimizer of the network configuration anytime, lowering energy losses and meeting current constraints when no faults are present in the network.

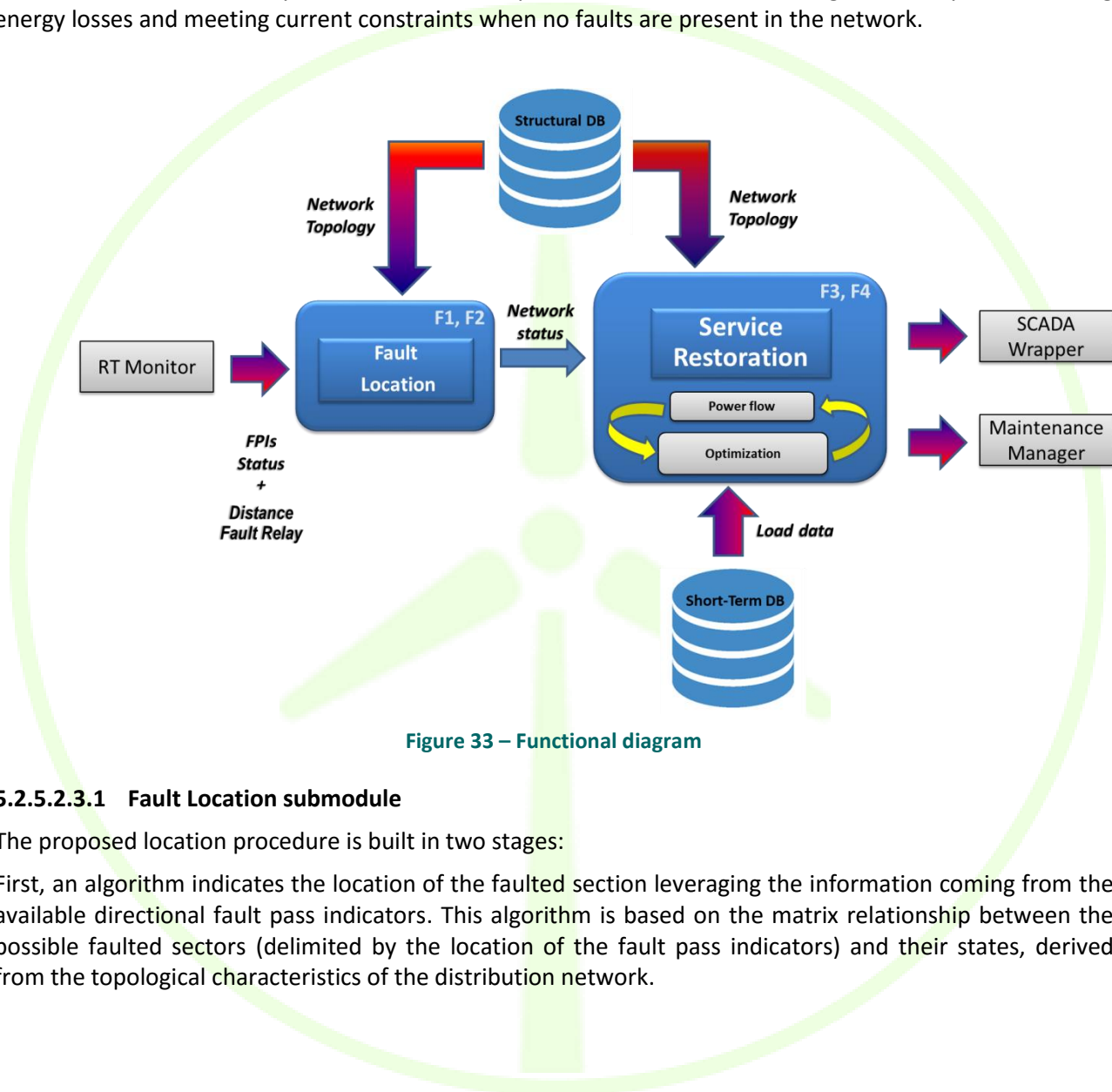


Figure 33 – Functional diagram

#### 5.2.5.2.3.1 Fault Location submodule

The proposed location procedure is built in two stages:

First, an algorithm indicates the location of the faulted section leveraging the information coming from the available directional fault pass indicators. This algorithm is based on the matrix relationship between the possible faulted sectors (delimited by the location of the fault pass indicators) and their states, derived from the topological characteristics of the distribution network.



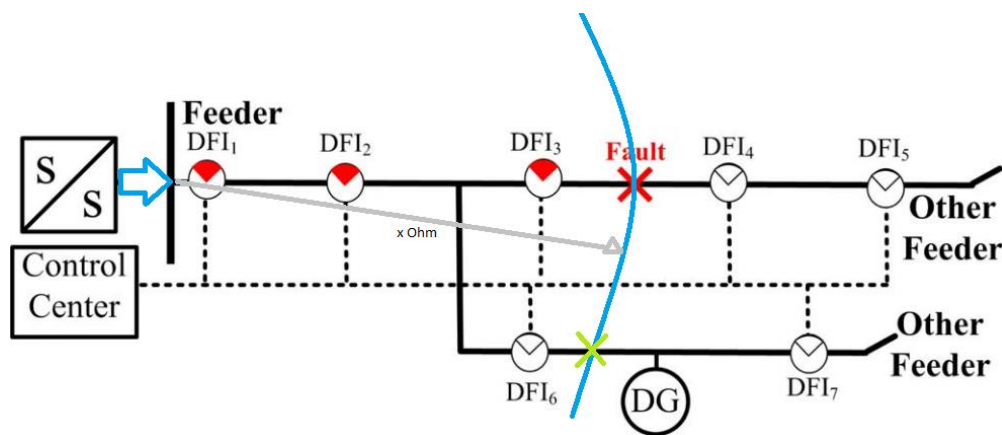


Figure 34 – Fault location principle. Source: [33]

Once the fault sector is located, the fault distance relays (when available) will determine the exact fault point location with an uncertainty interval according to the accuracy of the device. The point in the network whose ohmic distance matches the indicated fault distance within the faulted sector determined previously is the exact location of the fault. When there is not enough information from protective devices to determine this point, all possible candidates (line segments) are considered as faulty by this module.

