Rui Miguel Marques Pereira Trindade

Licenciado em Ciências da Engenharia
Eletrotécnica e de Computadores

Implementation of RESTful web-services as a platform for exchanging information between grid operators

Dissertação para obtenção do Grau de Mestre em

Engenharia Eletrotécnica e de Computadores

Orientador: Eng. Francisco Alexandre Ganho da Silva Reis, Full Professor/ Chief Engineer, & FCT UNL, REN, R&D NESTER

Júri

Presidente: Eng. João Almeida das Rosas
Arguente: Eng. João Francisco Alves Martins

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

September, 2019

Este documento foi gerado utilizando o processador (pdf)LaTeX, com base no template "novathesis" [1] desenvolvido no Dep. Informática da FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

If your hate could be turned into electricity,
it would light up the whole world

Nikola Tesla

Acknowledgements

There are several people that made this dissertation possible either in a direct or in an indirect way, by helping me with my work or giving me the motivation to keep on trying. Firstly I thank the Faculdade de Ciências e Tecnologias for allowing me to explore and learn about my area. I also thank my Thesis Adviser Eng. Francisco Reis for proposing the topic and allowing me to explore and develop this subject further. During the development of the thesis he supported me with all the information that I required in order to progress. Besides that he managed to find people that could help me in the challenges I had throughout the development of the thesis. I also need to thank the researchers that helped me with the development of both the API abd the Server, Eng. Nuno Amaro, Eng. Mohammed Al-saadi as well as other researchers and staff in R&D Nester.

I have in great regard the actions and the help I received from my colleagues and friends from Faculdade de Ciências e Tecnologia. They helped me in times of doubt and despair as well as in the good and great moments, not only during the thesis but also during our course in college. It would be remiss of me if I did not mention my Beloved Family and their constant and unconditional support. They might not have helped me directly in the execution of the thesis but they allowed me enjoy life and all its wonders. This comes from the bottom of my heart but all of you deserve a big hug that lasts for a good while. Now with this said... I might not be a doctor, but now I can fix televisions.

There are some friends that I need to thank for giving me laughs and headaches, these made me get new outtakes on life and life experiences. For my fellow scouts from the 1264-Falagueira group here is a thank you for all the camps and stories we share. For the volunteers and beneficiaries at Refood Benfica here is a thank you for all the laughs and the good times shared. As for my closest friends thank you for bearing with me and putting up with me and my temper as well as let there be more great times to come and experiences to share.

Abstract

This work addresses the information exchange between System Operator for operational planning purposes. These System Operators are responsible for the correct functioning of their electrical grid. With the increasing usage of renewable energy sources the coordination and information exchange between system operators must be improved. Amid the several potential areas of coordination between System Operators (TSO and DSO) the focus of this work relied on information exchange concerning the three phase Short circuit currents at the interface buses between TSO and DSO

In order to make this information exchange a web-service Architecture is proposed. The web-service Architecture used was the RESTful which needs to follow the European Union's specifications of data exchange. A structure for the type of information to be exchanged between Service Operators was developed and mapped to the Portuguese case, the information that was exchanged was the values of S_{cc} and I_{cc} of a given electrical grid (Short Circuit Power, Short Circuit Current) during a 24 hour period. An API (application programming interface) was developed that retrieves the information from a given grid and allows the required calculation.

To test the implementation of the solution developed, a IEEE 14 based grid was developed using profiles of generation and loads based on real case data from REN (*Redes Energéticas Nacionais*). To simulate a 24 hour scenario 24 different cases were developed and tested in order to get a profile of the evolution of S_{cc}. As for the RESTful Server it was tested with the API's output files to ensure the correct transmission of data between the API and the Server.

The results obtained on both the API and the Server are positive and reflected the expected outcome. The information retrieved from the API is validated, its output is well structured and contains the necessary information for the data exchange. As for the Server, the tests made were successful in testing the performance and fidelity of the information exchange and all developed functionalities.

Keywords: RESTful, Python, Short-Circuit, TSO, DSO, 504-Protocol, PSS/E

Resumo

Neste trabalho é abordado a troca de informação entre Operadores de Sistema, para alcançar um melhor planeamento operacional. A responsabilidade destes Operadores de Sistema baseia-se em zelar pelo correto funcionamento da sua rede elétrica. Com a maior produção de energia renováveis, têm de ser melhorados as vias de coordenação e a troca de informações entre os Operadores do Sistema. O foco deste trabalho foi a partilha de informações sobre correntes de curto-circuito trifásicas nos barramentos de interface entre os Operadores de Sistema (TSO e DSO).

Para a troca de informação entre Operadores de sistema ser realizada foi desenvolvido um *web-service*. O *web-service* desenvolvido tem como estructura a arquitectura RESTful, tendo por base as normas definidas pela União Europeia. Para o desenvolvimento da troca de informação entre Operadores de Serviço a estrutura do conteúdo da informação foi definida, e aplicada num caso baseado na realidade Portuguesa. O conteúdo da informação trocada é retirada dos barramentos de uma rede eléctrica, esta informação é referente a valores de corrente de curto-circuito trifásica simétrica para o horizonte temporal de 24 horas. Tendo para este efeito sido desenvolvido uma API (interface de programação de aplicativos).

Os testes feitos à API usaram uma rede baseada na IEEE 14, os valores usados para os perfis de geração e de cargas foram baseados em dados de casos reais da REN (Redes Energéticas Nacionais). Os cenários desenvolvidos corresponderam a discretizações de 1 hora no horizonte de 24 horas, serviu para obter um perfil de S_{cc} em cada barramento da rede. Após a correr a API os ficheiros resultantes foram utilizados para testar a troca de informação com o servidor RESTful.

Os testes realizados à API e ao servidor foram bem-sucedidos e os resultados retirados refletem o resultado esperado. Recorreu-se à validação dos valores retirados da API recorrendo a software próprio, com isto constatou-se que a informação retirada da API está valida e passível para a troca de dados. Em relação ao Servidor, todos os testes às funcionalidade do mesmo obtiveram a resposta esperada.

Palavras-chave: RESTful, Python, Curto-circuito, TSO, DSO, Protocolo-504, PSS/E

Contents

List of Figures

List of Tables

Listings

Glossary

Local Network	more commonly known as a local area network (LAN), it is a computer network of limited range (house, laboratory, school) that connects all the machines that are using it. These types of networks can be both wired or wireless, they allow information sharing among other functionalities..
Power Flow	numerical analysis of the flow of electric power in a given electrical system. A power-flow study focuses on various aspects of AC power parameters, such as voltages, real power and reactive power. It analyses the power systems in normal steady-state operation. [4].
Python	is a high level, object oriented programming language that was multiple uses and applications developed. The design philosophy of the python language emphasises code-readability, clear and logical code for small and large-scale projects. Possesses libraries that make it possible to create applications from web development to software development and scientific applications..
Server	is a computer program or a device that provides functionality for other programs or devices, called "clients". Servers possess several functionalities or services that can be used these range from data storage, remote computing and communication between clients. To a single server various users can accessed by many clients and the uses made to these machines are seen every where in this day and age..
Short Circuit	it is a phenomena that happens in electrical circuits when two nodes of different voltage values are directly connected to each other. When this happens an abnormal electrical current is generated limited by the Thevenin equivalent resistance, this current can cause damage to the elements present in the circuit.[4].

System Operator — is the legal responsible of the operation, maintenance and development of a given electrical grid system. Different System Operators have different responsibilities that are related to their own system and the others connected to it. These might be production, transmission or distribution of energy among other functions..

Thevenin theorem — under sinusoidal conditions, circuit-theory theorem stating that the electric current in a passive linear two-terminal network connected to any two terminals of a linear network is equal to the voltage between the two terminals before the connection divided by the sum of the impedance of the two-terminal network and the impedance of the network seen from the two terminals before the connection. [4].

Web-service — is a solution commonly used on the communication between two different application. Web-services allow applications to exchange data in order to establish a interaction between the application and the web-service and between two applications through the web-service. The interaction process between application and web-service needs to be in a format that is accepted by the web-service's own structure. .

Acronyms

ACER	Agency for the Cooperation of Energy Regulators.
API	Application Programming Interface.
ARM	Advanced RISC Machine.
BLE	Bluetooth Low Energy.
BUC	Business Use Cases.
CEER	Council of European Energy Regulators.
CPU	Central Processing Unit.
CRUD	Create Read Update Delete.
CSI	Camera Serial Interface.
CSS	Cascading Style Sheets.
DSI	Display Serial Interface.
DSO	Distribution Service Operator.
DTD	Document Type Definition.
ECACER	European Commission, Agency for the Cooperation of Energy Regulators.
EDPD	Energias de Portugal Distribuição.
EHV	Extreme High Voltage.
ENTSO-E	European Network of Transmission System Operators for Electricity.
EU	European Union.
FCR	Frequency Containment Reserves.
FRR	Frequency Restoration Reserves.
GPU	Graphics Processing Unit.
GUI	Graphical User Interface.

HTML	HyperText Markup Language.
HTTP	Hypertext Transfer Protocol.
HV	High Voltage.
ICC	Short Circuit Current.
ICCP	Inter-Control Center Communications Protocol.
IDE	Integrated Development Environment.
ie	id est.
IEC	International Electrotechnical Commission.
IEEE	Institute of Electrical and Electronics Engineers.
JSON	JavaScript Object Notation.
LPDDR	Low-Power Double Data Rate.
LV	Low Voltage.
MV	Medium Voltage.
OS	Operating System.
p.u.	Per unit.
PDIRT	Plano de Desenvolvimento e Investimento da Rede Nacional de Transporte.
PSS	Power System Simulation.
RAM	Random Access Memory.
REN	Rede Energéticas Nacionais.
RES	Renewable Energy Source.
REST	Representational State Transfer.
RR	Replacement Reserves.
RT	Real Time.
SCADA	Supervisory Control and Data Acquisition.
SCC	Short Circuit Apparent Power.
SGU	Significant Grid Users.
SOAP	Simple Object Access Protocol.

SOC	System on Chip.
SOG	System Operation Guideline.
SQL	Structured Query Language.
SUC	System Use Cases.
TSO	Transmission Service Operator.
UI	User Interface.
UK	United Kingdom.
URI	Uniform Resource Identifiers.
URL	Uniform Resource Locator.
USB	Universal Serial Bus.
UTF	Unicode Transformation Format.
VM	Virtual Machine.
XML	Extensible Markup Language.

Chapter 1

Introduction

> *The scientific man does not aim at an immediate result. He does not expect that his advanced ideas will be readily taken up. His work is like that of the planter — for the future. His duty is to lay the foundation for those who are to come, and point the way. He lives and labors and hopes.*

[Tesla, N. 1919. My Inventions: The Autobiography of Nikola Tesla]

1.1 Motivation

For electrical energy to reach our homes it has to travel through a vast grid that spans across whole country side. Due to legislative reasons the whole grid can not be owned by a single energy company, dividing it into transmission and distribution grid each one having different functions. The transmission grid connects the main energy generators (thermal and hydro generation) to the distribution grid and generally covers more territory. The distribution grid on the other hand, is directly connected to the everyday client, hence is located near energy consumption centres. The entities or companies that are responsible for these grids are called Transmission System Operator (TSO) and Distribution System Operator (DSO). In Portugal the TSO is REN (*Redes Energéticas Nacionais*) and the major DSO is EDPD (*Energias de Portugal Distribuição*). Communication between both providers is then key to ensure the correct grid operation and provide a stable service to their clients. Such communication, however is challenging especially when more and more types of generation sources are connected at lower voltage levels. This results in a lack of insight of TSO into DSO's gird activities and vice-versa. Therefore, a structured communication strategy needs to be put in place between both companies to ensure relevant information exchange is achieved, based on European approved standards.

To guarantee a safe energy delivery operation a platform for the sharing information needs to be put in effect. For the normalisation of this communication, European directives are defined (network codes and standards), especially made for the content and extent of the information to be shared by both sides.

The information to be exchanged is used in different time-frames (from real time to long-term planning). The main objective of this work, is to study, simulate and propose a solution to allow information exchange between system operators for operational planning purposes. Among the potential list of applications, the main focus of the work concerns the exchange of Short Circuit values, Current and Power (I_{cc} and S_{cc}), between system operators. This will allow both system operators to better coordinate operational actions with updated information.

Figure 1.1: High voltage Lines [1]

1.2 Objectives

The goal is to develop a framework in which information exchange concerning the S_{cc} is done. This is accomplished in several steps bearing in mind a forecast of topology, generation and load for the next 24 hours. Based on that the S_{cc} at a given node is calculated along with the contribution from neighbouring buses, from the High Voltage grids (HV, grids up to 60kV) and the Extreme High Voltage ones (EHV, grids of voltage of 150kV or above). To do this in an automatic way an Application Programming Interface (API) was developed and a dedicated information framework was implemented based on web-services. The information exchange was designed specifically for this application and a standardised format was used (XML). The output of this platform or an API has two possible file types, an Excel file and a XML file (Extensible Markup Language).The Excel file is created for ease of visualisation of the collected information. In this file the relevant information of the grid ran in the API is stored and includes:

The I_{cc}, the contributions from the the HV and the EHV grids and the Short Circuit Power (S_{cc}).

To make this exchange of information between service operators possible, a Server was developed. To create this server, the RESTful web-service Architecture was used as a basis. RESTful uses HTTP requests more specifically Get and Post. The implementation of this structure needs to be effective and secure enough to allow the flow of information without leaks or content corruption. The Server will be regarded as a third party that receives information from both TSO and DSO.

The basic structure of the system is as follows:

1. Run the prediction cases.

2. Calculate I_{cc}, I_{cc} contributions and S_{cc}.

3. Create Excel spreadsheet and XML file.

4. Send XML file to the Server.

5. Retrieve XML file from the server and manipulate the information.

The development of both the API and the Server are based on the programming language Python. It should be mentioned however that for the Server any language that can develop a RESTful architecture is valid. Since there are several Python libraries that have functionalities specifically for RESTful, Python was then chosen as the programming language.

1.2.1 Research Question

After clearly establishing the overall aim of the work and having the primary goal defined, the following research question can be made:

Can the implementation of a Web-service bring a more reliable and effective information exchange platform between system operators

This will be used as a guiding thread throughout the development process. The question will be answered by the platform implementation, assessment and validation.

1.2.2 Hypothesis

With the research question defined and research information a hypothesis can be postulated to infer the course of action. It has to encompass relevant development and context to the subject discussed.

Can a RESTful web-service be used as an effective and secure process of data exchange and prediction of Transmission and Distribution grids.

1.3 Dissertation Structure

This dissertation is divided in 6 chapters:

- **Chapter II - Technological Background.**

  In this chapter the basic foundations of the work developed are explored. It is divided into two separate sections, one regarding Grid Dynamics and the other the RESTful Web-services. In the former the focus is in the calculation of power flow and short circuit values and its correspondent theoretical analysis. For the RESTful Web-service section, the detailed description of the inner workings of the Architecture and examples of how it can be used.

- **Chapter III - TSO/DSO Cooperation Framework.**

  This chapter focuses on the European Framework on the relations between TSOs and DSOs and the data exchange between them. This refers to the type of exchanged information, the goals that the European Union faces and the current applied measures. Also on this chapter is highlighted the Portuguese standing on the implementation of solutions to facilitate the communication and control of TSO and DSO grids.

- **Chapter IV - Proposed Solution Implementation.**

  The developed implementation of both API and Server is explored in this chapter. First the dynamic between Client using the API and Server is detailed, and how this solution agrees with the current European Framework.It is then followed by an elaboration of each part of the implementation the Client's API and the RESTful Server. In the Client API section all details and functionalities used when this platform is used are explained. As for the Server section explores the web-service characteristics, main features and Server's used hardware.

- **Chapter V - Case Studies and Results.**

  Following theimplementation demonstration in previous Chapters here the testing, validation and results applied to the solution are detailed. The Client API required a set of case files in order to investigate whether the API returned acceptable results. For that reason a set of grids representing a 24 hour period was developed for testing, these were based on real data cases from the Portuguese system. As for the Server, the execution of the basic functions is specified, by highlighting a step-by-step approach on how to work with it. Furthermore, it also elaborates on the tests made to measure the Server's performance while performing the tasks mentioned earlier in the chapter.

- **Chapter VI - Conclusions.**

  After the development of the implementation, testing and validation, conclusions from the developed work can be drawn. For that, both Client API and Server are reviewed separately, highlighting all findings and relevant conclusions. Additionally, a section specifying future work on this implementation highlights the possible routes to further expand its uses and applicability.

Chapter 2

Technological Background

This chapter focus is on the basis of the whole dissertation, the basic concepts, examples and the methods of usage. Both for the Grid dynamics point and the RESTful web-service method of operation. In the grid dynamics this chapter will cross on the definition of the overall grid dynamics and the parameters that have influence over its own functioning. In specific the dynamics of the short circuit current and its importance in a grid's power flow and structural protection.

As for the RESTful web-service define its structure, method of operation and examples of usage. This will explain the origin of the architecture, creator and reason of creation. The method of operation will be explained with all the possible operation accessible by the web-service.

2.1 Grid Dynamics

2.1.1 Power Flow

To find the solution to the power flow of a grid it is considered the steady state condition as a basis. To be able to calculate the power flow of the grid the following steps are required:

1. Reach a valid mathematical model that accurately represents the system at hand. This includes load, generation and other elements of energy grids.

2. Define the type and characteristics of all the Bus bars in the grid.

3. With the previous data one can obtain:

    The argument and amplitude of the voltage in all the buses.