#### 5.2.5.2.3.2 Service Restoration Submodule

The algorithm for optimal service restoration provides the switching scheme for fault isolation and optimal network reconfiguration, minimizing the total load out of service as well as other user-weighted operational indicators, while fulfilling also regular operational constraints.

The optimal network restoration is based on a set of operational criteria and constraints:

- Grid after the restoration shall be radial
- Voltage may be out of specified limits (configurable by user) in no more than  $n$  (configurable by user) nodes.
- Non restored load after network reconfiguration should be as low as possible.
- Total power losses after network reconfiguration should be as low as possible.
- Number of switches to be operated during restoration should be as low as possible.
- Relative node voltage deviation should be as low as possible
- Relative overload of lines should be as low as possible
- Load quadratic unbalance between network lines should be as low as possible

#### Operation

This algorithm performs sequentially the following set of steps:

1. Isolation of faulted segment/area so as to obtain a network free of elements affected by the fault.
2. Building of an initial set of feasible radial networks
3. GA genetic algorithm is launched with specialized operators to obtain an optimal solution according to operational criteria and constraints. Evaluation of each possible solution (both target function and constraints fulfilment) requires running a fast power flow module (BFS) with the demand scenario at the moment of the fault.

### **Problem formulation**

The general problem of optimally reconfiguring a distribution system can be formulated as:

$$\text{Min } F(G)$$

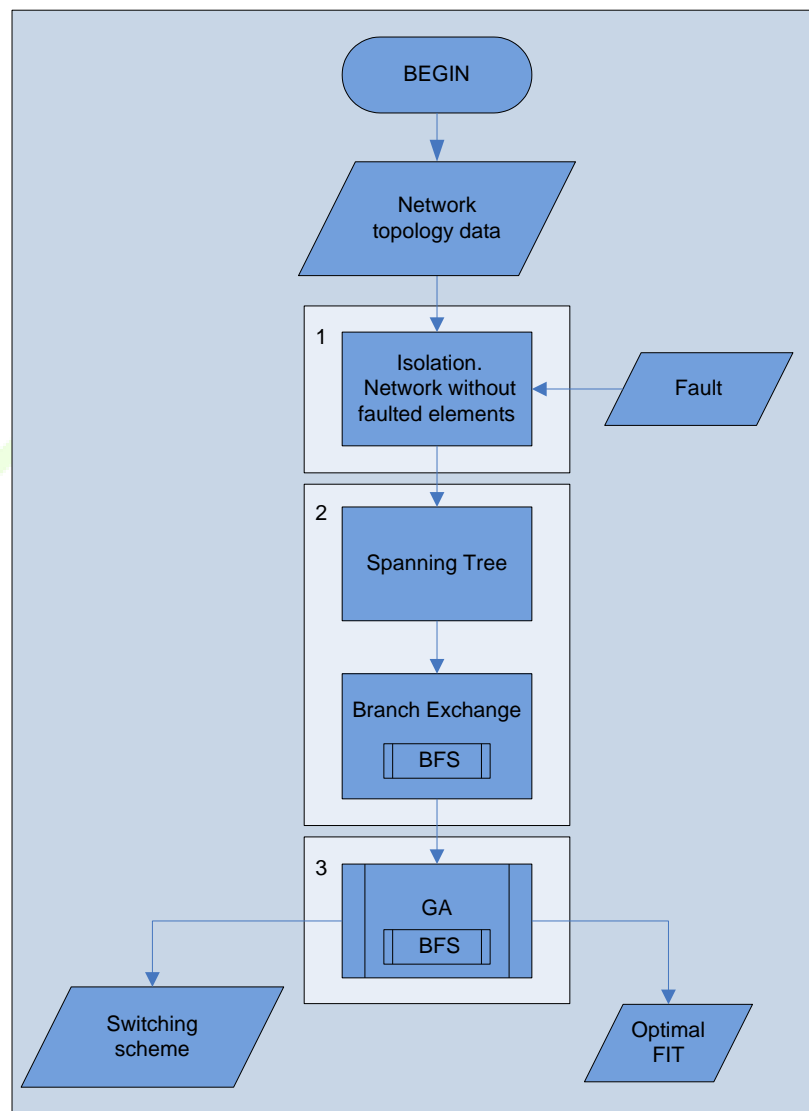
$$H(G) = 0$$

$$I(G) \leq 0$$

*G is a forest*

where:

- $G$  is the graph representing the distribution network configuration
- $F(G)$  is the objective function
- $H(G)$  the equality constraints representing the power flow equations
- $I(G)$  the inequality constraints representing the network operational constraints



**Figure 35 – Basic Service Restoration flowchart**

The target function to be minimized can be considered as a combination of several weighted objectives. With regard to constraints, these are managed through two strategies:

- A. Elimination of unfeasible individuals (deadly penalty function). It consists in assigning the maximum penalty to individuals who do not comply with some hard constraints:
  - a. Radiality of the resulting grid
  - b. Voltage violation in more than a limited number of nodes (configurable by the operator)
- B. Penalty functions. Soft constraints are introduced in the target function through penalty terms. This is the strategy adopted with most constraints.

Assuming the fulfilment of power flow equations, the target function to be minimized is considered as several weighted objectives  $F_x$  as well as the penalty of several soft constraints  $C_y$  and thereby it is expressed as follows:

$$FIT = \sum_x \alpha_x \cdot F_x + \sum_y \beta_y \cdot C_y$$

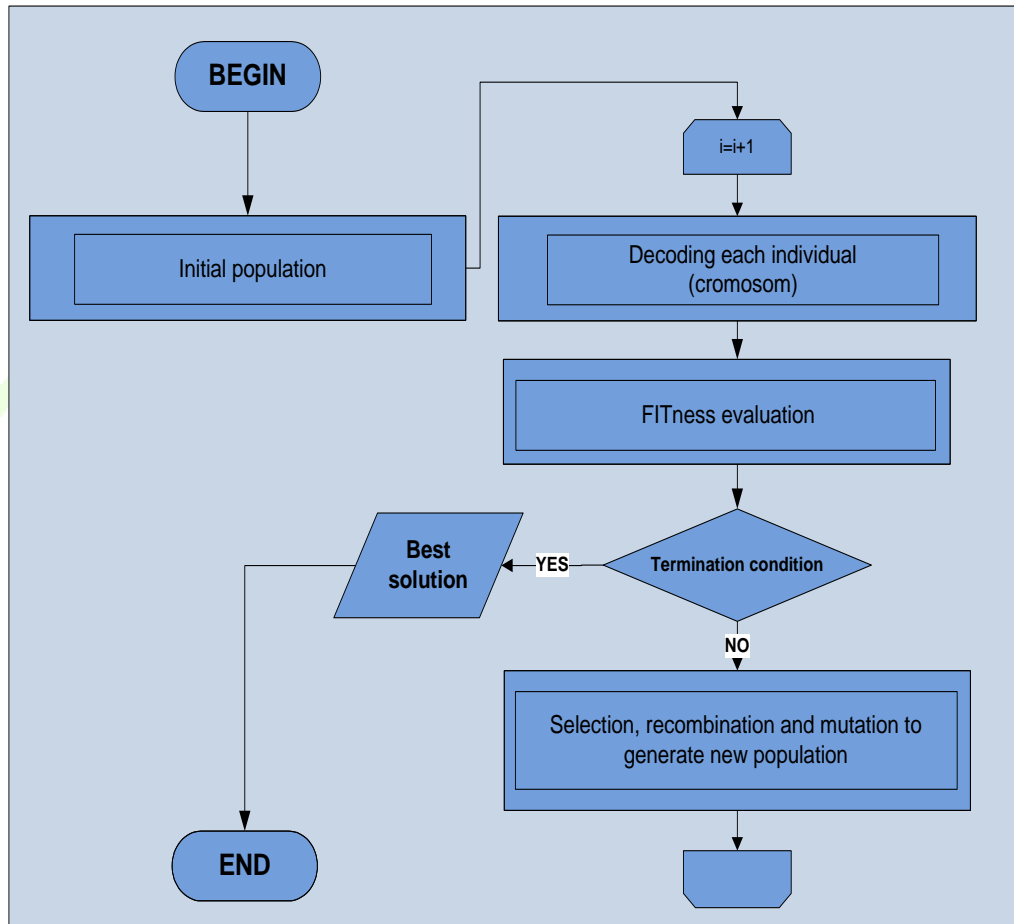


Figure 36 – Basic Genetic Algorithm flowchart

### Objectives

- Non restored load after network reconfiguration should be as low as possible.
- Total power losses after network reconfiguration should be as low as possible.
- Number of switches to be operated during restoration should be as low as possible.

### Soft constraints

- Relative node voltage deviation should be as low as possible
- Relative overload of lines should be as low as possible
- Load quadratic unbalance between network lines should be as low as possible

### 5.2.5.2.3.3 Fast Power Flow Submodule

The method implemented here is the backward /forward sweep (BFS), simpler and more efficient for the resolution of radial systems compared to Newton-Raphson methods.