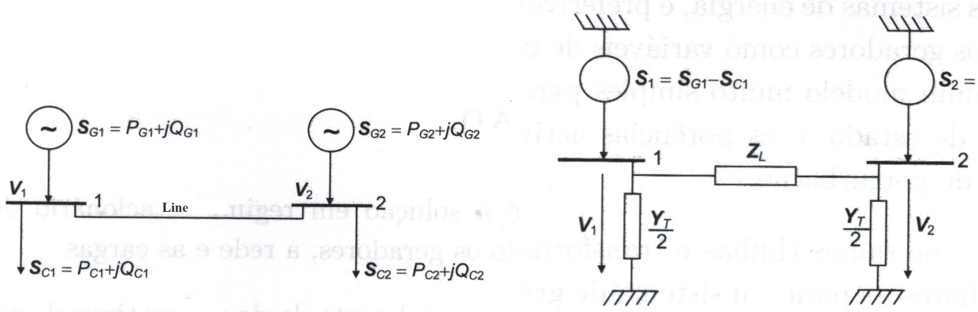
    The branch's power values (active, reactive)

2.1.1.1 Grid Analysis

Two Bus Grid

Consider the two bus grid shown in Figure 2.1. In the figure it can be seen two different representations, the One-line Diagram (Figure 2.1a) and the equivalent single-phase Diagram (Figure 2.1b) [2]. The One-line Diagram shows the generation, charges and lines discriminated. The equivalent represents the elements in a way that can be mapped to a numerical solution.

While the equivalent single-phase Diagram joins the generation and loads, resulting in the injected power, the line appears with characteristics that approximate it's overall functioning.



(a) Two Bus One-line Diagram

(b) Two Bus equivalent single-phase Diagram

Figure 2.1: Two Bus Network [2]

1. In each bus S_i represents the balance at each node between generation and load (2.1,2.2):

$$\mathbf{S}_1 = P_1 + jQ_1 = P_{G1} - P_{C1} + j(Q_{G1} - Q_{C1}) \tag{2.1}$$

$$\mathbf{S}_2 = P_2 + jQ_2 = P_{G2} - P_{C2} + j(Q_{G2} - Q_{C2}) \tag{2.2}$$

   With:

   - S_{G1}, P_{G1}, Q_{G1}-Generated Power in Bus 1
   - S_{C1}, P_{C1}, Q_{C1}-Consumed Power in Bus 1

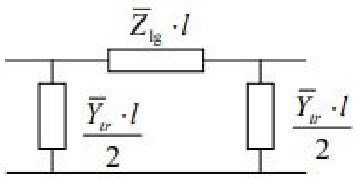
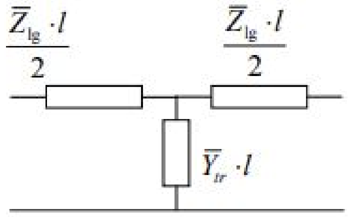
2. Bus Voltage (V_1, V_2) (V)

3. Line Impedance and Transverse Admittance (Z_L, Y_T) (Ω, S)

$$\mathbf{I} = \frac{\mathbf{S}^*}{\mathbf{V}^*} = \frac{P - jQ}{\mathbf{V}^*} \tag{2.3}$$

The model of the lines have many solutions, the most common are the following two, the Π and the T, as represented in the Table 2.1. In order to ease the analysis of the Π

Table 2.1: Line Models [3].

Π Approximation	T Approximation
	

approximation is best. In terms of error of the obtained value, it only reaches a significant value of error (>2%) when the line is of a length superior to 500km [3].

By using the equation 2.3 and applying the first of Kirchhoff's laws [4] it can result in the expressions of each buses (equations 2.4, 2.5). In the equations previously mentioned it can be defined all the admittance values, admittance between buses and the buses own admittance (equations 2.6,2.7). The values can be organized in matrix form, which allows to have a generalized analytic expression (equations 2.8, 2.9, 2.10) [2].

$$\mathbf{I}_1 = \frac{\mathbf{S}_1^*}{\mathbf{V}_1^*} = \frac{Y_T}{2}V_1 + \frac{1}{Z_L}(V_1 - V_2) \tag{2.4}$$

$$\mathbf{I}_2 = \frac{\mathbf{S}_2^*}{\mathbf{V}_2^*} = \frac{Y_T}{2}V_2 + \frac{1}{Z_L}(V_2 - V_1) \tag{2.5}$$

- y_{11}, y_{22} - It is composed by the resulting sum of all admittances connected to the bus.

$$\mathbf{y}_{11} = \mathbf{y}_{22} = \frac{\mathbf{Y}_T}{2} + \frac{1}{\mathbf{Z}_L} \tag{2.6}$$

- y_{12}, y_{21} - Between buses, defined by the symmetric value of the admittance in the direct connection between buses.

$$\mathbf{y}_{12} = \mathbf{y}_{21} = -\frac{1}{\mathbf{Z}_L} \tag{2.7}$$

Converting into Matrix form:

$$[\mathbf{Y}] = \begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{bmatrix} \tag{2.8}$$

$$[\mathbf{S}] = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \tag{2.9}$$

$$[\mathbf{V}] = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \tag{2.10}$$

Developing this matrix forms into the previous equations we can get a more complete version (equation 2.11):

$$\left[\frac{\mathbf{S}^*}{\mathbf{V}^*}\right] = [\mathbf{Y}]\,[\mathbf{V}] \tag{2.11}$$

N Bus Grid

Here there must be a generalisation of grid analysis, for any kind of grid. This grid results of any combination of elements, such as bus bars, lines, transformers and any other applicable element, within reason. To obtain the general expression that represents this new grid, the basis can be similar to the ones shown in equation 2.4 and 2.5. These equations allow us to reach the values of current crossing in a certain line. To change these expressions from the two bus grid reality to the new n bus grid one some changes need to be done. These being of nomenclature instead of 1 and 2 for buses, instead can be exchanged for a more generic variable. This allows any two buses that are connected through a line can be used in this formula. If a bus k is taken under consideration and the intended target is to get the expression that relates the apparent power, voltage and the admittance to any other bus l.

The equations 2.12, 2.13, 2.14 and 2.15 represent the extension of the equation 2.11 to a grid with n buses [2].

$$\frac{\mathbf{S}_k^*}{\mathbf{V}_k^*} = \sum_{l=1 k\neq l}^{n} \frac{Y_{Tm}}{2} V_k + \sum_{l=1 k\neq l}^{n} \frac{1}{Z_{Lm}}(V_k - V_l) = \sum_{l=1 k\neq l}^{n} (\frac{Y_{Tm}}{2} + \frac{1}{Z_{Lm}})V_k + \sum_{l=1 k\neq l}^{n} (-\frac{1}{Z_{Lm}})V_l \tag{2.12}$$

Where y_{kk} and y_{kl} are given by:

$$y_{kk} = \sum_{l=1 k\neq l}^{n} (\frac{Y_{Tm}}{2} + \frac{1}{Z_{Lm}}) \tag{2.13}$$

$$y_{kl} = (-\frac{1}{Z_{Lm}}) \tag{2.14}$$

$$\frac{\mathbf{S}_k^*}{\mathbf{V}_k^*} = y_{kk} V_k + \sum_{l=1 k\neq l}^{n} y_{kl} V_l = \sum_{l=1}^{n} y_{kl} V_l \tag{2.15}$$

The admittance matrix in this generalisation will be of a dimension of n × n:

$$[\mathbf{Y}] = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nn} \end{bmatrix} \tag{2.16}$$

2.1.1.2 Type of buses

Before starting any calculations of the power flow some specific characteristics need to be defined. These are regarding the buses and the information that is known about them.

There are several configurations, the most used are three, Reference bus, PQ buses and PV buses, the bus configurations and requirements are shown in table 2.2 [2].

There is only one bus in each grid that can be defined as Reference Bus, for this bus the voltage and argument of the voltage is defined. This bus is mostly associated to swing energy generation since both the active and reactive power generated can vary. For this reason a reference bus generates or absorbs energy if the grid possesses an excess or a deficit of said energy respectively[2].

When the generated active and reactive energy is known the bus is a PQ type. This category of buses is the exact opposite of the reference bus, since the attributed and calculated variables are switched. The most common of the bus categories, the PQ Buses constitute 80% to 90% of the total buses in a grid. They are a very versatile type because that can encumber the configuration of both load and generation[2].

Finally the PV buses are used when only the active power generation is known from the bus with this category. This types of buses are associated with generation buses, since the active power and voltage associated to it is defined. The active power the normal power output from the generation machines associated to the bus. As for the voltage it becomes a parameter that the user can control[2].

Table 2.2: Test results summary [2].

Type of Bus	Known Variables		Attributed Variables		Calculated Variables	
Reference Bus	Pc	Qc	V	θ	Pg	Qg
PQ (Charge or Generation)	Pc	Qc	Pg	Qg	V	θ
PV (Generation)	Pc	Qc	Pg	V	Qg	θ

After defining the grid's bus types comes the solving of the power flow. By starting with the equation 2.11 as a base and extending for all the n buses, the result is an equation system with n equations. However, the amount of variables that need to be solved is superior to n making it impossible to calculate without the aid of an mathematical model. These models use several iterations to get to the closer and most approximate value in the whole grid There are three main methods of reaching the equations values Gauss-Seidel, Newton-Raphson and fast decoupled. To start all of these methods require a prediction of the starting values, this is made to compensate the sheer amount of variables to calculate. This first iteration generates a second set of values and the process repeats itself. After each iteration the values are compared with the previous set throughout the iterations need to converge, in order to reach a conclusive answer. If there is no convergence there is no solution of the grid. The quality of the mathematical methods enunciated before, is measured by the speed that is capable of reaching a acceptable solution and it's reliability. This is important for grids with a larger scope and complexity, allowing a faster and accurate prediction solution [2].

To get the power that flows in the lines, there is only need of a given buses voltage and their line current (equation 2.17)[2]. Given by the following expression

$$S_{kl}^{line} = V_k \, (I_{kl})^* = \frac{1}{Z_{line}^*} \, (V_k - V_l^*) \, V_k \; + \; \frac{Y_T^*}{2} V_k^2 \tag{2.17}$$

2.1.2 Contingency Analysis

The test of resilience or contingency analysis, refers to a given grid's stability and its reaction to adverse effects. It is a crucial part of developing a resilient and safe grid [2]. Since the real live implementation of the grids is subject to external conditioning that puts the grid in an altered state of functioning, it is required to have this kind of analysis done. The test in question is what is called a n-1 criteria, in this test an element of the grid in question is removed from operation and the the power flow is calculated again. The criteria is made to ensure the safety standards of operation in any eventual scenarios. The scenarios covered by this criteria focus on situations where lines, transformers or other structural elements are put in an out of service status, for any given reason. In a real grid this can happen by accident (lines being cut, transformers blowing up, among others) or be planned by the grid operators (maintenance of lines, transformers and other elements present in the grid). The n-1 criteria is repeated for the crucial structural elements of the grid if not all of them. This criteria is more prominently used for TSO grids to ensure the quality of the service provided to the consumer [2].

In Figures 2.2 and 2.3 an example of the criteria is illustrated. For the example a grid with 3 buses is considered, using the characteristics of an EHV grid (220kV) these include the lines, loads and generation. The values used for the examples below were based on a three bus grid detailed in [2]. This is a theoretical example to prove the effect of the n-1 criteria in a grid. The Figure 2.2 represent the whole grid in a standard functioning state. In contrast, Figure 2.3 represents when the n-1 criteria is applied by removing the line between buses 1 and 2.

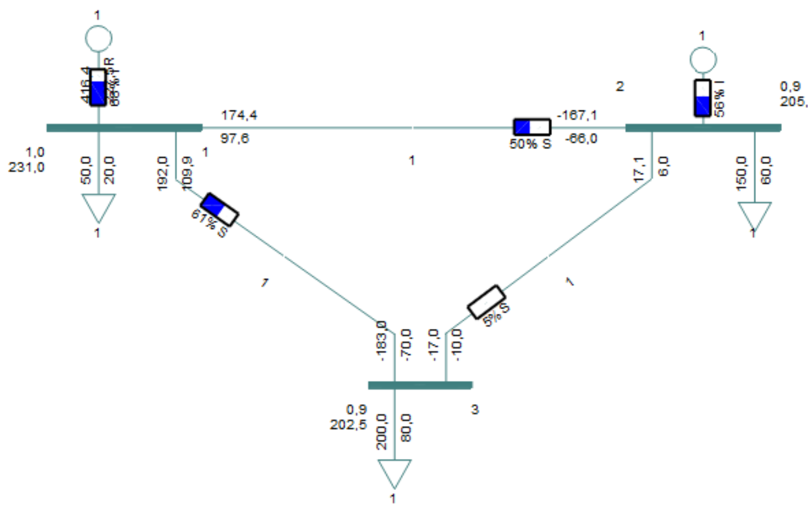


Figure 2.2: 3 Bus Grid

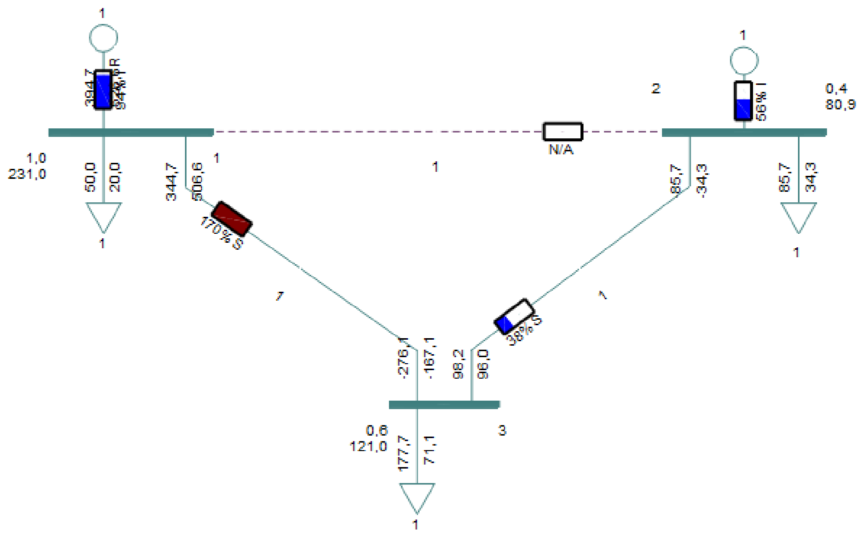


Figure 2.3: Grid with n-1 Criteria [2]

In the Figures above there are some visual indication on the power flow values and the grid's state. The blue bars represent the rate of usage of the lines and generators, and the numbers that accompany the lines are the active and reactive power that flows through each line. After removing one line, the power that passed through it is diverted to the remaining ones. This affects the rate of the neighbouring lines, that if not planned properly could overload and damage the grid [2]. In this case by testing the n-1 criteria the rate of the line that connects buses 1 and 3 registers an overload of 170% of the maximum rate. In this example an overload of this calibre can prove fatal to a grid [2]. This example was simulated with the aid of the Siemens PSSE software.

2.1.3 Short Circuit Calculation

Short circuits in these types of grids can prove to be very harmful, since they drastically alter the power flow in any point of said grid. This phenomena can cause severe damage to a grid for that reason it should be taken under consideration when planning. Even though this is reason enough to get this information the short circuit current and power can be useful when analysing bigger and complex grids [2].

The Short circuit current can be derived from several types of faults and with it have different values of current. The varying types of faults can be divided into four categories, Phase-Earth, Phase-Phase-Earth, Phase-Phase and Three Phase Fault. The one that can reach the highest values of current is the Three Phase Fault which is the main focus. The way to calculate this Fault current is shown in equation 2.18 Where:

1. The pre-fault voltage (V_k^0) at node k
2. The grid impedance (calculated by using the Thevenin Theorem Z_{T_k}).

The equation 2.18 allows the calculation of the current at a given bus k [2].

$$I_k^{cc} = \frac{V_k^0}{\sqrt{3}Z_{T_k}} \tag{2.18}$$

The short circuit power can be calculated like so:

$$S_k^{cc} = \sqrt{3}V_k^0\, I_k^{cc} = \frac{(V_k^0)^2}{Z_T} \tag{2.19}$$

With the equations of both the short circuit current and power defined, some simplifications can be used. The simplifications that will be used is to convert all the values in the equations from their original units to p.u.. P.u stands for per unit and it is a dimensionless unit that is obtained by dividing the actual value with a base value. For example in bus bar with a nominal voltage of 220kV and the present voltage in the bus is 231kV the p.u. value of the voltage is shown in equation 2.20.

$$V_pu = \frac{(V_k)}{V_{nom}} = \frac{231}{220} = 1,05p.u. \tag{2.20}$$

It serves as a sort of percentage when comparing with the nominal value. Using this simplification on the previously defined equations can show to be extremely helpful for their calculation. For example, if the nominal voltage is considered as the pre-fault voltage and the units were to be converted from SI to p.u., the simplification is displayed in 2.21.

$$S_k^{cc} = \frac{V_N^2}{Z_T} => S_k^{cc} = I_k^{cc} = \frac{1}{Z_T} \tag{2.21}$$

With this proves that the short circuit power and current in p.u are the inverse value of the impedance in the fault location. To get the grid's value of impedance it requires the conversion of all the structures into a equivalent impedance. Generators, transformers and lines have the impedance value associated to them, however, loads are normally considered a passive element (elasticity equal to 2) allowing it to be disregarded from the overall calculations. This is due to their characteristics such as the elasticity that dictates that it has constant impedance in function of the voltage. On the other hand active grids that are connected to the main grid and have an elasticity of 0 are considered for the short circuit calculations. This is due to their inherit characteristic of having constant power that does not change in spite of voltage swings. For these kinds of grids the equivalent impedance is calculated through either the short circuit Power or Current. After having either of those values the equivalent impedance can be calculated by using the expression reached in equation 2.21 [2]. To reiterate this expression only works when the units of all the parameters are in p.u..