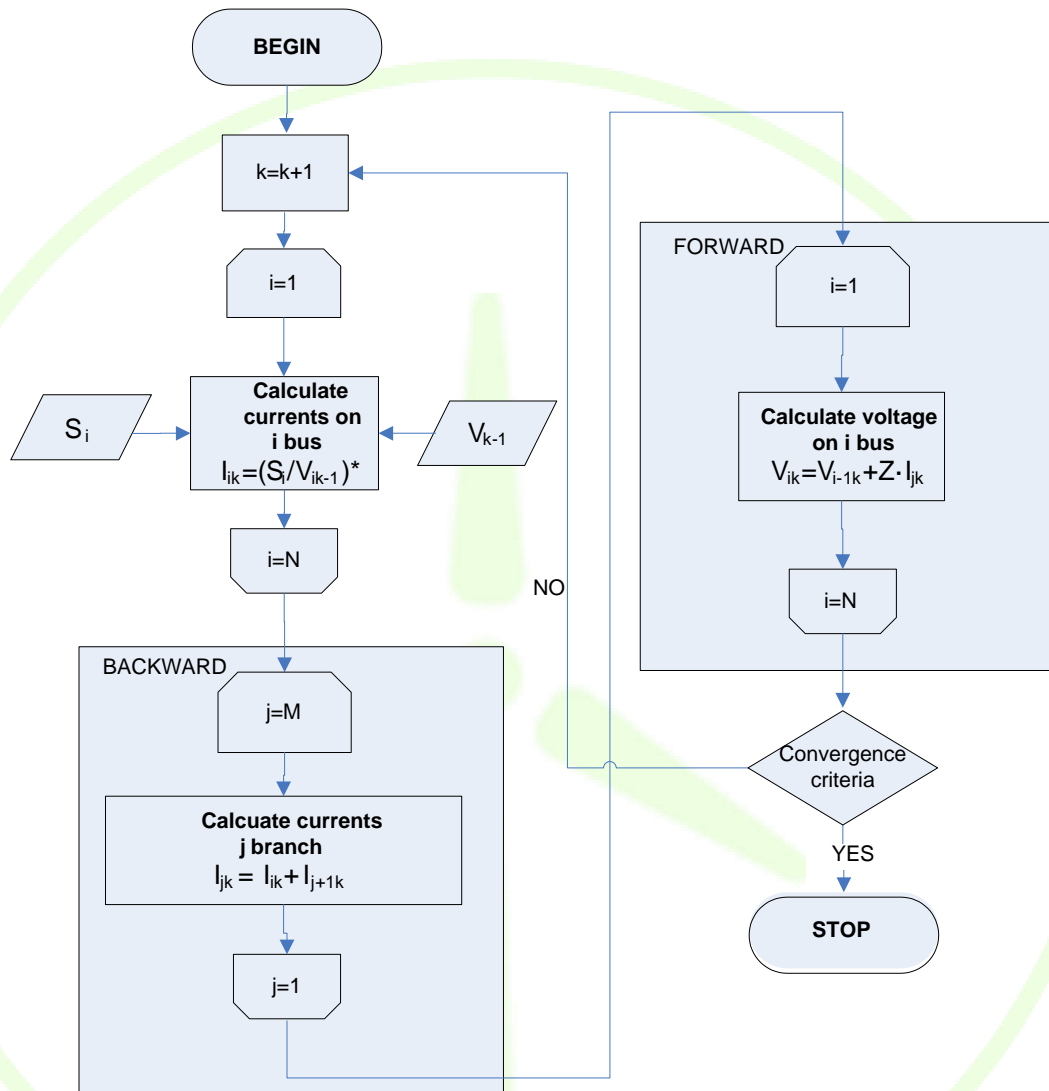


Figure 37 – Basic BFS flowchart

## 5.3 INTEGRATION OF DMS SERVICES TO SUPPORT DISTRIBUTION GRID AND SCADA OPERATION.

The core functionalities described at the previous section are the fundamental modules that will realise the advanced services of the Distribution Management System. Each of them requires a set of inputs stemming, mainly, from the various information systems of the grid operator. The input data can be,

generally, classified at the following categories:

- Real time or near real time data.  
This data include measurements from the metering infrastructure of the distribution grid and various types of alarms (outage detection, limit violation alarms etc).
- Historical and offline data.  
This data include the load profiles retrieved from the metering equipment and stored at a long term database, the history of the occurrence of various events, device status (open, close), the set of the electric system parameters and the grid topology.
- Spatial data.  
Data about the locations and shapes of grid geographic features and the relationships between them, stored as coordinates and topology.
- Other kind of data.  
Other data that are not included at the previous categories like the weather forecasting.

Most of the previously described data are collected from the various information systems operated by the DSO. More specifically, the SCADA (Supervisory Control and Data Acquisition) system can provide some of the real-time or near real time data, like voltages at various grid nodes (if these values are not directly collected from the respective metering infrastructure), device status, fault indications and fault currents. The spatial data is provided by the GIS (Geographic Information System). The AMI (Advanced Metering Infrastructure) provides the metering data retrieved by the metering infrastructure either for real-time use, for those that is possible, either for later use. For the latter case the data is aggregated and stored at a historical database. The rest of the data are provided either from external data sources, like the weather forecast or the spot price of the wholesale market, or they are inserted directly by the grid operator. An important issue that occurs, however, is the fact that, most commonly, all this data needs to be collected from individual and heterogeneous information systems, with different architectures and different data and information models. As a result, special interfaces - the wrappers - that facilitate the data transfer between each of the source systems and the DMS tool are necessary.

The described functionalities provide the results either directly to the user, through a GUI or a report, or to another functionality. In the former case, the results need to be displayed in a coherent and consistent way so that the grid operator will get the necessary information in order to proceed to the appropriate action. For the latter case, the data need to be appropriately processed in order to be possible to be handled. More information considering the interaction of the different functionalities and their integration can be found in deliverable D13.1 [14], where the architecture of the developed DMS tool (WG Cockpit) is described.

## 6 CONCLUSIONS

The present deliverable describes how the core business lines that could be achievable by a RESCO have been designed and implemented in the WG RESCO tool. The main objective is focused on end users who do not want to invest in a RES system but are willing to somehow have an active role in the new energy markets. With this objective, there are three options that have been considered for the RESCO:

- The RESCO pays a fee to the end user for using the whole produced energy.
- The production is shared between the customer and the RESCO.

- The production is consumed by end users and they pay a fee to the RESCO for maintenance tasks.

To achieve these business models, it is necessary to ensure some features. On one hand, the RESCO Tool must be able to provide a response according to the operator needs; on the other hand, the RESCO Tool must assure a user-friendly interface to facilitate the operation with the end users. Finally, an interoperability with other WiseGRID Tools by means of the WG IOP is necessary. This communication is essential for an optimal operation of the system.

The architectural decision for the WG RESCO tool has been described in Figure 1. This decision is not trivial, and the proposal is the most logical solution for solving the initial approach. The objective of this decision is to ensure a strong, secure and optimal communication among the different actors involved in the process. The architecture is organised in three big sections, plus a Big Data Platform. These sections are based in an acquisition module, communication mixer and processing module. The Data capture/provider is in charge of providing information from various sources to the other applications, and more specifically to the front-end section represented by WG RESCO Tool Web Application, which includes every application related with the WG RESCO tool. This information exchange is made through the WG IOP.

The used information employed in calculus processes or monitored in the single systems will be stored in a common cloud database called “wgresco”. Due to the technical requirements of the application, a database which makes an easy use of big information flows is needed for the correct operation of the WG RESCO tool. For this reason, a NoSQL database managed by MongoDB, an open source database manager will be the chosen one. This option allows an easy vertical and horizontal scalability, and it is able to acquire any technical restriction imposed by the application.

There are two main options for the deployment the application:

- Use a server machine
- Use a virtual machine

The last option is the selected one since it enables the deployment of the RESCO tool either as a host service or as an application running in a cloud service provider.

The second part of this report focus to describe the models and functionalities able to make the energy grid smarter than the traditional one.

The transition from passive grids to Active Distribution Networks has been accelerated, and it is accompanied with a better knowledge about the grid status requirements. This need has been solved by means a new methodology which combines load flow mathematics and statistical analysis in order to reduce measurement mistakes and replace bad data.

On the other hand, these concepts are strongly related with Energy Quality term, which is defined as the deviation of both voltage and current from their ideal, due to several factors (e.g.: increasing of the level penetration of electric vehicles). This deviation can be characterized and predicted by means the study of harmonics, voltage balance between three phases and voltage dips.

In order to improve grid status and ensure the best quality for the energy supply, the grid status related with constrains states is important to be foreseen by the tool. First of all, the tool requires some input data provided by SCADA topology-related systems and nodal information of each actor involved into the process (generators and consumers/prosumers). With this information, the steady state is calculated and analysed, obtaining susceptible congestion points. This allows to evaluate actions to reduce the congestions as much as it is possible. Consequently, the main functionalities considered more relevant are

- Demand and Production Forecast.
- Energy Quality Monitoring.
- Load Flow/State Estimation.



- Congestion Forecast.
- Outage Management – Fault Location, Isolation and Restoration

By using, analysing and evaluating the results of these core functionalities the grid operator will be assisted in the operation, maintenance and planning of the distribution system. These functionalities are the fundamental modules that will realise the advanced services of the Distribution Management System and that are further analysed in the WP13 inside the deliverable D13.1 [14] and implemented inside the WG Cockpit.

## 7 REFERENCES AND ACRONYMS

### 7.1 REFERENCES

- [1] [Online]. Available: <https://github.com/creationix/nvm>.
- [2] “D2.1 WiseGRID requirements, Use cases and pilot sites analysis”.
- [3] [Online]. Available: <http://nobelgrid.eu/>.
- [4] “D4.2 WiseGRID interoperable Integrated Process (WG IOP)”.
- [5] “D5.1 WiseGRID cloud-based big data infrastructure”.
- [6] [Online]. Available: <https://medium.freecodecamp.org>.
- [7] [Online]. Available: <http://meanjs.org/>.
- [8] [Online]. Available: <http://expressjs.com>.
- [9] [Online]. Available: <https://angularjs.org/>.
- [10] [Online]. Available: <https://nodejs.org/en>.
- [11] [Online]. Available: <https://nginx.org/en/>.
- [12] [Online]. Available: <https://getbootstrap.com/>.
- [13] [Online]. Available: <https://www.mongodb.com>.
- [14] “D13.1 WiseGRID Cockpit Design”.
- [15] X. Guo and J. Su, “Improved Support Vector Machine Short-term Power Load Forecast Model Based on Particle Swarm Optimization Parameters,” *Journal of Applied Sciences*, vol. 13, no. 9, pp. 1467-1472, 2013.
- [16] N. Hatziaargyriou, J. Amantequi, B. Andersen, M. Armstrong, P. Boss, B. Dalle, G. De Montravel, A. Negri, C. A. Nucci and P. Southwell, “Electricity Supply of the Future,” in *CIGRE WG “Network of the future”*, 2011.
- [17] M. Paolone, J. Y. Le Boudec, S. Sarri and L. Zanni, “Chapter 6: Static and Recursive PMU-based state

estimation processes for transmission and distribution power grids,” in *Advances in Power System Modeling, Control and Stability Analysis*, IET, 2016.

- [18] Ali Abur and Antonio Gómez Expósito, *Power System State Estimation - Theory and Implementation*, NewYork, NY: CRC Press, 2004.
- [19] J. J. Grainger and W. D. Stevenson, *Power System Analysis*, NewYork, NY: McGraw-Hill International Editions, 1994.
- [20] S. Sarri, *Methods and performance assessment of PMU-based real-time state estimation of active distribution networks*, PhD thesis, 2016.
- [21] N. Balu, T. Bertram, A. Bose, V. Brandwajn, G. Cauley, D. Curtice, A. Fouad, L. Fink, M. G. Lauby, B. Wollenberg and J. N. Wrubel, “Online power system security analysis,” *Proceedings of the IEEE*, vol. 80, no. 2, pp. 262-282, 1992.
- [22] Z. Huang, K. Schneider and J. Nieplocha, “Feasibility studies of applying kalman filter techniques to power system dynamic state estimation,” in *Inter. Power Eng. Conf. 2007 (IPEC 2007)*, Singapore, 2007.
- [23] E. Ghahremani and I. Kamwa, “Dynamic State Estimation in Power System by Applying the Extended Kalman Filter With Unknown Inputs to Phasor Measurements,” *IEEE Transactions on Power Systems*, vol. 26, no. 4, pp. 2556 - 2566, 2011.
- [24] K. Christakou, J. Y. LeBoudec, M. Paolone and D. C. Tomozei, “Efficient Computation of Sensitivity Coefficients of Node Voltages and Line Currents in Unbalanced Radial Electrical Distribution Networks,” *IEEE Transactions on Smart Grid*, vol. 4, no. 2, pp. 741 - 750, 2013.
- [25] K. Christakou, D. C. Tomozei, J. Y. LeBoudec and M. Paolone, “GECN: Primary Voltage Control for Active Distribution Networks via Real-Time Demand-Response,” *IEEE Transactions on Smart Grid*, vol. 5, no. 2, pp. 622 - 631, 2014.
- [26] M. E. Baran and A. W. Kelley, “State estimation for real-time monitoring of distribution systems,” *IEEE Transactions on Power Systems*, vol. 9, no. 3, pp. 1601-1609, 1994.
- [27] S. Sarri, L. Zanni, M. Popovic, J.-Y. Le Boudec and M. Paolone, “Performance Assessment of Linear State Estimators Using Synchrophasor Measurements,” *IEEE Transactions on Instrumentation and Measurement*, vol. 65, no. 3, pp. 535 - 548, 2016.
- [28] S. Sarri, M. Pignati, P. Romano, L. Zanni and M. Paolone, “A Hardware-in-the-Loop test platform for the performance assessment of a PMU-based Real-Time State Estimator for Active Distribution Networks,” in *PowerTech, 2015 IEEE Eindhoven*, Eindhoven, 2015.
- [29] “<https://www.usef.energy/>,” [Online].
- [30] “D6.2 Storage as a service and Innovative optimized solutions”.
- [31] “D7.1 WiseCOOP and WiseCORP Apps Design”.
- [32] “D9.1 WiseEVP Design”.
- [33] J. Teng, W. Huang and S. Luan, “Automatic and fast faulted line-section location method for distribution systems based on fault indicators,” *IEEE Transactions on Power Systems*, vol. 29, no. 4, pp. 1653-1662, 2014.