The following example, taken from [2], analyses the following grid (Figure 2.4) for the three phase fault that happens in bus 3 [2]. The grid represented has a 60kV nominal voltage(V_{nom}) and a nominal power of 10MVA(S_n). The lines L1 and L2 have different lengths being 50 and 80 km, respectively, as for the transformer has a nominal power

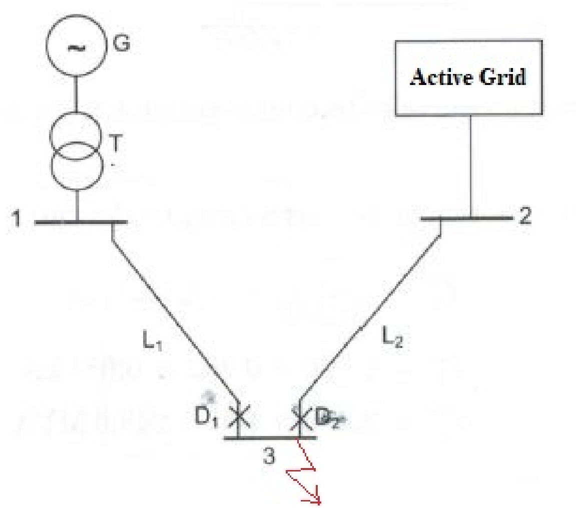


Figure 2.4: Three Bus Grid [2]

of 20 MVA (S_{nT}). The values shown in 2.22 belong to each of the structures in the grid that will also be used for the impedance calculation [2]. From top to bottom the values highlighted are:

$$\begin{cases} S_c c = 1000MVA, & Active\ Grid's\ Short\ Circuit\ Power \\ X''_d = 0,15p.u., & Generator's\ Transit\ Reactance \\ X_{Tcc} = 0,12p.u., & Transformer's\ Short\ CircuitReactance \\ X_L = 0,4\Omega/km, & Line's\ Reactance\ per\ km \end{cases} \tag{2.22}$$

To get the reactance of the generator and the transformer some adjustments are required, since they both are connected to a grid with a different nominal power. To get this new reactance value the reactance of the indivudual element is multiplied by the factor of the nominal power, as shown in equations 2.23 and 2.24 [2].

$$X_G = X''_d \frac{S_n}{S_{nT}} = 0,15\frac{10}{20} = 0,075p.u. \tag{2.23}$$

$$X_T = X_{Tcc} \frac{S_n}{S_{nT}} = 0,12\frac{10}{20} = 0,06p.u. \tag{2.24}$$

As for the line since the values given are not in p.u. they require a conversion. For that conversion a value of the base reactance needs to be calculated, this is achieved by dividing the nominal power of the grid with the square of the nominal voltage. This base value is then multiplied by the reactance of the line, as represented in equation 2.25 [2].

$$X_L = X_L Length \frac{S_n}{V_{nom}^2} <=> X_{L1} = 0,4\times 50\frac{10}{60^2} = 0,0556p.u., X_{L2} = 0,4\times 80\frac{10}{60^2} = 0,0889p.u. \tag{2.25}$$

Due to it being dependant of the nominal voltage of the grid the reactance can be smaller if a higher nominal voltage is used. This phenomena results in smaller reactances for higher voltage AC lines [2].

The active grid connected to the main one has an influence defined by its short circuit power. In order to get the impedance associated with it, the short circuit power is first converted to p.u. and then transformed into the impedance value. First to get the power value in p.u. the short circuit power from the active grid is divided by the nominal power of the main grid as the equation 2.26 shows. After that the inverse value of the power calculated is the impedance, and because it is a reactive impedance the value is imaginary (equation 2.27) [2].

$$S_{cc}(pu) = \frac{S_n}{S_{cc}} = \frac{1000}{10} = 100p.u. \tag{2.26}$$

$$Z_{cc} = j\frac{1}{S_{cc}(pu)} = j\frac{1}{100} = j0,01p.u. \tag{2.27}$$

As seen here the short circuit power of the bordering bus is the required parameter in order to measure the influence of a separate grid in the I_{SC} calculation. This simplifies the calculation of the influence by foreign grids in the short circuit current calculation [2].

With all the reactances calculated the grid can be decomposed into the reduced single line diagram(Figure 2.5). Applying the Thevenin theorem, to this new diagram, there is need to find an equivalent impedance in series with the voltage source in bus three with the pre fault voltage as its output. The simplification of the grid deals with the reactances as normal impedances and apllies the same reasoning as basic circuit theory. This simplification is demonstrated in equation 2.28 [2].

$$\begin{aligned} Z_{Thev} &= j(X_{L2} + X_{cc})//j(X_{L1} + X_T + X_G) \\ &= j(0,0889 + 0,01)//j(0,0556 + 0,06 + 0,075) = j0,0651p.u. \end{aligned} \tag{2.28}$$

This equation shows that with a higher number of branches in the grid the lower is the equivalent impedance. For that reason more meshed and complex grids have lower impedance than a radial counterpart [2].

Now to get the values of the Short circuit power and current it is applied the expression proven in 2.21 and multiplied by the grid's nominal values of both the power and current. This is shown in the following equations 2.29, 2.30 and 2.31 [2].

$$S_{cc}^3 = S_n\frac{1}{Z_{Thev}} = 10 \times \frac{1}{0,0651} = 153,6MVA \tag{2.29}$$

The value of the nominal current is defined by the nominal values of power and voltage, when the voltage is higher the lower is the base current. Which in turn result in smaller values of $I_{cc_{3\phi}}^3$ in higher voltage grids, making it different values of short circuit values for grid operators on different voltage values [2].

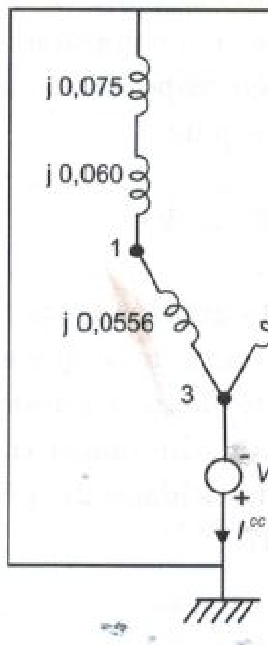


Figure 2.5: Reduced Single Phased Diagram [2]

$$I_n = \frac{S_n}{V_{nom}} = \frac{10 \times 10^6}{\sqrt{3}60 \times 10^3} = 96,225A \tag{2.30}$$

$$I^3_{cc_{3\phi}} = I_n \frac{1}{Z_{Thev}} = 96,225A \times \frac{1}{0,0651} = 1478,11A \tag{2.31}$$

Within the same grid the values of short circuit power and current in a bus can vary due to the different loads in the grid itself. This influence is not from the loads themselves since they are passive elements and are not considered for the impedance calculation. However, they influence the generation and the power flow of the grid which can result on less or more voltage in that bus. For that reason the short circuit values can mimic the load diagram . It is believed that with lower loads the short circuit current is also lower [2].

In conclusion in the short circuit analysis the smaller the impedance of the grid the higher its power and current are. In this example there were highlighted characteristics that can lower or increase the impedance value, such as the complexity of the grid, nominal voltage and the presence of neighbouring grids (belonging to another grid operator or a energy producer). The uses for these values are for making a choice in the calibre of the bus's protection against a short circuit. This information can also be used as way to relay the grid's influence with neighbouring grids, as also shown in the example [2]. This is a crucial step in the power flow analysis of several grids interconnected with each other, and an important basis for the information to share between TSOs, DSOs and SGUs (Significant Grid Users). The Figure 2.6 is a diagram that illustrates that different grids in real examples are next to one another. As seen in this example the presence of neighbouring active grids has influence in the functioning grid's power flow and the short circuit values [2].

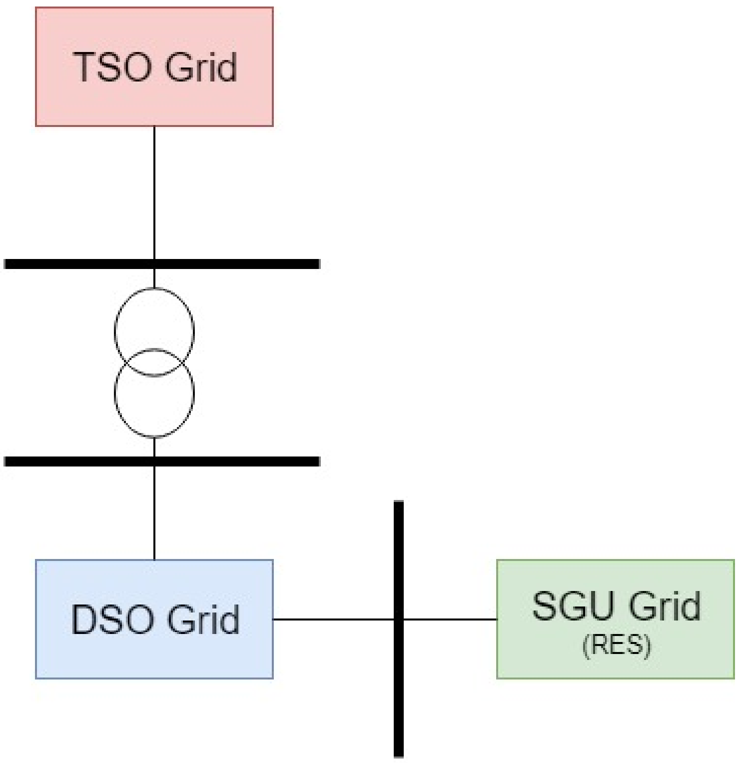


Figure 2.6: System operators interconnection Diagram [2]

2.2 RESTful Web service, applications and optimization

2.2.1 Overview

REST or Representational State Transfer, is a web-service framework or architecture that allows connections through the internet or any network. It was first defined by Dr. Roy Fielding, the REST's architecture style can be described as a "distributed hyper-media application". This means that when compared with other web-service architectures it has some differences. In REST as in other web-services establishes a connection between a Server and a Client, what differs is the way the information is categorised as well as the Client's session information. In REST the information is categorised as a form of hyper-media in the used browser, allowing execution of requests faster than other architectures. Due to a uniform structure in the request method, the client's state does not need to be sent. This allows for a better privacy, protection and performance of the web-service, since it does not need to send the state separately [5].

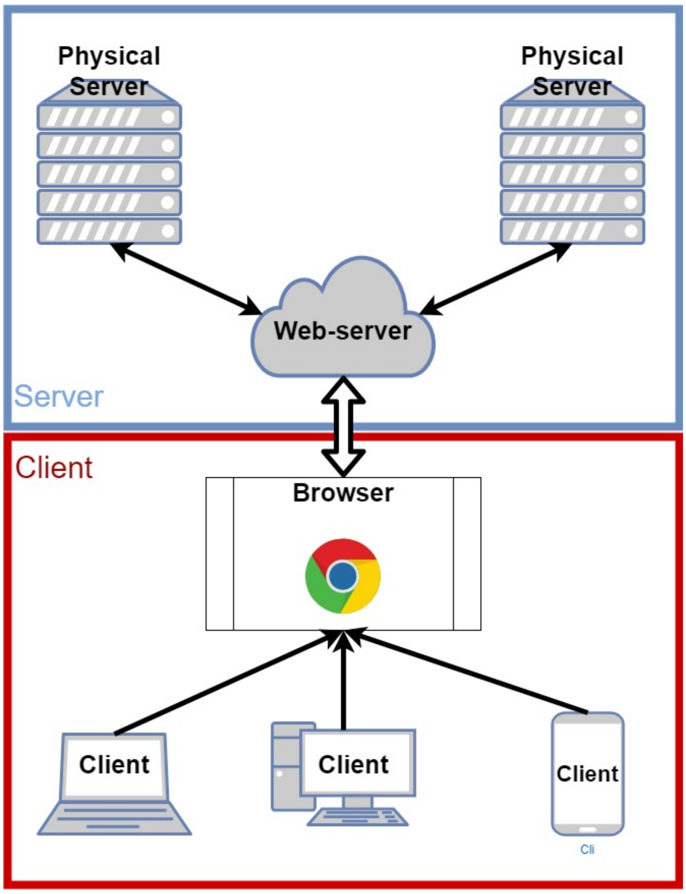


Figure 2.7: REST basic structure [5]

The Figure 2.7 above gives a sense on the basic structure and interactions of the REST framework. This framework is divided in two components, as mentioned before, the Client and the Server. The Client constitutes any user that tries to contact with the Server, and the Server both the hardware and software that processes and responds to the Client's requests. The Client establishes contact with the server through a browser of his/her choice. Depending on the request made by the Client the browser establishes connection with the Server by means of the web-server. This web-server processes the request received and reacts accordingly, this means returning a message, accessing another web-page or interacting with the physical database. These interactions are fully dictated by the web-server acting as the brains of the Server[5].

The REST architecture can result in the creation of new promising and pioneering applications. It allows for a easier production and deployment of services. By making every element/resource available, the RESTful service allows representation in different environments. Not just for software applications but also consumer electronics and wearable technology among others [6].

Although neither a standard nor a technology, REST is a framework style defined by a series of constraints. All REST web services that follow the constraints can be called a RESTful web service [5]. The six constraints that define a RESTful web-service Architecture are the following:

1. **Client-server interaction**

   There is a split between the Client or user interface and the server. The client can be developed separately from the server, the server only acts when called upon by a request-response [6],[5].

2. **Stateless**

   The server does not store the state of the client, the client possesses the state of the application. Allowing to be up-scaled and the possibility of concurrent computing with ease, since it does not require to manage the state information [6],[5].

3. **Cache usage**

   The contents referred to a request made to the server are present in the communication chain. In a way preventing the client repeating the same request, reducing the system inertia and its load rate, increasing the overall productivity [6],[5].

4. **URI Structure**

   The URI or Uniform Resource Identifiers is the standardized structure that RESTful uses to defined the requests sent. It allows a easier interaction with the server improving its visibility [6],[5].

5. **Layered system**

   This architecture can be segregated into independent layers, each one with a specific purpose. Therefore a single layer only interacts with those in its vicinity.this layered system, acts in a predictable way and also reduces the risk of coupling functionality [6],[5].

6. **On-demand code**

   To allow a better interaction with the system, the client can actively download and execute code. This can be favorable because it allows to add new application codes giving a great expansive and evolutionary capabilities [6],[5].

2.2.2 Data Exchange

The RESTful web service data exchange uses a request and response approach, using HTTP requests. These have their equivalent as RESTful requests, POST/CREATE, GET/READ, PUT/UPDATE, DELETE, or more commonly known as CRUD [5]. This request is sent to the server using the URI which when it reaches there identifies the requested entity/resource. This could be a Image, Text, XML, JSON file depending on the request or scope sent by the client [7].

The data stored in the REST Database can be organised with different layers [8]. For example a collection can have appended several resources, this means that a RESTful request has different responses according to the scope attributed [9]. In the following a more detailed description of each of these requests is given:

1. CREATE/POST - as the name implies it creates a resource or a collection. Returns a affirmative or negative response derived on if it has able to succeed or not [9].

2. READ/GET - retrieves a resource or collection from the Database. Returns the file/files if successful or a negative response if unsuccessful [9].

3. UPDATE/PUT - changes the information already in the Database. If successful returns an affirmative response, if there is no file in the Database returns a negative response [9].

4. DELETE - removes resource or collection from the Database. If successful returns an affirmative response, if there is no file in the Database returns a negative response. In the case of a collection it also deletes all the resources appended to it [9].

The Figure 2.8 illustrates the data exchange process in the RESTful web-service. Starting in the browser comes the HTTP requests, which were detailed prior, that reach the web-server. The web-server reacts according to the request given, which can be a web-page (HTML, Hyper-Text Markup Language), a file (XML,JSON, among others) or just a standard response code (404, 200, 500). When interacting the files or structures that take physical space, the web-server communicates with the database. The web-server proceeds by executing the function requested by the client and returning the appropriate response. The RESTful has some limitations in terms of the files that can be exchanged, however, that can be circumvented. There are also some limitations in size of the files but that seems to vary from implementation to implementation. Another limitation comes in the form of the maximum size of the URL (Uniform Resource Locator), being less than

2000 characters long for a request made [5] [9]. However, when compared with other web-services, such as SOAP (Simple Object Access Protocol), REST is easier to work with, more versatile in terms of file format to be exchanged and Closer to other Web technologies in design philosophy. This is because SOAP is slower, can only work with XML files and is mostly used with older technologies that do not work well on the Internet [9].

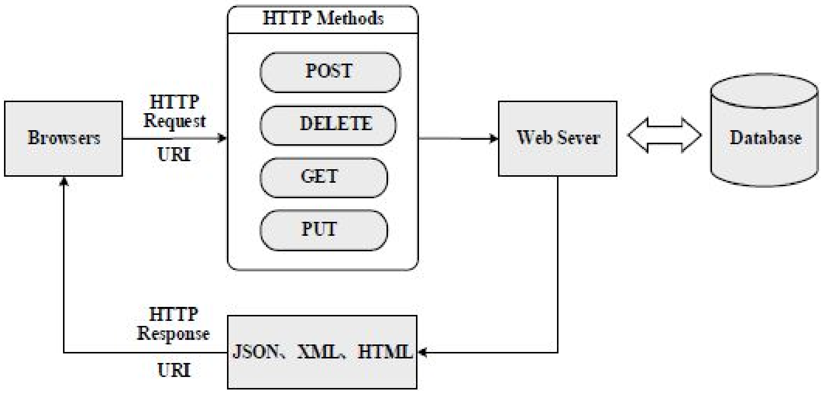


Figure 2.8: RESTful response-request [5]

2.2.3 Protection and Encryption

Due to its inherit structure the RESTful web-service is reliable in terms of information and user protection. This happens because of the stateless approach and the information processing being done in the Client's machine. Although it can be safe according to its framework other precautions can be taken to further ensure the security [5].