## 7.2 ACRONYMS

**Table 13 – List of Acronyms**

| Acronyms List |  |
|---------------|--|
| RES           | Renewable Energy Source                  |
| DMS           | Distribution Management System           |
| SCADA         | Supervisory Control And Data Acquisition |
| AMI           | Advanced Metering Infrastructure         |
| DSO           | Distribution System Operator             |
| PV            | Photovoltaic                             |
| USEF          | Universal Smart Energy Framework         |
| REST          | REpresentational State Transfer          |
| API           | Application programming interface        |
| ICT           | Information Communication Technology     |
| DB            | Data Base                                |
| GPS           | Global System Positioning                |
| ADN           | Active Distribution Network              |

## 8 ANNEX A

### 8.1 SWAGGER SPECIFICATION

```
{
  "swagger": "2.0",
  "info": {
    "description": "Maintenance Manager",
    "version": "1.2",
    "title": "maintenance_manager"
  },
  "tags": [{
    "name": "incident",
    "description": "Methods related to incidents"
  },
  {
    "name": "crew",
    "description": "Methods related to crew"
  },
  {
    "name": "device",
    "description": "Methods related to devices"
  }
  ],
  "paths": {
    "/api/incident": {
      "get": {
        "tags": ["incident"],
        "summary": "Get list of incidents",
        "produces": ["application/json"],
        "responses": {
          "200": {
            "description": "List of incidents (with details)",
            "schema": {
              "type": "array",
              "items": {
                "$ref": "#/definitions/Incident"
              }
            }
          }
        }
      },
      "post": {
        "tags": ["incident"],
        "summary": "Create new incident",
        "produces": ["application/json"],
```

```

"parameters": [{
  "in": "body",
  "name": "body",
  "description": "New incident details",
  "required": true,
  "schema": {
    "$ref": "#/definitions/NewIncident"
  }
}],
"responses": {
  "200": {
    "description": "Details of created incident",
    "schema": {
      "$ref": "#/definitions/Incident"
    }
  }
},
"/api/incident/{id}": {
  "get": {
    "tags": ["incident"],
    "summary": "Get incident data",
    "produces": ["application/json"],
    "parameters": [{
      "in": "path",
      "name": "id",
      "description": "Incident id",
      "required": true,
      "type": "string"
    }],
    "responses": {
      "200": {
        "description": "Incident details",
        "schema": {
          "$ref": "#/definitions/Incident"
        }
      }
    }
  },
  "post": {
    "tags": ["incident"],
    "summary": "Create incident",
    "produces": ["application/json"],
    "parameters": [{
      "in": "path",
      "name": "id",
      "description": "Incident id",
      "required": true,
      "type": "string"
    }],
    "responses": {
      "200": {
        "description": "Incident details",
        "schema": {
          "$ref": "#/definitions/Incident"
        }
      }
    }
  }
},
"/api/incident/{id}/tasks": {
  "get": {
    "tags": ["incident"],
    "summary": "Get incident tasks",
    "produces": ["application/json"],
    "parameters": [{
      "in": "path",

```

```

        "name": "id",
        "description": "Incident id",
        "required": true,
        "type": "string"
    }],
    "responses": {
        "200": {
            "description": "List of related tasks",
            "schema": {
                "type": "array",
                "items": {
                    "$ref": "#/definitions/Task"
                }
            }
        }
    }
},
"/api/device": {
    "get": {
        "tags": ["device"],
        "summary": "Get list of devices",
        "produces": ["application/json"],
        "parameters": [{
            "in": "query",
            "name": "type",
            "description": "Device type (for filtering results)",
            "required": false,
            "type": "string"
        },
        {
            "in": "query",
            "name": "id",
            "description": "Device id (for filtering results)",
            "required": false,
            "type": "string"
        }
    ],
    "responses": {
        "200": {
            "description": "List of devices",
            "schema": {
                "type": "array",
                "items": {
                    "$ref": "#/definitions/Device"
                }
            }
        }
    }
}
}

```

```

    },
    "post": {
      "tags": ["device"],
      "summary": "Register a new device",
      "produces": ["application/json"],
      "parameters": [{
        "in": "body",
        "name": "body",
        "description": "New device details",
        "required": true,
        "schema": {
          "$ref": "#/definitions/Device"
        }
      }],
      "responses": {
        "200": {
          "description": "Inserted device details",
          "schema": {
            "$ref": "#/definitions/Device"
          }
        }
      }
    },
    "/api/device/{type}": {
      "get": {
        "tags": ["device"],
        "summary": "Get list devices for a given type",
        "produces": ["application/json"],
        "parameters": [{
          "in": "path",
          "name": "type",
          "description": "Device type",
          "required": true,
          "type": "string"
        }],
        "responses": {
          "200": {
            "description": "List of devices of a given type",
            "schema": {
              "type": "array",
              "items": {
                "$ref": "#/definitions/Device"
              }
            }
          }
        }
      }
    }
  }
}