Now because it is a internet service it requires a amount of security in its operations, while maintaining a lightweight implementation. There could be associated a encryption signature to each message. So if an unauthorised entity tries to access the server without using this signature it will not be processed [10]. This makes a attempt of spoofing a message to the server harder to be pulled off.

2.2.4 Real Life Applications

Due to its lightweight structure and versatility there are great potential for the real life implementations. These implementations range from websites, in standard consumer electronic, wearable technology, among others. The web-sites use are the basic structure in terms of implementation of the web-service. As for consumer electronics if they have wi-fi capabilities they can be programmed to be interacted remotely by a user through the web-service platform, and allow the user to control the connected appliances [6].

Chapter 3

TSO/DSO Cooperation Framework

To ensure that the energy produced by power plants reaches the consumer, it has to pass through different grid owners. Each grid is operated by a different entity, the more prominent and important are the TSO and DSO. These two, throughout Europe, have different responsibilities depending on the country. TSO manages the transmission grid and others that need to be guaranteed the balancing of the system, as for the DSO the distribution grid, both ensuring its delivery with high level of quality of service. These two are the more prominent, however, other grids can connect to these bigger ones. Grids that do this and their operation affect the TSO's and DSO's power flow are denominated as SGUs, Significant Grid Users. The SGUs are more prominently energy producers that do not belong to either the TSO or the DSO [11].

The TSO and DSO must ensure a economic feasibility and efficiency in order to contribute to the European energy market. The TSO and DSO have make sure that the social welfare maximisation with a fair cost, also their data is handled with transparency and confidentiality [11].

This chapter will present the main research papers and documentation, regarding the cooperation of TSO/DSO, to be utilised during the development of the thesis. For that reason the main focus of the research will be in the operation between TSO and DSO, the regulations and European legislation. This will build upon the background specified in Chapter 2 bringing it closer to the main issue of the thesis. To start the European reality of the energy landscape and legislation will be presented, explaining the types of information to exchange and the goals of the exchange implementation. After that there will be a focus on the Portuguese framework since this case will be relevant for the implementation of the solution.

3.1 European TSO/DSO directives

3.1.1 European Energy Landscape

As the energy production moves to more sustainable systems, like renewable energy, the planning and operation of the energy grid changes. This transition has a high impact on the TSOs and DSOs since there are more and more energy producers injecting power to the grid [12]. With solar, wind and other type of energy sources as the main contributors to this distributed energy production landscape. This distributed landscape adds the existence of active grids in both the TSO's and DSO's grid. As elaborated in Chapter 2.1.3 active grids have influence in both the power flow and the short circuit values of the whole grid. These changes to the grid can have devastating effects, such as turning the planned protections of the bus bars inadequate to the new reality [12]. And with the growth of RES, Renewable Energy Sources, this trend will not disappear.

The growth rate of energy production in the European Union might not be as great as the rest of the world. However, non renewable sources of energy are being replaced and it is enough to have a significant impact on both the TSOs and DSOs [2]. This increase in renewable energy production as shown in the Figure 3.1 that shows the different energy landscape from 2000 until the year 2040.

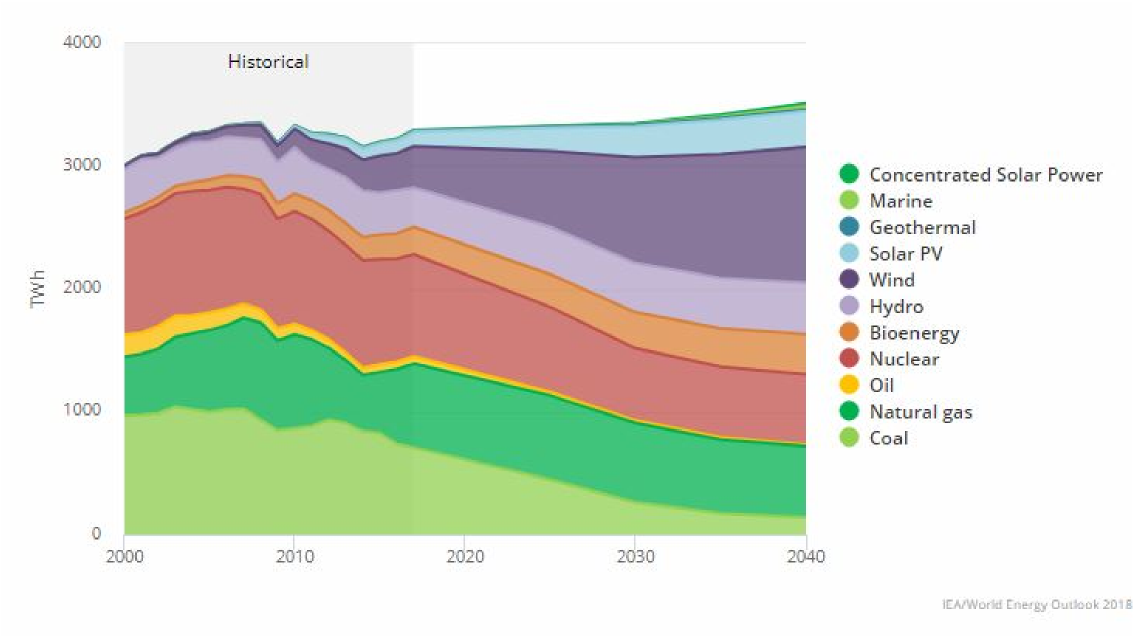


Figure 3.1: European Energy Production Statistics, with discriminated sources [13]

Countries like Denmark, Portugal and Spain are integrating successfully these kinds of energy sources [12]. On the other hand, all this newly generated energy enters the grid at lower voltages, are only considered as an estimated guesses into its operations. This increases the uncertainty of the predictions made of the grid's power flow. While this continues, the conventional generation (Coal, Gas, Nuclear) still is crucial to Europe's power flow [12].

This new types of injection of energy increases the level of uncertainty present in the network, this switches the view of them from deterministic to only a possibility. For that reason the EC, ACER and CEER released statements and protocols to increase the interactions between TSOs and DSOs. The more interconnected these two are the more effective the network planning in all time frame (long/short-term) and emergency system operation are [12]. In summary the upsides of the good communication of TSOs and DSOs are the following:

- Optimization of TSOs/DSOs power systems (short/long term), reflecting on fewer costs and a service with higher quality [12].
- Allows for a better implementation and integration of renewable energies, due to a better observable and controllable aspects of the grids [12].

3.1.2 Observability Area & Responsibility Area

Due to the fact that the new RES (Renewable Energy Sources) are being connected at a lower voltage both the observability and the control ability is a challenge for both system operators.

TSOs, DSOs and SGUs are responsible for their own grid defined as responsibility area. Because all the grids are not isolated the operation of one grid can influence those surrounding it. Therefore to ensure each grid is working as expected, the operators need to have the information of the the other grids that directly influence their responsibility area. This is what it is called observability area. This also means that any grid that has influence in the operators can fall in the Observability area even if it is not directly connected to it [11]. Figure 3.2 shows a practical example of the Responsibility Area.

In the EU the physical border between TSO and DSO varies quite significantly as illustrated in Figure 3.3 This delimitation encumber different ownership of different structures by either the TSO or the DSO. The case represented by the Figure 3.3a both the transformer and the Bus at the lower voltage is owned by the TSO, which is the case used in Germany, Portugal and the UK. The opposite, the DSO owning the transformer and the higher voltage Bus is represented by the Figure 3.3d. The Figure 3.3b shows that the transformer is the responsibility of the TSO but the lower voltage Bus is the DSO responsibility, this case is also used in Germany. And finally Figure 3.3c shows the higher voltage Bus belongs to the TSO and the transformer to the DSO, used in France and Slovenia [12].

Besides the borders between the TSO and DSO varying from country to country, the voltage that these buses operates also are not the same. In Figure 3.4 a graph that represents the voltage levels (in kV) in the border of the TSO and DSO. It is seen a difference in voltage between the countries where the TSO owns the HV bus and the ones that belong to the DSO [14].

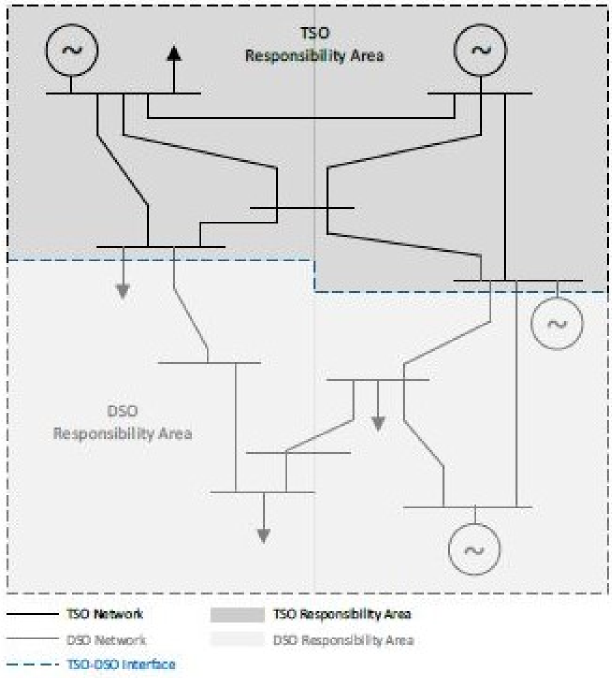
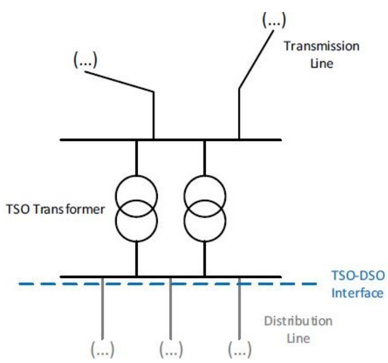
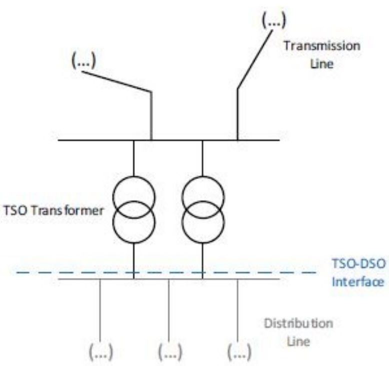


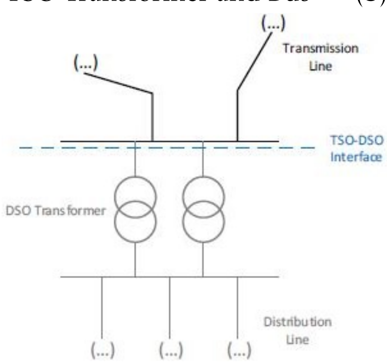
Figure 3.2: Responsibility Area Representative Diagram [12]



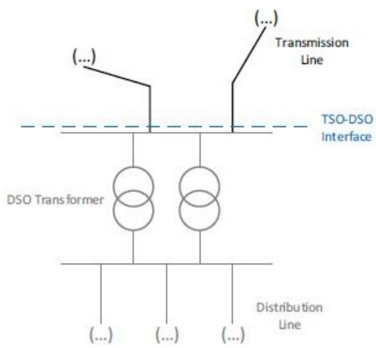
(a) TSO Transformer and Bus



(b) TSO Transformer, DSO Bus



(c) DSO Transformer, TSO Bus



(d) DSO Transformer and Bus

Figure 3.3: Delimitation of the TSO/DSO physical interface [12]

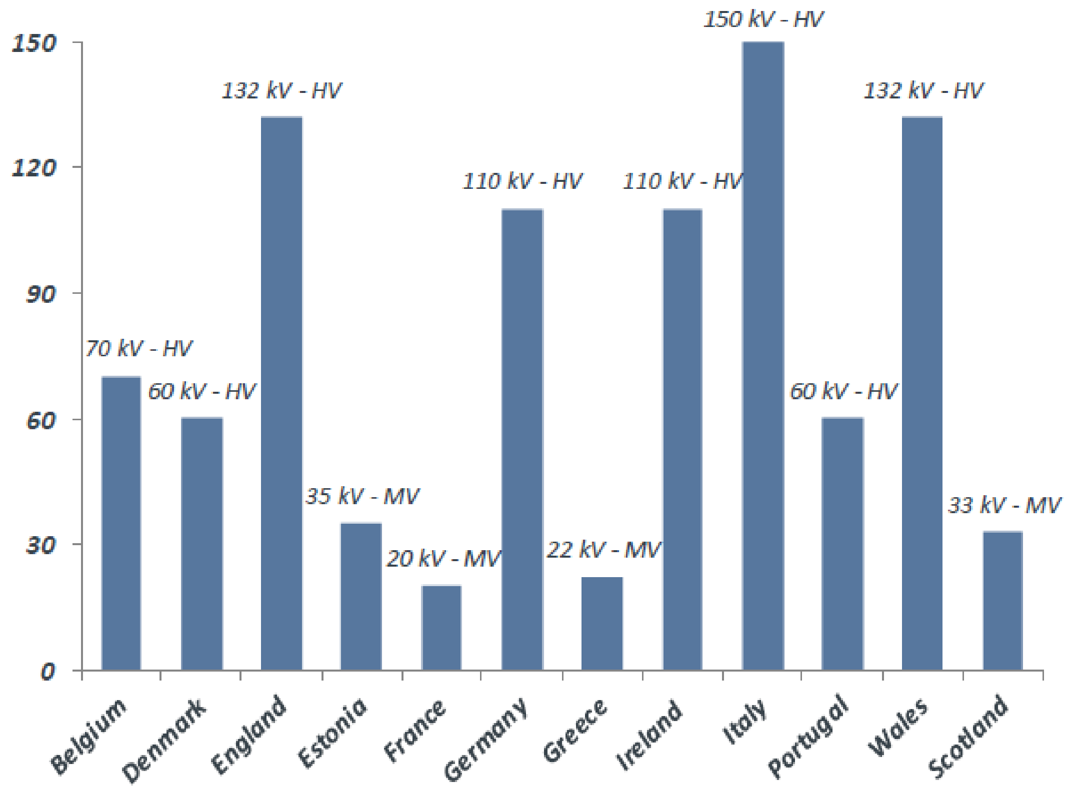


Figure 3.4: European TSO-DSO border nominal voltage values [14]

In terms of observability the more information that the service provider has the better is the forecast. However, due to legal reasons this can prove to be difficult, since the grid's structure is confidential. So defining the amount of observability is not simple, furthermore there is no European regulation that explicitly defines the most adequate observability area definition between TSOs/DSOs/SGUs [12].

3.1.3 Data Exchange Characteristics

The data exchange should fall along the lines that is beneficial to both DSOs and TSOs. In the Figure 3.5 is defined the principles in which data is exchanged [12].

The contents of data are divided into different types such as Real-Time, Scheduled and Structural information. Real-Time refers as the current state of the grids, like topology, power flow and all information in the observability Area. Scheduled information is a prediction of the grid in a point in the future, up to one year away. The scope of the information encompasses the unavailability of a given structure in the grid or the prevision of generation and consumption. Finally the Structural information represents the characteristics of the observable grid (buses, transformers, lines) as well as their dynamic and static models. In Figure 3.6 an example of a observability area is shown, in here the structures shared with the neighbouring system operator are in the common observability area. In this example the area visible to each system operator can be seen [12].

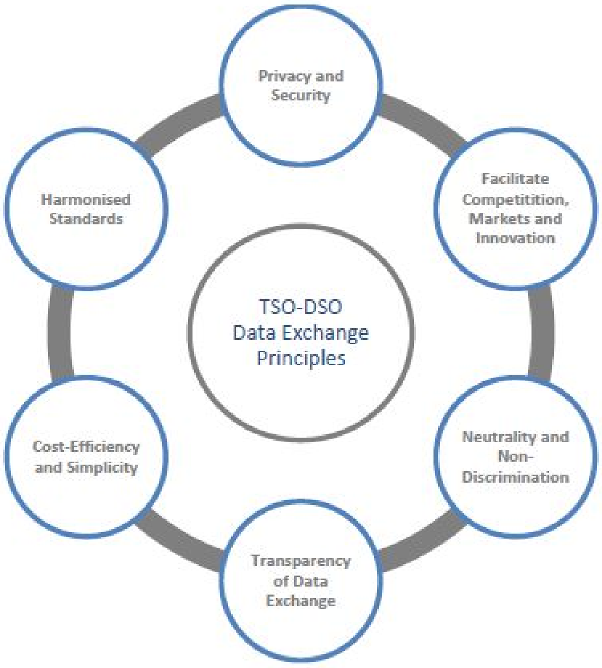


Figure 3.5: TSO/DSO Data Exchange Principles [12]

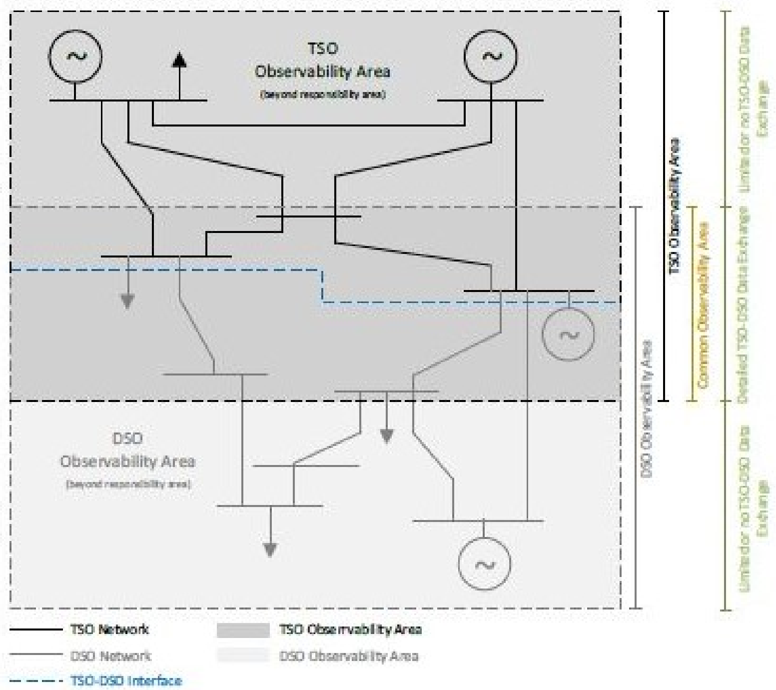


Figure 3.6: Observability Area Representative Diagram [12]

Such as the contents of data, frequency of exchange and the level of data aggregation or granularity [12]. The frequency of data exchange differs depending on the type of information between TSO/DSO. For Real-Time data exchange is required a greater frequency of data exchange, to allow a more updated and reliable readings. On the other hand, Scheduled information does not require that high level of a sampling rate. The granularity of the information exchange between TSO/DSO is dependent of the task to be executed. This characteristic also helps to define the observability area of the service providers [12].