```



```

},
"/api/device/{type}/{id}": {
  "get": {
    "tags": ["device"],
    "summary": "Get list devices for a given type",
    "produces": ["application/json"],
    "parameters": [{
      "in": "path",
      "name": "type",
      "description": "Device type",
      "required": true,
      "type": "string"
    }, {
      "in": "path",
      "name": "id",
      "description": "Device id",
      "required": true,
      "type": "string"
    }],
    "responses": {
      "200": {
        "description": "List of devices",
        "schema": {
          "type": "array",
          "items": {
            "$ref": "#/definitions/Device"
          }
        }
      }
    }
  },
},
"/api/devicetypes": {
  "get": {
    "tags": ["device"],
    "summary": "Get list of defined device types",
    "produces": ["application/json"],
    "responses": {
      "200": {
        "description": "List of defined device types",
        "schema": {
          "type": "array",
          "items": {
            "$ref": "#/definitions/DeviceType"
          }
        }
      }
    }
  }
}

```

```

    }
  },
  "/api/crew": {
    "get": {
      "tags": ["crew"],
      "summary": "Get list of workers",
      "produces": ["application/json"],
      "responses": {
        "200": {
          "description": "List of workers",
          "schema": {
            "type": "array",
            "items": {
              "$ref": "#/definitions/Worker"
            }
          }
        }
      }
    },
    "/api/crew/{id}": {
      "get": {
        "tags": ["crew"],
        "summary": "Get details of a worker",
        "produces": ["application/json"],
        "parameters": [{
          "in": "path",
          "name": "id",
          "description": "Worker id",
          "required": true,
          "type": "string"
        }],
        "responses": {
          "200": {
            "description": "Worker details",
            "schema": {
              "$ref": "#/definitions/WorkerDetails"
            }
          }
        }
      },
      "/api/crew/{id}/incidents": {
        "get": {
          "tags": ["crew"],
          "summary": "Get list of incidents assigned to a worker",
          "produces": ["application/json"],
          "parameters": [{

```

```

        "in": "path",
        "name": "id",
        "description": "Worker id",
        "required": true,
        "type": "string"
    }],
    "responses": {
        "200": {
            "description": "List of UUIDs of assigned incidents",
            "schema": {
                "type": "array",
                "items": {
                    "type": "string"
                }
            }
        }
    }
},
"/api/crew/{id}/tasks": {
    "get": {
        "tags": ["crew"],
        "summary": "Get list of tasks posted by a worker",
        "produces": ["application/json"],
        "parameters": [{
            "in": "path",
            "name": "id",
            "description": "Worker id",
            "required": true,
            "type": "string"
        }],
        "responses": {
            "200": {
                "description": "Worker details",
                "schema": {
                    "$ref": "#/definitions/WorkerTask"
                }
            }
        }
    }
},
},
"definitions": {
    "Incident": {
        "type": "object",
        "properties": {
            "mRID": {
                "type": "string",

```

```

    "description": "UID of the incident"
  },
  "workId": {
    "type": "integer",
    "description": "Numeric ID"
  },
  "typeId": {
    "type": "integer",
    "description": "0=corrective, 1=preventive"
  },
  "type": {
    "type": "string",
    "description": "Type description"
  },
  "priority": {
    "type": "integer",
    "description": "Priority 0-100"
  },
  "stateId": {
    "type": "string",
    "description": "State code R=registered, N=notified, F=finished"
  },
  "state": {
    "type": "string",
    "description": "State description"
  },
  "startDate": {
    "type": "string",
    "description": "Start data"
  },
  "endDate": {
    "type": "string",
    "description": "End date"
  },
  "workerId": {
    "type": "string",
    "description": "Assigned worker id"
  },
  "workerName": {
    "type": "string",
    "description": "Assigned worker name"
  },
  "notificationDate": {
    "type": "string",
    "description": "Notification date"
  },
  "updateData": {
    "type": "string",

```

```

        "description": "Last update date"
    },
    "communicationDate": {
        "type": "string",
        "description": "Communication date TODO"
    },
    "deadlineDate": {
        "type": "string",
        "description": "Deadline"
    },
    "creatorId": {
        "type": "string",
        "description": "ID of the notifier of the incident"
    },
    "creator": {
        "type": "string",
        "description": "Description of the notifier of the incident (e.g. system, customer...)"
    },
    "assetId": {
        "type": "string",
        "description": "ID of asset"
    },
    "asset": {
        "type": "string",
        "description": "Name of asset"
    },
    "assetType": {
        "type": "string",
        "description": "Type of asset"
    },
    "incident": {
        "type": "string",
        "description": "Description of the incident"
    },
    "details": {
        "type": "string",
        "description": "Free text with further details"
    }
},
"NewIncident": {
    "type": "object",
    "properties": {
        "deviceType": {
            "type": "string",
            "description": "Device type code"
        },
        "deviceId": {

```

```

        "type": "integer",
        "description": "Device ID"
    },
    "typeId": {
        "type": "integer",
        "description": "0=corrective, 1=preventive"
    },
    "notification": {
        "type": "string",
        "description": "Notification code"
    }
}
},
"Task": {
    "type": "object",
    "properties": {
        "sortNum": {
            "type": "integer",
            "description": "Order of the task"
        },
        "date": {
            "type": "string",
            "description": "Date of task registration"
        },
        "status": {
            "type": "string",
            "description": "Status code of the incident after task N=notified R=registered F=finished"
        },
        "worker": {
            "type": "string",
            "description": "Name of the worker that performed the task"
        },
        "details": {
            "type": "string",
            "description": "Data associated to the message field"
        },
        "message": {
            "type": "string",
            "description": "Free text describing the task, including placeholders for details"
        }
    }
},
"Worker": {
    "type": "object",
    "properties": {
        "mRID": {
            "type": "string",
            "description": "UID of the worker"
        }
    }
}