The communication between the service operators must be a secure one, preventing leaks or decoy data to enter the system. It also must keep the integrity of the data, and strive for simplicity in order to fit in other business models. This method should promote competition in the energy industry, in order to promote innovation. Also to improve innovation, there has to be a transparency in the access of data within boundary framework [11]. This transparency will result in more accurate prediction of the grid's power flow and allow for better service provided by the system operators.

3.1.4 Data exchange Regulation Framework

To reach the EU target model for the internal electricity market, three network codes were introduced by the ENTSO-E. These three network codes (Market, System Operation and Grid Connection) are meant to increase integration, efficiency and standardisation of the European electricity market. All to achieve the current EU's energy and climate targets for the period from 2021 to 2030. These targets are, to reduce by 40% the emissions greenhouse gasses from 1990, have 32% of overall renewable energy consumption and a reduce in 32,5% of energy consumption [12]. These objectives are displayed in the Figure 3.7, that has the final targets to reach by 2030 and also the previous 20/20/20 targets to be reached by 2020.

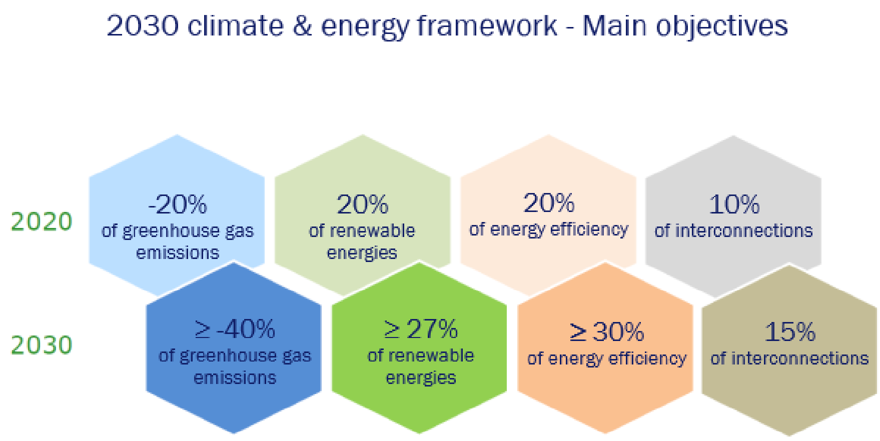


Figure 3.7: EU Key Targets on Energy [12]

In terms of operations there exists specific guidelines, System Operation Guideline (SOG), that establish previsions in what concerns data exchange between TSO-TSO, TSO-SGU and TSO-DSO [12]. Focusing more on TSO-DSO data exchange previsions, the relevant articles will be divided into their respective type of information (real-time, structural, scheduled and regarding all the data exchange).

Starting with the articles that encompass all the data exchange:

- Defines the data exchange. The observability area, limits set by the DSO, as well as other information relevant for the normal operation of the transmission grid [12].

- Article 27 - Data exchange between TSOs, DSOs and Significant Grid User connected to the Distribution Network. This article involves the TSO, DSO and SGUs [12].

- Article 28 - Data exchange between TSOs and Demand Facilities directly connected to the Transmission System. This article involves the TSO and DSO [12].

- Article 29 - Data exchange between TSOs and Demand Facilities connected to the Distribution Network or Aggregators. This article involves the TSO and DSO [12].

- Article 33 - Contingency list. Is composed by the real-time and scheduled information, there can be data aggregation. This article involves TSO, DSO and SGU [12].

- Article 40 - Organization, responsibilities and quality of data exchange. The TSOs are required to develop a methodology of obtaining the data. Process for managing data between the service operators (TSO, DSO, SGU). The data expected is the Time that the information refers to, it's level of data aggregation and frequency. This article involves TSO, DSO and SGU [12].

- Article 53 - Data exchange between TSO and DSO-connected demand facilities or third parties participating in demand response. The SGUs have to provide this information to the TSO, active power maximum and minimum, availability window, forecast, RT active and reactive power, in RT or Scheduled. This article involves TSO, DSO and SGU [12].

- Article 182 - Reserve providing units connected to the DSO grid. There has to be cooperation between TSO and the DSO to maximize the SGU's efficiency. This article involves TSO, DSO and SGU [12].

Real-time data exchange is where the system operators share with one another the current state of the grid within the observable area. This information is refreshed in a span of 15 to 30 seconds [12]. The articles that refer to real-time data exchange will be presented:

- Article 20 - Real-time data exchange between TSOs and DSOs within the TSO's Responsibility Area. This article involves TSO and DSO [12].

- Article 23 - Defines the protocols of preparation, activation and coordination of remedial actions as to be coordinated with DSO. This article involves the TSO and DSO [12].

- Article 26 - Real-time data exchange between TSOs, DSOs and SGUs connected to the Distribution Network. This article involves TSO, DSO and SGU [12].

- Article 44 - Data exchange between TSOs and DSOs within the TSO's control area. It is the exchange of the observability area characteristics information between TSOs and DSOs (topology, technical grid data, generation, consumption). This article involves TSO and DSO [12].

- Article 50 - Data exchange between TSOs, DSOs and distribution-connected power generating modules. SGUs should provide the following information, active and reactive power, current and voltage, except when aggregated data is considered sufficient . This information can be given directly to either the TSO or the DSO. This article involves TSO, DSO and SGU [12].

Structural information refers to all the permanent characteristics and topology of the grids and facilities of the neighbouring system operator. This information regards specifically substations, transformers, reactors and capacitors, under the observability area [12]. The following are the articles regarding the structural data exchange:

- Article 19 - Structural data exchange between TSOs and DSOs within the TSO's Responsibility Area. This article involves TSO and DSO [12].

- Article 21 - Structural data exchange between TSOs, owners of Interconnectors or other lines and Power Generating Modules directly connected to the Transmission System. This article involves the TSOs [12].

- Article 24 - Availability of the TSO's means, tools and facilities. Are the minimum in order to ensure tool compatibility. This article involves TSO, DSO and SGU [12].

- Article 43 - Structural data exchange. It is the exchange of the observability area characteristics information between TSOs and DSOs (substations, lines, transformers, SGUs, reactors and capacitors for each substations). This article involves TSO and DSO [12].

- Article 48 - Data exchange between TSOs, DSOs and distribution-connected power generating modules. This also expands to SGUs connected to DSO they can deliver the information directly to the TSO or through the DSO. SGUs should provide the following information, installed capacity, reserve participation FCR-FRR-RR and voltage data. This article involves TSO, DSO and SGU [12].

Scheduled or forecast information is the expected behaviour of the grids and facilities in a given time frame in the near future with a maximum distance of one year. This forecast includes generation and consumption schedules and also outage plans like unavailability of facilities in the observation area [12]. The articles regarding the scheduled data exchange are the following:

- Article 17 - Scheduled and forecast data exchange between TSOs and DSOs. This article involves TSO and DSO [12].
- Article 22 - Scheduled data exchange between TSOs, owners of Interconnector or other lines and Power Generating Modules directly connected to the Transmission System. This article involves the TSOs [12].
- Article 25 - Scheduled data exchange between TSOs, DSOs and Significant Grid Users connected to the Distribution Network. This article involves TSO, DSO and SGU [12].
- Article 49 - Data exchange between TSOs, DSOs and distribution-connected power generating modules. The SGU should provide the following information, scheduled unavailability or limitation [12]. This article involves TSO, DSO and SGU [12].
- Article 51 - Data exchange between TSOs and DSOs concerning significant power generation modules. TSO defines the sample rate and scope required for the SGUs. TSO and DSO define the SGU that do not require to provide Real-Time information. In those cases there must be an agreement between TSO and DSO to deliver a aggregated data of the SGU [12]. This article involves TSO, DSO and SGU [12].

After defining the characteristics of the data to be exchanged between service operators, now the overall dynamic of the data exchange is described. In Figure 3.8 is detailed the dynamic between TSO and DSO when sharring information, according to TDX-Assist [12]. The first one to act is the TSO that shares the information with the DSO, then the DSO uses the information received to update its own. After this the DSO sends back the information to the TSO which now can also has the complete information at hand [12].

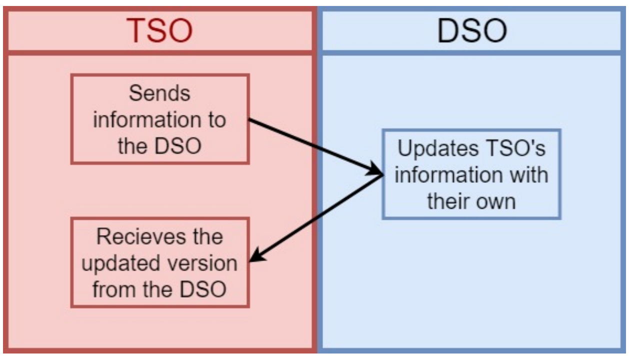


Figure 3.8: TDX-Assist TSO/DSO Framework

3.1.4.1 504 Protocol

This protocol (IEC 62325-504) is part of the IEC framework (International Electrotechnical Commission) for energy market communications, more specifically the use of web-services for data exchange [15]. This framework is supposed to function in near real-time, in a secure way and it can be applied as the solution to other types of integration problems outside its scope [15]. The 504 has some base requirements that need to be met in order for its effective implementation [15]. As most of them encompass the minimal technologies and methods to be used for the base 504 to work [15]. These are referent to the type of file of the exchanged information, this being the XML (Extensible Markup Language) which is a language designed to store and transport data. Besides the file type there must other information that characterises the exchanged information itself. As for the normative web-service required the protocol states that a SOAP based web-service is ideal for the implementation. SOAP stands for Simple Object Access Protocol, it is a message based web-service that can exchange XML files. Nonetheless, a web-service to fall under the requisites of the 504 protocol needs to be able to exchange XML files freely and be able to replicate the operations that are the foundation of the protocol. For that the RESTful architecture is a valid substitute for the protocol's web-service used instead of SOAP [9]. As for the main operations of the with protocol they are three with distinct functionalities, these operations are:

- List messages - Operation that queries the database for a entry, that the user defines, and displays the information on screen[15]. This can be achieved trough a GET request.

- Get messages - Finds a certain file entry in the database and imports the XML file to the client's computer[15]. This can be achieved trough a GET request.

- Put messages - Saves to the database a specific XML file provided by the client along side the information specific to the file [15]. This can be achieved trough a POST request.

As seen in this protocol the only type of requests needed are the GET and POST, simplifying the server functionality. All of them have certain dynamics in terms of information that enters and leaves for each request [15]. In the table 3.1 there is all the information exchanged in each action. Each one of the operations has a request and a response, where the request is what is made by the user and the response is what the server returns to the user [15]. In each one of these interactions, albeit request or response, there are some information that needs to be sent or received when they are executed. There are three types of information in each operation, Mandatory (M) information that needs to be sent, Choice (C) the user must put at least one of these and Optional (O) where it is not required to convey this information [15].

Table 3.1: 504 Service Requirements[15]

	List		Get		Put	
	Req	Res	Req	Res	Req	Res
Message ID	O	M	C	O	O	-
Message Version	O	M	C	O	O	-
Internal Code	C	M	C	O	O	-
Status	C	O	O	O	O	-
App Time frame	C	M	O	O	O	-
Server Time	C	M	O	O	O	-
Message Type	O	M	O	O	O	-
Data Owner	O	M	O	O	O	-
File	-	-	-	M	M	-

*M-Mandatory, O-Optional, C-Choose

3.2 Portuguese Case

The transmission grid is operated by Rede Electrica Nacionais, S.A (REN, Portuguese TSO) , it functions at EHV (400, 220 and 150kV) covering the whole continental part of the country. In 2017, the grid had 8 907 km of line in all it's extent and managed to transport 49,6 TWh of electrical energy [16]. On the other hand, the distribution grid is controlled predominately by EDP Distribuição (Portuguese DSO), functioning in HV (60kV), MV (30, 15 and 10kV) and LV(400V). In 2014, the grid had 224 000 km of line in all it's extent and managed to transport 44 TWh of electrical energy to 5.5 million costumers [2]. In the Figure 3.9 below shows the the transmission grid across the country.

The energy consumption and production in Portugal is highly dependant on renewable energies, in 2017 the represented 40% of all the energy produced in the country. As shown in the Figure 3.10, the renewable energy has been in a constant rise except in the year 2017. In that year it can be seen that of all the renewable only the hydraulic reduced it's production [16]. This results are fruit of the combined effects of the drought felt in the country that year and the enormous amount of wild fires. And due to that deficiency of energy production led to a compensation with the non-renewable energy sources. Without that case, we can see that the amount of renewable energy production is increasing in Portugal. The predicted scenario for the years 2020 and 2030 are shown in Figure 3.11. In similarity with other developed countries the growth rate in this prediction matches the normal growth rate of these kinds of countries. The growth rate used is of 0,4% making the production of energy in 2030 of 54200 GWh and the use of fossil fuels is reduced to less than 10000 GWh in 2030. However this is a hopeful scenario where the thermal energy is reduced to close to 20% of the total energy production [2].

Figure 3.9: Transmission grid of Portugal [16]

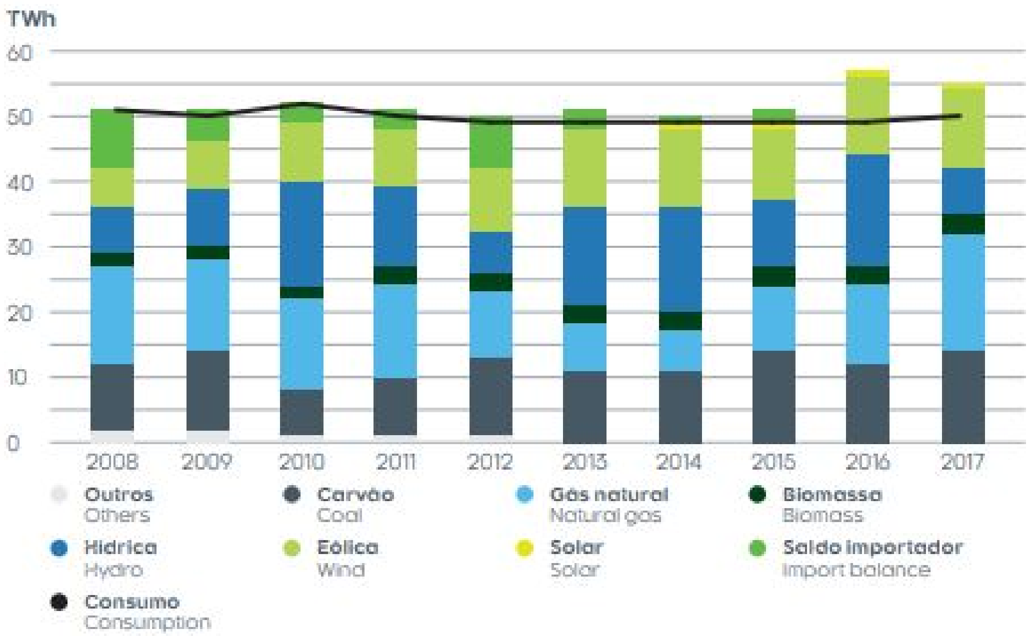


Figure 3.10: Consuption vs Production from 2008 to 2017 [16]

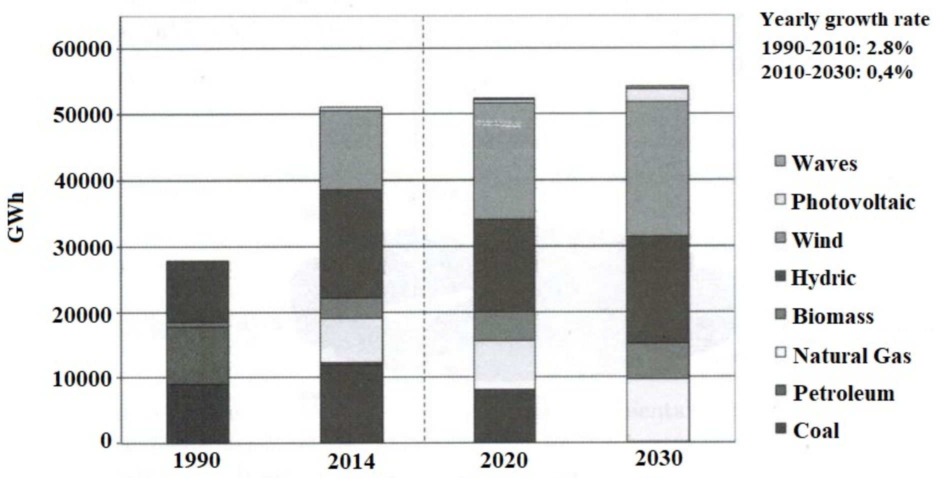


Figure 3.11: Portuguese Power Prediction [2]

3.2.1 BUC - Business Use Cases

As mentioned previously there are types and methods of data exchange. The Real-Time Data exchange between EDPD and REN is based on a ICCP. The TSO and DSO possess a SCADA system connected to the ICCP,that enables the Real-Time information monitoring and analysis of the data to be exchanged. The ICCP also facilitates SGUs to connect to the system and sharing their readings allowing better predictions[12]. The Figure 3.12 illustrates the data exchange flow and the ICCP interface between REN and EDPD.

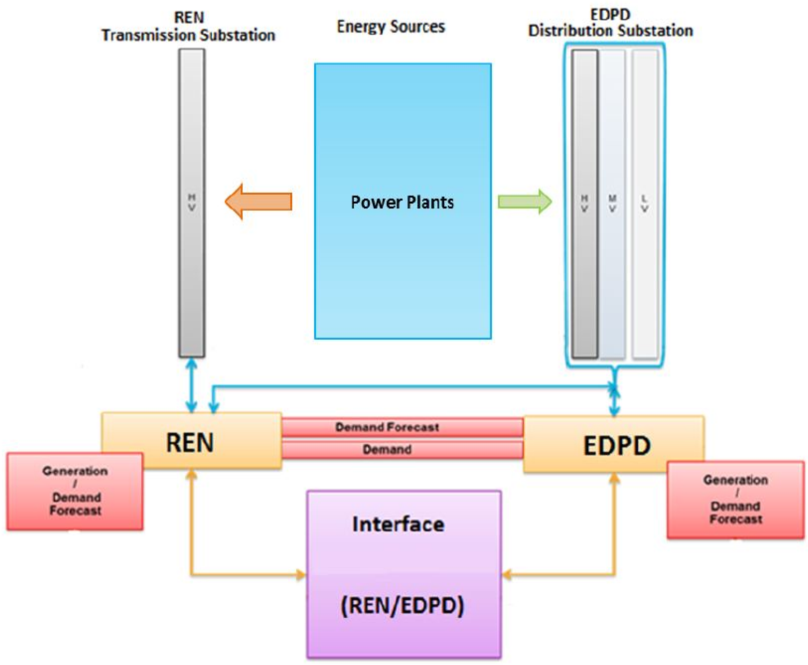


Figure 3.12: REN and EDP interface and information flow between grid users [12]

The data exchange that is shared between REN and EDPD amounts to the Real-Time values referring to more than 40 transmission substations and more than 20 distribution substations, Active power, Reactive power, Bus Voltage and Current, etc [12]. The frequency that this information is shared is within every 10 seconds, this exchange gives a good tool that allows the observation of the network conditions[12].

This system is an effective tool that ensures a reliable and accurate exchange of real time data between the Portuguese TSO and DSO. This has allowed an improved relationship between the TSO-DSO, resulting in a more expansive observability area, in particular in wind powered generation. This is vital in the implementation of a more distributed energy generation like the RES, such as wind and solar.

Although the ICCP being a good tool in the Portuguese energy market there are some other aspects that both the DSO and TSO want to exchange between them [12]. The aspects that the system operators want to improve upon are as follows:

- Outage scheduling and restructuring of protocols of both operators.
- Operational protocols, information and schemes in incident conditions.
- Structure scheme of both grids as well as the constant update of this information.
- Expediting the safe and correct implementation of Renewable Energy Solutions.

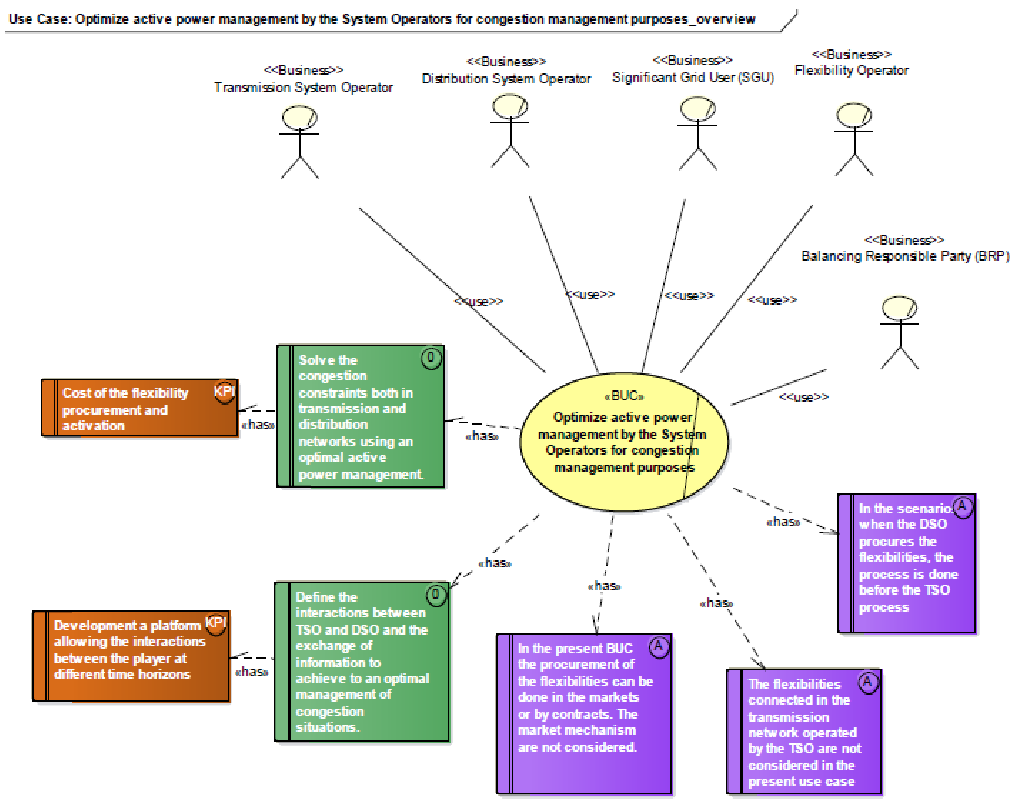


Figure 3.13: Example of a BUC and SUCs [12]

To tackle these aspects the Portuguese TSO and DSO have defined three BUC (Business Use Case). The BUC describes the actions of an actor or actors and a system that defines the overarching need that needs to be met. A BUC has several SUCs (System Use Cases) that can be defined, these SUCs define different tasks inside each BUC. These SUCs are akin to tasks that are required to be completed in order to the whole BUC to be finished. Only when all the SUCs of a BUC are successfully implemented, that is when the BUC is complete. This is exemplified in Figure 3.13 where the different actors (TSO, DSO, SGU) interact with the BUC, and the BUC divides itself into several SUCs.

The three BUC to be implemented have different goals in mind, one of them focuses more on Operational Planning (Scheduled information), while the other two in System Operation (Real-Time information). The Figure 3.14 represent the different BUC that are being developed in Portugal, Slovenia and France, the Portuguese ones are represented in green. Also bellow the three BUCs are defined as well as their corresponding SUCs [17].

Synergy between TSO and DSO of Operational Planning activities.

- This use case aims to improve the programming of the networks operation of the TSO and DSO. In order to do it the system operators defined the information exchange between them. The DSO, every 15 minutes, makes a forecast of the distributed generation connected to it and categorises its contribution by energy origin type (wind, solar, hydro, etc.) and point of origin. After each forecast the DSO shares the data with the TSO resulting in a more secure and efficient operation of the power system[17].
- For the system operators there exists neighbouring grids that affect the operation of the its grid, such as network loops (for example a line belonging to the DSO that connect to two TSO bus bars). Due to that a observability area needs to be defined, this definition will benefit both system operators allowing more accurate power flow calculations[17].
- The exchange of data for this BUC requires to done 72 hours ahead of the predicted time, providing a new version every 24 hours[17].

Supervision and control of Real-Time system coordination.

- Aims to improve the already supervised observable area by increasing the level of detail of data exchange. Also expanding this level of detail to the non-observable elements of the neighbour's grid[17].
- Enable of enhanced supervision and control over the TSO's and DSO's grids. To reach this goal the exchange of real-time information between TSO and DSO needs to be guaranteed[17].
- The observability area needs to be updated periodically or whenever it is necessary. This must ensure a fortuitous observability of the neighbours grid, in spite of any change made to the grid[17].

Fault location at high voltage line bay.

- Focus on faults occurring in the border of TSO and DSO, improving their location and effect. For that goal having the information from the TSO side available to the DSO will give it the ability to improve the location accuracy of faults in their lines. This needs information to flow in real time in order to accurately detect the location. The goal of this BUC is to improve the DSO's performance optimising the service[17].

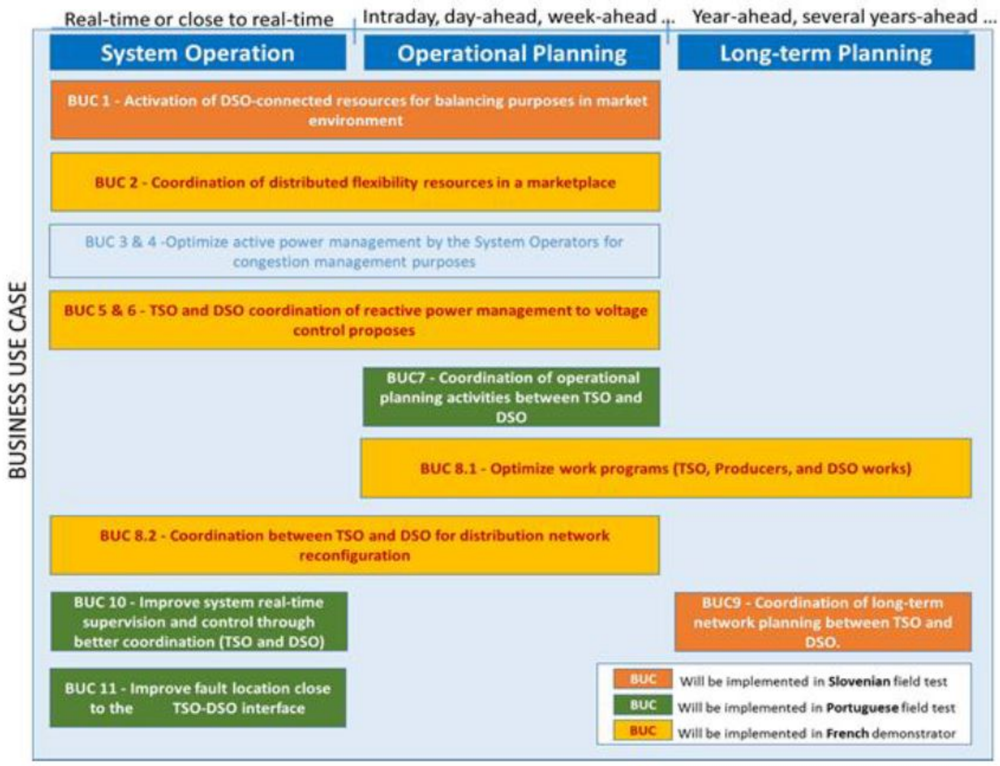


Figure 3.14: European BUC to be implemented [17]

3.2.2 Planed Short Circuit Current Range

There are plans for the grid development in the Portuguese Electrical grid infrastructures plan (PDIRT,*Plano de Desenvolvimento e Investimento da Rede Nacional de Transporte*) [4]. These plans, are bound by legislative obligations and rules of the electrical sector. In them there is a particular section addressing to the the expected short circuit currents. In this section there is limitations specified for the buses in the Portuguese TSO grid. These limits define the maximum and the minimum short circuit current range allowed in the bus, as depicted in Figure 3.15 [4].

There are three values of current in this Figure, $I_{cc}min$, $I_{cc}max$ and $I_{cc}limit$. These are reached through running simulations of the system operator's grid and finding the lowest value($I_{cc}min$), the highest ($I_{cc}max$) and the one that the buses protections allow

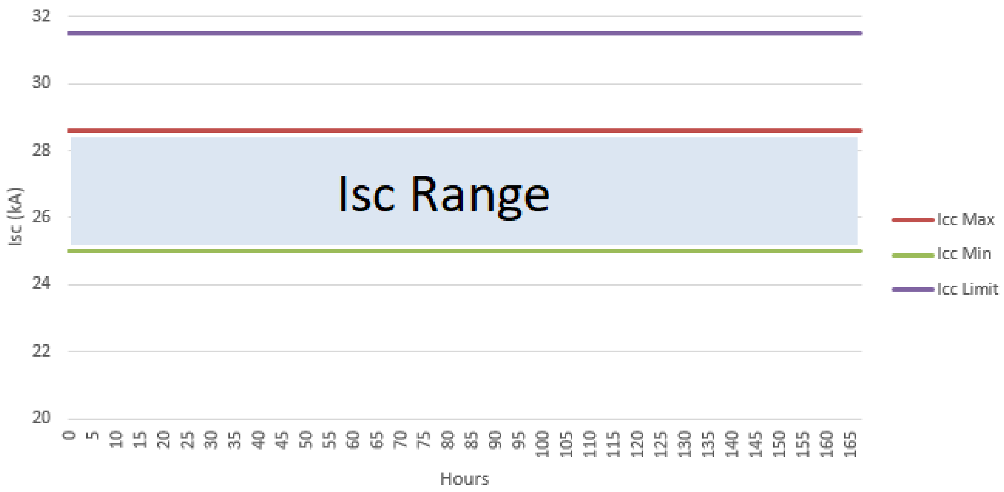


Figure 3.15: PDIRT Isc values and range for the grid

($I_{cc}limit$). The allowed range of short circuit current is between the $I_{cc}min$ and $I_{cc}max$. The value for $I_{cc}max$ is different for REN and EDPD being 25kA for REN and 31kA for EDPD, this is applied for all the buses in their grids. With PDIRT the values stop from being the same for every bus in REN's grid to have specific values for each and every one of them. Nonetheless, it is a rigid limit and does not factor in the present value of short circuit current of the grid. For Figure 3.16 itis based on the information from Figure 3.15 and adds a profile of the short circuit current in the Transmission grid bus during a week. With this it can be seen that the current is inside the designated range, however, it has time frames where the distance between the actual values and the limitations is big.

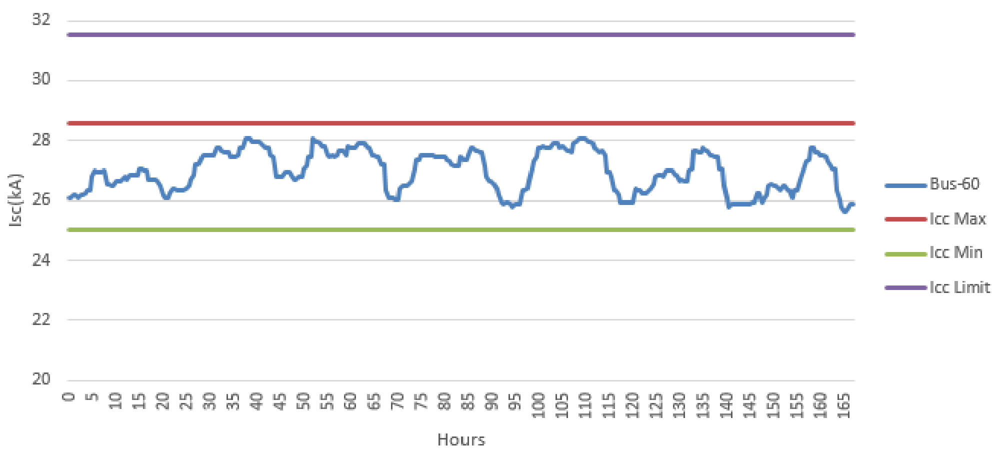


Figure 3.16: Hourly dynamic compared with the PDIRT short circuit currents

Chapter 4

Proposed Solution Implementation

4.1 Overall structure

The implementation of the web-services has different levels of information processing and interaction. These can be divided in two separate instances the Client architecture and the RESTful Server. The client architecture resides in each client's machine and having an output that can be read and exchanged. The RESTful Server functions as a third party database that collects all the data sent by all the clients and the RESTful web-service functions. The figure 4.1 is a representation of the developed implementation. The client is responsible for the generation of the XML file with the established structure and correctly interact with the Server, as shown in Figure 4.1. The TSO, DSO and SGU machines connected to the network can preform the three basic 504 protocol operations (Put, Get and List). As for the Server its function also depends on the interaction from the Client. As stated in the characteristics of the RESTful web-service it is to a cloud service, however the RESTful web-service alone is not enough requiring a database. This database's function is just to store the information provided by the web-service. The interaction between the database and the web-service depends also on the Client's input. If there is some inquiry made by the Client, the web-service has to search the physical database to find the intended data. The developed work has also divided into the two sections of the service, Client and Server, both with different dynamics and requirements.

The Client's PSS/E Python API intends to retrieve short circuit values from a grid and transfer that information into a structured file. This aims to get the accurate value of all the short circuit Currents and Power in every Bus of the inputted grid. The ideal output is a XML file with all the relevant information retrieved from the Python API. The XML must have a identification structure, the values of three phase currents and the contribution from HV and EHV. There is also the possibility of gathering all the

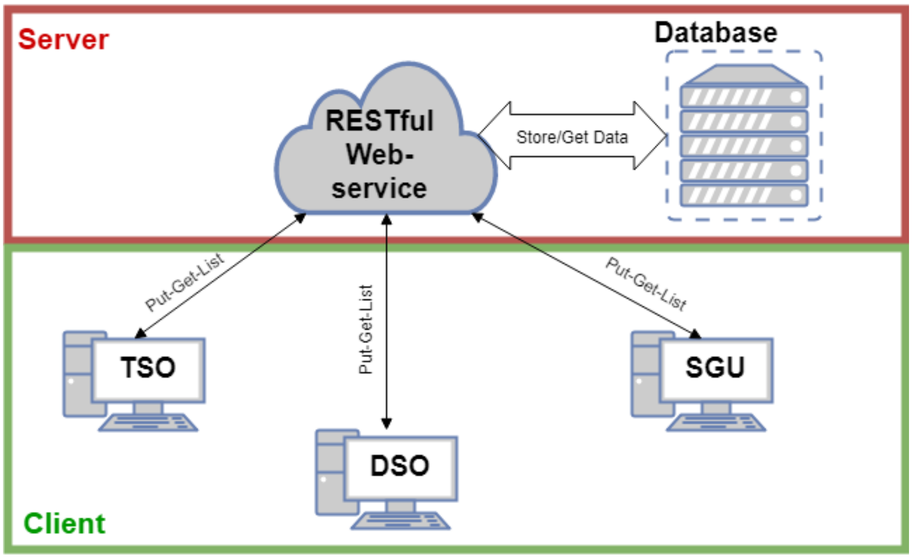


Figure 4.1: Service Structure.

information in an Excel file. With this there should be some control options for the user in terms of the amount of buses the information is displayed. This user interaction on the bus subsystem choice is done in several ways. Either the user picks all the bus bars, the ones that belong to the TSO or DSO, the buses in the border of the system operators or pick each bus singularly. Also there a functionality that allows the user to define the intended subsystem from an Excel file with a given structure.