```

```

    },
    "name": {
      "type": "string",
      "description": "Name of the worker"
    },
    "idCard": {
      "type": "string",
      "description": "ID of the worker (company code)"
    },
    "contractStartDate": {
      "type": "string",
      "description": "Initial date of contract"
    },
    "contractEndDate": {
      "type": "string",
      "description": "Final date of contract"
    },
    "work": {
      "type": "string",
      "description": "UID of the currently assigned incident (first one of the list)"
    }
  },
  "WorkerDetails": {
    "allOf": [
      {
        "$ref": "#/definitions/Worker",
        {
          "type": "object",
          "properties": {
            "mainAddress": {
              "type": "object",
              "properties": {
                "streetDetail": {
                  "type": "object",
                  "properties": {
                    "name": {
                      "type": "string",
                      "description": "Main address street"
                    }
                  }
                }
              }
            },
            "townDetail": {
              "type": "object",
              "properties": {
                "name": {
                  "type": "string",
                  "description": "Main address town"
                }
              }
            }
          }
        }
      ]
    }
  }
}

```



```

        "code": {
            "type" : "string",
            "description": "Main address ZIP code"
        },
        "country": {
            "type" : "string",
            "description": "Main address country"
        }
    }
}
}
}
}
}
},
"WorkerTask": {
    "allof": [
        {
            "type": "object",
            "properties": {
                "mRID": {
                    "type" : "string",
                    "description": "UID of the associated incident"
                }
            }
        },
        {
            "$ref": "#/definitions/Task"
        }
    ],
    "Device": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "description": "Type of device"
            },
            "id": {
                "type": "number",
                "description": "Id of device"
            },
            "description": {
                "type": "string",
                "description": "Description of device"
            },
            "latitude": {
                "type": "number",
                "description": "Location of device (latitude)"
            }
        }
    }
}

```

```
deviceType : {
  "type": "object",
  "properties": {
    "type": {
      "type": "string",
      "description": "Type of device"
    },
    "description": {
      "type": "string",
      "description": "Description of device"
    }
  }
}
```

## 8.2 CIM METERREADING MESSAGE EXAMPLE

```
{  
    "Meter" : {  
        "mRID" : "BBB5976",  
        "reason" : "",  
        "remark" : "",  
        "value" : "ok"  
    },  
    "Readings" : [{  
        "ReadingType" : "0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1",  
        "reportedDateTime" : "02/19/2018 08:45:10",  
        "timeStamp" : "2018-02-19T08:45:19.970Z",  
        "value" : "02/19/2018 08:45:20",  
        "ReadingQualities" : {  
            "comment" : "Current time"  
        }  
    }, {  
        "ReadingType"
```

```

"0.26.0.1.15.1.12.0.0.0.0.0.0.0.0.224.3.73.0",
    "reportedDateTime" : "02/19/2018 08:45:20",
    "timeStamp" : "2018-02-19T08:45:20.249Z",
    "value" : 0.3,
    "ReadingQualities" : {
        "comment" : "Reactive Energy QI - Total"
    }
}, {
    "ReadingType" :
"0.26.0.1.16.1.12.0.0.0.0.0.0.0.0.224.3.73.0",
    "reportedDateTime" : "02/19/2018 08:45:20",
    "timeStamp" : "2018-02-19T08:45:20.269Z",
    "value" : 0.3,
    "ReadingQualities" : {
        "comment" : "Reactive Energy QII - Total"
    }
}, {
    "ReadingType" :
"0.26.0.1.17.1.12.0.0.0.0.0.0.0.0.224.3.73.0",
    "reportedDateTime" : "02/19/2018 08:45:20",
    "timeStamp" : "2018-02-19T08:45:20.289Z",
    "value" : 0.3,
    "ReadingQualities" : {
        "comment" : "Reactive Energy QIII - Total"
    }
}, {
    "ReadingType" :
"0.0.0.12.0.1.15.0.0.0.0.0.0.0.0.224.0.33.0",
    "reportedDateTime" : "02/19/2018 08:45:20",
    "timeStamp" : "2018-02-19T08:45:20.480Z",
    "value" : 50.09,
    "ReadingQualities" : {
        "comment" : "Instantaneous net Frequency any phase"
    }
}, {
    "ReadingType" :
"0.0.0.0.0.1.54.0.0.0.0.0.0.0.0.128.0.29.0",
    "reportedDateTime" : "02/19/2018 08:45:20",
    "timeStamp" : "2018-02-19T08:45:20.500Z",
    "value" : 235.87,

```

```

        "ReadingQualities" : {
            "comment" : "Instantaneous voltage L1"
        }
    }, {
        "ReadingType" :
"0.26.0.1.19.1.12.0.0.0.0.0.0.0.0.224.3.73.0",
        "reportedDateTime" : "02/19/2018 08:45:20",
        "timeStamp" : "2018-02-19T08:45:20.558Z",
        "value" : 45680.878,
        "ReadingQualities" : {
            "comment" : "Reactive Energy - Total"
        }
    }, {
        "ReadingType" :
"0.26.0.1.1.1.12.0.0.0.0.0.0.0.224.3.72.0",
        "reportedDateTime" : "02/19/2018 08:45:10",
        "timeStamp" : "2018-02-19T08:45:20.569Z",
        "value" : 12354.467,
        "ReadingQualities" : {
            "comment" : "Active Energy + Total"
        }
    }, {
        "ReadingType" : "0.0.0.0.0.1.4.0.0.0.0.0.0.0.128.0.5.0",
        "reportedDateTime" : "02/19/2018 08:45:20",
        "timeStamp" : "2018-02-19T08:45:20.581Z",
        "value" : 5.35,
        "ReadingQualities" : {
            "comment" : "Instantaneous current L1"
        }
    }, {
        "ReadingType" :
"0.26.0.1.1.1.12.0.0.0.0.0.0.0.224.3.73.0",
        "reportedDateTime" : "02/19/2018 08:45:20",
        "timeStamp" : "2018-02-19T08:45:20.593Z",
        "value" : 34567.678,
        "ReadingQualities" : {
            "comment" : "Reactive Energy + Total"
        }
    }
}

```

```
]
}
```