As for the RESTful Server the goal is to create a stable interaction with the client, as well as the safe and correct relationship between the web-service and the database. For the correct implementation it is required to create a server service that can successfully enable contact between the service operators, and that the information that travels is the correct one. Implementing all the used requests (According the 504 protocol) and allow multiple different information instances (different days) to be allocated by this server. Because it deals with sensible information confidentiality and information security are a key priority for this kind of data exchanges.

4.1.1 TDX-Assist

With the structure and the delimitation of the Server and Client well defined, the dynamic between both of them and the different Clients also can be established. This dynamic has as a basis the TDX-Assist with the message and communication structure of the 504 protocol. The Figure 4.2 has a clear representation of the overall process structure. The first interaction with the server comes from the TSO when the obtain the forecast data, of a given date, in an XML file and send to the server using the PUT Message operation. After that the DSO searches in the server if there is any information regarding the date in question from the TSO, by using the LIST Message. In the eventuality that the information that the DSO searched exists, it retrieves said file from the server using GET

Message. This downloads the TSO grid information to the DSO machine. After this the information present in the XML file is imported to the DSO's grid files referent to the date in questions and with this updating its information with the TSO contributions. After these grid files update a new XML with all the contributions is created and the process described is repeated in reverse from DSO to TSO. In the case of a SGU authorised to share its information with the server this dynamic is still applied. This changes in the Figure 4.2 the TSO with SGU and the DSO with the grid operator in which the SGU is connected to.

Figure 4.2: TDX-Assist TSO/DSO [12]

This is the over all structure that this dissertation implementation is based on being a guiding requirement for both the Client's and the Server's API. With this now the goal of the project is clear and the implementation can be presented.

4.2 Client Architecture

4.2.1 API Functionalities

The developed API receives a grid from the user and returns a file containing relevant information from the grid. For that to be accomplished there are several parts in the Python script that need to be taken into consideration. The API has three defining components that have their specific task. These three are, the GUI & Input, Main Body and Output. In terms of order in which they are run in the script follow the order in the Figure 4.3.

Figure 4.3: Client API structure

4.2.1.1 GUI & Input

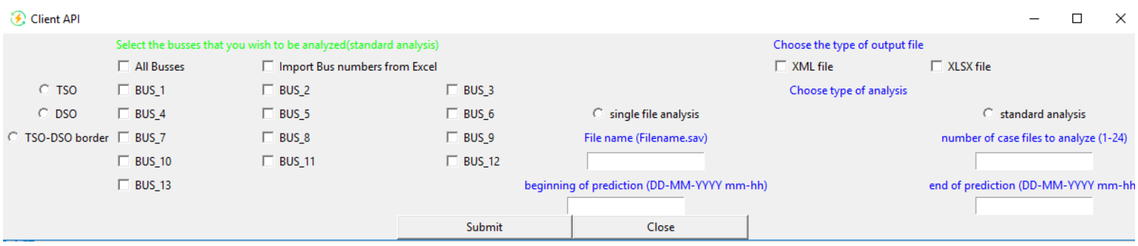


Figure 4.4: Client API GUI Window

As the script is run the user is prompted with the window displayed bellow in Figure 4.4. In this window the user can choose the grids that will be taken in consideration, the buses that will be analysed and the preferred output. For each of this choices the script is prepared to effectively operate with all possibilities. The choices that refer to the type of analysis and it's extent are variables used in the Main Body of the script. While the ones referring to the output type and the date of analysis are directed to the Output portion of the script.

Starting from left to right of the window, there is the bus subsystem analysis portion, this defines the buses that the user wishes to be analysed, as shown in the Figure 4.5 . This left section information is to be used on the Main Body of the API. There are the individual bus button that represent each bus in the grid. There are the TSO, DSO and TSO/DSO Border buttons, these create a subsystem based on the System Operators

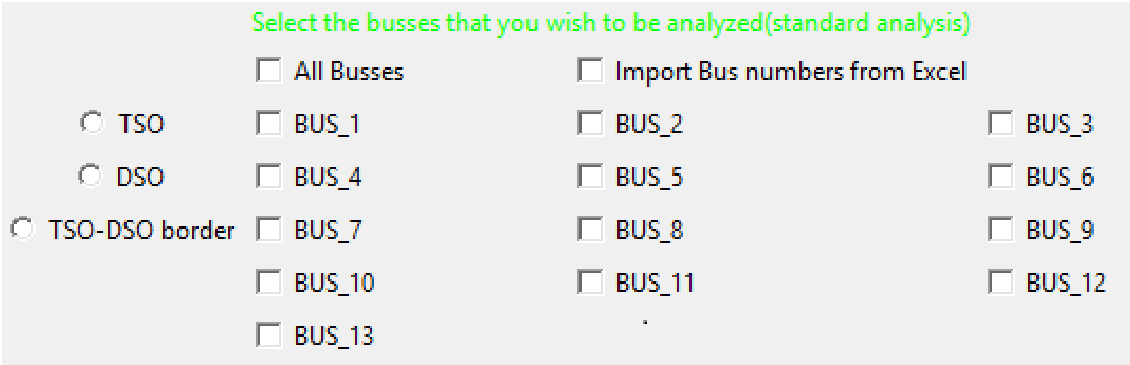


Figure 4.5: Bus Subsystem Section

structures. TSO for buses with base voltage of 70kV or above, DSO for buses below the 70kV base voltage and the border for buses that are neighbouring the other System Operator. The All Buses button enables the analysis of, as the name implies all buses. This choice has priority over all the others since it being active means it has the consent of the user. To finalise the left section of the window, there is the import form an excel file button. This reads from a defined .XLSX file which buses will be subjected to the analysis. This file must be previously prepared by the user in order to have the appropriate format. The format of this excel file is as follows:

- First column: Number of the bus
- Second column: Name of the bus
- Third column: Planed Isc limit value of the bus

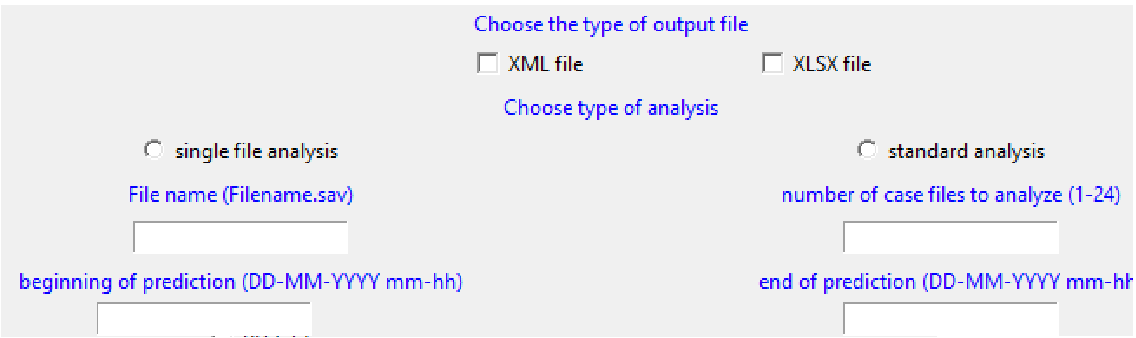


Figure 4.6: Analysis and Output definition Section

Now for the right side of the window displayed in Figure 4.6. Starting from top to bottom. first there is the output options these define the type of files created at the end of the script. The two types of outputs are XML and Excel files, the script can create both of them at the same time or just one of them. Next there is an option of making a standard analysis or giving the user the option of selecting a different file. The user is only allowed to make one of these at a time. Below each option button there is a text box that the user must fill. For the single case file option it is the .sav file name, the file must be in a directory defined in the script. On the standard analysis option the user must fill the text box with a number between 1 and the maximum amount of cases. If the user fills with an invalid answer or a number that exceeds maximum number of cases the API defaults to the maximum number of cases. After that there is another two input text boxes where the user must specify the date and hour of the beginning and the end of the prediction cases. After all the mandatory fields are filled the user can press the submit button in order for the script to advance for the next phase.

4.2.1.2 Main Body

For the Main Body of the Client's API, it was divided into three parts. These are represented in the Figure 4.7 as well as it's interactions with the other components of the API. The API is protected against user misuse, however some other errors might also occur. For that some protective measures were implemented, but only the more obvious and apparent ones.

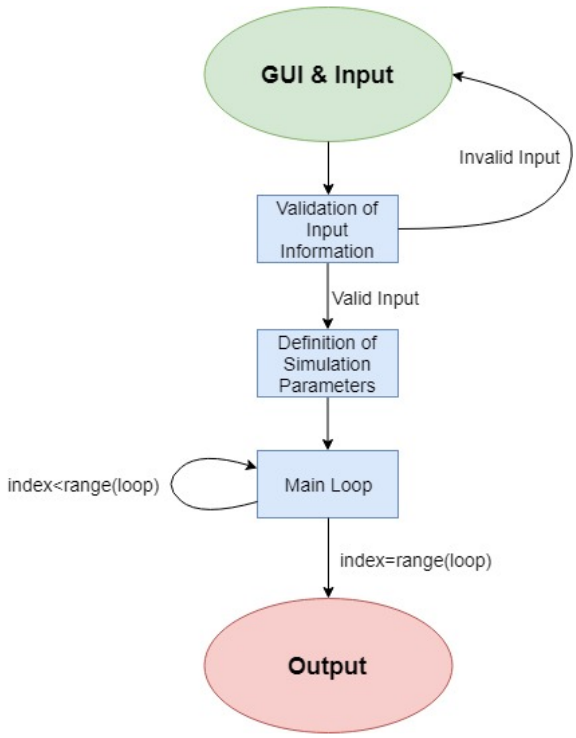


Figure 4.7: Client API Main Body summarised structure

- **Validation of Input**

  In this first step before progressing further the API checks if the user submitted correctly all the required fields. These include the definition of the bus subsystem, type of analysis, the prediction time frame, and if chosen the single file analysis the correct case file name. This are put into place in order to protect the script from unexpected and misuse by the user's side.

- **Definition of Simulation Parameters** This second step retrieves the actual information the user inserted and converts it into the entry parameters of the simulation. In this information treatment include the definition of an array with the intended bus numbers, the amount of cycles the main loop will do and the name of the .sav case file to be analysed. The array of bus numbers is required for the definition of the Bus Subsystem, if the user chooses all buses this step can be skipped. This is achieved by the definition of a variable that forces the Psspy functions to run all buses. The amount of cycles is defined depending on the option of analysis type the user has chosen. For the single file it runs the main loop one time, but if it's the standard analysis this is defined by the user. As for the grid's case file name it only changes when the user defines the analysis type as single file analysis. As for the standard analysis it will use the default location and name defined in the script.

- **Main Loop**

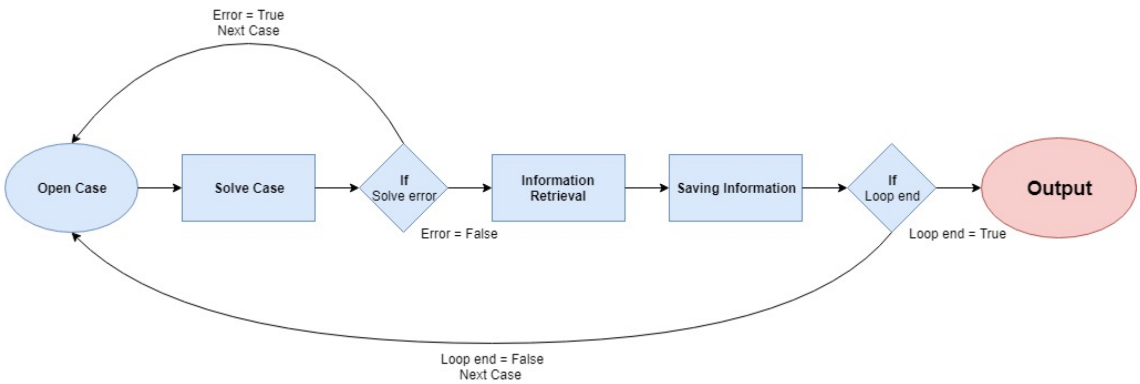


Figure 4.8: Main Loop structure

This step of the Main Body is where the actual analysis takes place. As shown in Figure 4.8 the detailed functioning of this step.

First, the selected case file is opened and solved. The case is opened with a specific Psspy function. The opened case depends on the option chosen by the user, either the standard file or the one inserted manually. As for the solve case, it is also used a Psspy function, this function returns the error values that might occur. As a protection to the rest of the loop if the function results in error, that case file is skipped and opens the next one. The errors that the solve function detects if the simulation has unsuccessful or if the a tolerance requirements are reached.

Second, with the case file solved there can be querying concerning the information of the analysed grid. After solving the case file the API follows the following steps.

Creates a bus subsystem, this only occurs when the user doesn't choose the all bus option. This is used to define the scope of the information used on the next Psspy functions.

Gets the values of the three phase fault and the current contributions from the neighbouring buses. All these values are retrieved in p.u. therefore they must be submitted to a change of base in order to be converted into Ampere. Also the contributions must be organised into two categories, High voltage and Extreme High voltage, depending on the neighbour's Base voltage.

Finally, The information of the fault currents is stored in a matrix of values to be used later in the output component. This matrix size depends on the amount of buses in the grid and the amount of cases that are analysed. For each case three rows of the matrix are created and each column represent an analysed bus.

4.2.1.3 API Output

The output of the API, as previously mentioned is controlled by the user, as shown in Figure 4.9. To the user is given two choices of types of outputs, these include a XML file and an Excel file. Both serve different functions and are structured to fit their purpose. The XML is more for the sharing of information between Client and Server, as for the Excel more for internal information visualisation and control.

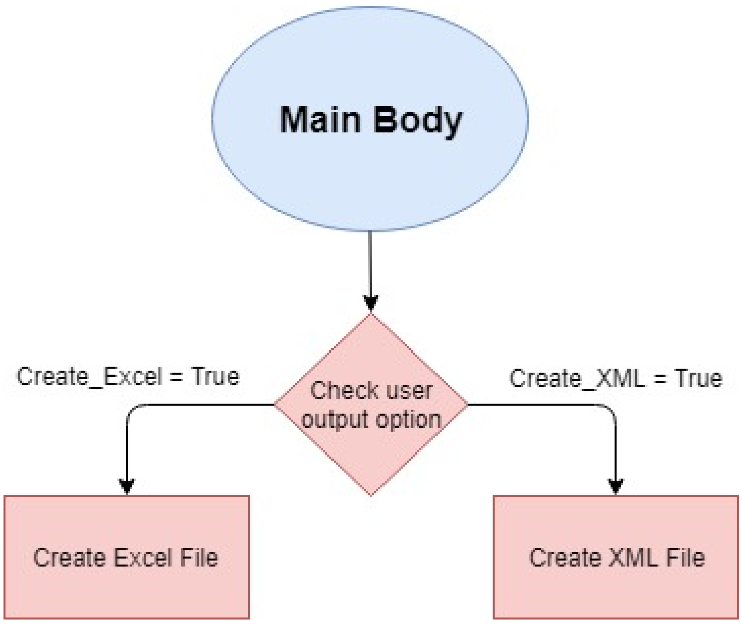


Figure 4.9: API Output structure

- **XML File**

  The XML will serve as the message basis, and travel between the Server and the Client. This is the main output for the communication in the project. Therefore it requires a message structure that includes all the relevant information. This is in order to minimise the size of the actual file, making it easier to transmit to the Server. The Python library used for the XML interactions has the built in XML library. The file is divided into two parts, the header and its contents.

The header is given the name of Global Data, and as the name implies this refers to the global identification of the file. The information in the header is the date and time of creation of the file, the prediction time frame given by the user and the date of the last update. However, since every time the API is run the script generates a new XML the date of creation always coincides with the last update date.

After the header comes the usable information in the file, named ID-ISCP in the XML. In this part of the XML the information of the three phase fault and the contributions is arranged by analysed bus and then save case.

The output should look like Listing 4.1. The version of XML used is the 1.0 and it has an encoding of UTF-8. In this example there was only one bus analysed(bus 5) and only one save case was run.

```
<?xml version='1.0' encoding='UTF-8'?>
<Prediction>
  <Global_Data>
    <Last_Update>07/08/2019</Last_Update>
    <Valid_From>14/08/2019 00:00</Valid_From>
    <Valid_To>14/08/2019 00:00</Valid_To>
    <Date-Time-Created>07/08/2019 13:23</Date-Time-Created>
  </Global_Data>
  <ID-ISCP>
    <Hour-0>
      <BUS_5         >
        <ISC-Total>2684</ISC-Total>
        <ISC-A>1811</ISC-A>
        <ISC-B>903</ISC-B>
      </BUS_5         >
    </Hour-0>
  </ID-ISCP>
</Prediction>
```

Listing 4.1: XML file content

In Figure 4.10 is represented a simplified version of the origins of the short circuit values in the XML file. This refers to the three values in the body of the file, ISC-A, ISC-B and ISC-Total of this Bus 5. Assuming that this Bus 5 belongs to the TSO, this being an EHV bus, the values represent different values of short circuit current. ISC-Total represents the total short circuit current felt in bus 5 when a fault is located in that same bus. As for ISC-A and ISC-B these are contributions from the rest of the grid and from neighbouring grids. ISC-A refers to the EHV contribution to the ISC-Total and the ISC-B to the HV contribution. In some cases the ISC-Total is the same as either ISC-A or ISC-B, this only happens when the bus is not located in a frontier position of different voltage levels (EHV, HV).

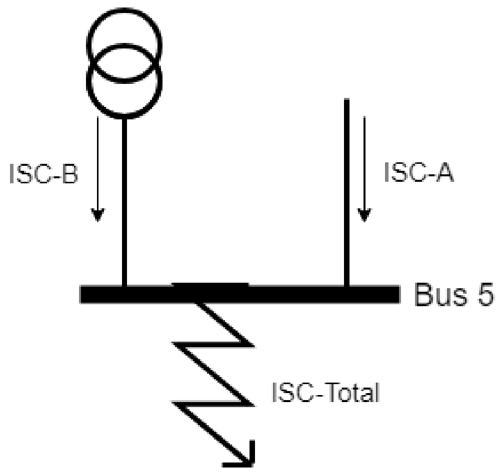


Figure 4.10: Bus short circuit current Diagram

- **Excel File**

  The Excel file's function is more of a control of the system made by the Client. This file is not to be exchanged with the server, but for the user visualise if any of the ISC limits have been broken. In Python the library used for the interactions with the excel files is Openpyxl. This is a very versatile tool and possesses great customisation options.

  As with the XML the Excel file also has a header to define the identity of the file. It has all the attributes of the XML. The name of the file depends on the date, time of creation and creator of the file, appearing with a structure as follows:

  "TSO name_DSO name_ICC_date of creation_time of creation".

  The TSO and DSO names can interchange depending on the creator. The content of the file akin to the XML also portrays the information retrieved from the previous module, three phase fault and contributions. Although, to this information is added the short circuit apparent power in MVA since this file format is easier to visualise. Using this ease of customisation there were some developed functions to analyse beforehand the results. To the three phase fault of each bus were applied three filters. These filters depending are outputted as a change of fill colour on the referred cell. The first filter is inputted by the user, if the bus subsystem is defined by the excel file mentioned in the GUI & Input function. This value is dependent by the planned short circuit current limit defined in PDIRT. If the three phase fault exceeds this value the cell is painted yellow. The second filter is the DSO short circuit current current limit, this value is 25KA. If the user bus subsystem is not defined by the Excel this becomes the value for the first filter. If this filter is passed the cell is painted orange. For the third and final filter it is the TSO short circuit current limit, with a value of 31KA. By passing this final filter the cell becomes red. To ease the detection of filter breaches, whenever one is crossed the identification of the referred case is also highlighted with the worse registered breach.

4.3 RESTful Server

The Server level must obey the prerequisites imposed by the European norms, legislation and protocols enunciated in Chapter 3 Section 3.1. This is required to have, in order for the information exchange between System Operators be secure. The RESTful Server was developed to count with the possibility of being scaled up in terms of information traffic and hardware. For the detailed composition of the Server itself there are aspects that were implemented in this project considering the Server growth. The chosen Language for the server was Python since there is several programming libraries developed for it.

4.3.1 Web-service Characteristics

This Server was developed basing on the RESTful Architecture explained in Chapter 2 Section 2.2. In order for the server work with the European directives and requisites there were some specific characteristics that were implemented. This solution encompasses the security and data acquisition and storage of the directives. The following are the core characteristics that allow this to be a viable Server.

4.3.1.1 SQL Database

This is one of the vital aspects of the Web-service since it saves in the database the information that the user inputted. This database saves two tables, one for users and other for files registered. The SQL, or Structured Query Language, is a used language to interact with databases. For that the Web-service requires a interaction with this language, in order to create, search, delete and update members of each table. This interaction between Web-service and SQL is made through Python since the basis of the server is that language. The library used is the SQLalchemy, this enables full access to the database, therefore perfect for the requirements. The two tables created have specific functions that make their existence essential for the Server to work.

- User Table - This table saves the accounts that are actually able to access the Server. The information in this table is to identify the user. The contents of the User table are id, username, email and password:

  Id - Is an internal identification of the account this is not asked to the user, it serves as a primary key. By being a primary key means that this is the primary way for the SQL to identify a given row of the User table.

Username - Is another way to differentiate the different rows of the tables since each user name must be unique. However, the difference between this and the id is that one is for the Server to identify and the other is easier for the Client to identify.

Email - Is for a future functionality in order to complete the authentication process. It is still part of the User table to serve as another unique identifying element.

Password - Serves the same purpose as in other web-sites with accounts. This is a way to protect the Server from being accessed by non authorised users.

- File Table - This table stores all the relevant information from a file saved in the database. This table must contain all the information necessary for the exchanges between Server and Client. The information stored in this table is as follows.

  Id - Functions the same way as the Id from the User table.

  Version - Refers to the integer value that defines the most updated value. If the value of the version is higher, means that the file in question has more update and recent information.

  Status - An integer value that varies between 0 and 1 that signifies that the file entry is the latest version. 1 signifying it is active and 0 as inactive.

  Data owner - A string value that has the name of the Service Operator that file is associated to.

  Type message - Characterises the type of the information that the file possesses, either scheduled, real time or structural.

  Date reference - The date or dates that the file prediction contents refer to.

  Date Span - Refers to the hours and minutes of beginning and end of the file contents.

  Server time stamp - Time that the server received the file, this is done by the server's internal clock.

  Data - Actual file content.

4.3.1.2 Log-in & Security

In order to limit the access to the main operations of the Web-service there were some structural functions implemented. Most of the security is made by the network and firewalls that the Server is connected to. This already limits the access of outsiders to the Server, still there are some additional measures that can be implemented.

The one applied is a log in and account verification. In the Server the access to the main page of Web-service there is a block in the form of a Log in form. This makes sure that only users that are stored in the database can gain entry. For that a log in and sign up page was created, this asks the essential components of the User table.

In this version everyone that was access to the web-service is able to create a valid user with ease. But this lays the ground work for the application of a administrator account that can give or refuse permission to access the rest of the Web-service.

4.3.1.3 Remote Accessing

The Server functions are accessible directly in the server through local host or remotely if the server is connected to a network that allows this function. As for the local approach can be unfeasible to work for a fluid data exchange. For that a way to communicate with the server had to be built in. In order to do that the Web-service needs to be hosted on a network that allows remote access. In order to accomplish that instead of inputting the local host on the URL the User inputs the remote address of the network and the port that the Web-service is allocated, this address is shown in Figure 4.11.

```
pi@raspberrypi:~ $ ifconfig

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.137.202  netmask 255.255.255.0  broadcast 192.168.137.255
        inet6 fe80::614d:4957:b854:9965  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:60:99:a0  txqueuelen 1000  (Ethernet)
        RX packets 8106  bytes 9433885 (8.9 MiB)
        RX errors 0  dropped 2  overruns 0  frame 0
        TX packets 4340  bytes 527827 (515.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 4.11: Local Host remote Address example

The remote access will substantially increase the amount of users accessing the Web-service. So the Server needs to be prepared for the onslaught of simultaneous sessions in a given time. For that a stress test is interesting in order to get the Server's reaction to several successive requests.

4.3.1.4 HTML & CSS display

This is more of a interface and graphic based feature. In order to simplify the UI, HTML and CSS (Hyper Text Markup Language and Cascading Style Sheets) was used. The HTML is normally used for Web-pages and complemented with the CSS for styling can become an intuitive interface.

With this interface allows anyone that knows how to use Web-pages to interact with the server. For that not a lot of background and training is required in order to operate. As an example of the output of this characteristic the Figure 4.12 shows the main window of Web-service. This page has the links to all of the operations available to the user, which include the 504 based exchanges, log out, return to the home menu and settings. All of these operations are seperate requests to the Server but are made in such a simple and seamless manner with the HTML and CSS.

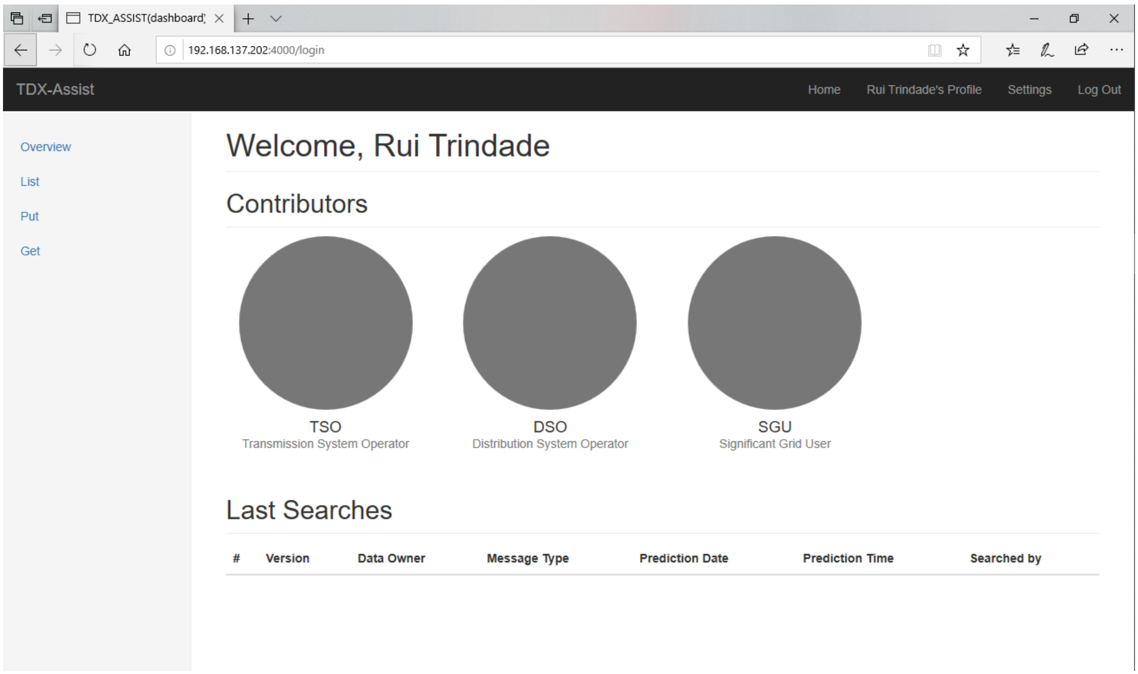


Figure 4.12: Dashboard Web-service page

4.3.2 Integrating the 504 protocol

As introduced in section 3.1.4.1 the 504 protocol is a standard of information exchange in Web-services associated with the energy market [15]. Since this thesis revolves around this exchange therefore this protocol should be the basis of communication. In order to successfully integrate the protocol the three main operations must be accurately implemented. Now for each operation the Server operates with a different dynamic in terms of the database and the information. This implementation using the RESTful Architecture only uses two of the four core requests, Get and Post. This is due to the information once stored in the server should not be erased but instead remove it from active status so that it is no longer visible [15].

4.3.2.1 Get Message

For this operation the user must input all the necessary fields in order to get the specific requested file. When the Server receives the validated submitted values from the user, it searches through the database in order to find the validated specific file. In the Listing 4.2 is the source code for the operation of the Get Message.

It is divided into two requests, Get and Post. Get is accessed when the user tries to enter the Get Message's page. Here is where a form asks the user the specific characteristics of the intended file entry. This form is part of a flask derived library that allows the creation of customisable forms, called FlaskForms. Also the Get function displays the HTTP page in order to visualise the form and every interaction possible with the user.

The Post is only called when the user clicks on the submit button present on the HTTP page. It verifies if the inputs are up to the form's standards if not it alerts the user. When the input from the user is valid the Server retrieves the active file (status=1) with the given attributes. In the eventuality that there is no entry in the database with the inputted information the Web-service just resets the page.

```
# Get Message

class download(Resource):
    @login_required
    def get(self):
        form = ExportForm()
        return make_response(render_template('export.html',
            form=form, name= current_user.username))
    def post(self):
        form = ExportForm()
        file_data = File.query.filter_by(status=1,
            dataowner = form.dataowner.data,typemsg=
            form.typemsg.data,datereference= form.Datereference.data
            ,dateSpan=form.DateSpan.data).first()
        print file_data
        if form.validate_on_submit() and file_data!=None:
            return send_file(BytesIO(file_data.data), attachment_filename=
               file_data.dataowner+file_data.typemsg+file_data.datereference
               +file_data.dateSpan+'.xml',as_attachment=True)
        return make_response(render_template('export.html',
            form=form, name= current_user.username))
```

Listing 4.2: Get Message Server responses

4.3.2.2 List Message

In this operation the Server must return the information in the database referring to the inquired entry. The Server must display only the file entries that are active (status=1) since those are the most up to date ones in the database. In Listing 4.3 is the source code of the Server's response to the user's interaction with this operation. As with Get Message this operation uses Get and Post.

The Get function works in a similar way as the Get function of Get Message. This being the initialisation of the form, which is a different one from the Get Message, and the display of the HTTP page.

As for the Post function it changes from the previous one. It is triggered by the same action as the Get Message Post function but that is the only similarity. Since this is a function to search in the database not all of the fields of the form need to be filled. In this search engine more than one result can appear, to be able to record all the outputs they are saved in a special deque array. A deque array is similar to normal python array, however, the deque has some helpful functionalities that the standard one lacks. The maximum size of the prediction is 100 results this defined by the maximum size of the array. Besides the single search there is another array that records all the last searches made by all users. this can be viewed in the Dashboard as a table with characteristics of the file searched as well as who searched them.

```
class search(Resource):
    @login_required
    def post(self):
        form = SearchForm()
        if form.validate_on_submit():
            if request.method == 'POST':
                comma =0
                strcode="File.query.filter_by("
                if form.Appdate.data!="":
                    strcode = strcode + "datereference=␣form.Appdate.data"
                    comma=1
                if form.ServerTimestamp.data!="":
                    if comma==1:
                        strcode= strcode+","
                    strcode =strcode +
                        "serverTimestamp=␣form.ServerTimestamp.data"
                    comma=1
                if form.id.data>0:
                    if comma==1:
                        strcode= strcode+","
                    comma =1
                    strcode = strcode + "id␣=␣form.id.data"
                if form.dataowner.data!="":
                    if comma==1:
                        strcode= strcode+","
                    comma =1
                    strcode = strcode + "dataowner␣=␣form.dataowner.data"
                if form.typemsg.data!="":
                    if comma==1:
                        strcode= strcode+","
                    strcode = strcode + "typemsg=␣form.typemsg.data"
                if comma == 1:
                    strcode= strcode+","
                file_data = eval(strcode+"status=1).all()")
                if file_data==[]:
                    return make_response(render_template('dashboard.html'
                        , name= current_user.username,val=len(last_pred),~
                        result=last_pred,user=user_search))
                pred.clear()
                for i in range(len(file_data)):
                    last_pred.appendleft(file_data[i])
                    pred.appendleft(file_data[i])
```

```
                    user_search.appendleft(current_user.username)
                return make_response(render_template('searchresults.html',
                    name= current_user.username,val=len(pred),result=pred,
                    user=user_search))


    def get(self):
        form = SearchForm()
        return make_response(render_template('search.html', form=form,
            name= current_user.username))
```

Listing 4.3: List Message Server responses

4.3.2.3 Put Message

Finally the Put Message is used to upload a file into the Server. The inputted file entry must be added with the complete information associated to it, and also according to the 504 protocol specifications. This is for the Server to be able to create a new element in the File Table of the database, and it cannot be accomplished with incomplete data. As with the previous two operations the Put Message uses Post and Get, this can be seen in the Listing 4.4.

As with the last operations the Get function is for initialising the form and displaying the HTTP page. The Post as stated before only acts if all the required fields in the form are correctly filled. When all the information is in the forms, the file in the assigned spot and the user presses submit the Post is able to correctly be processed. First the server creates directories in its memory in order to store the file. The file directories are created in the root of the server with the name of the type of message and inside of that one a directory with the date that the file refers to. After that the Server checks if there is another entry in the File table with similar characteristics, differing only in the Id and the version, if so changes the old entry status from 1 to 0. Following that change creates the new file entry with the information provided by the form. The file is saved in two places, as a binary file in the File Table and as the actual file in the created directory.

```
# Put Message
class upload(Resource):
    @login_required
    def post(self):
        form = ImportForm()
        if form.validate_on_submit():
            if request.method == 'POST' and 'photo' in request.files:
                if os.path.exists('/home/pi/projetos/start/Library/'
                    +form.typemsg.data)==False:
                # cria directoria typemessage
                    os.chdir('/home/pi/projetos/start/Library')
                    os.mkdir(form.typemsg.data)
                if os.path.exists('/home/pi/projetos/start/Library/'
```

```
                    +form.typemsg.data+ '/' + form.Datereference.data)==False:
                # Creates Datereference directory
                    os.chdir('/home/pi/projetos/start/Library/'
                        +form.typemsg.data)
                    os.mkdir(form.Datereference.data)
                app.config['UPLOADED_PHOTOS_DEST']=
                    '/home/pi/projetos/start/Library'+'/'+form.typemsg.data
                    + '/' + form.Datereference.data
                configure_uploads(app, photos)
                file_content= request.files['photo']
                now = datetime.datetime.now()
                old_f=File.query.filter_by(status = 1,
                    dataowner = form.dataowner.data,
                    typemsg= form.typemsg.data,
                    datereference= form.Datereference.data
                    ,dateSpan=form.DateSpan.data).first()

                if old_f != None:
                    old_f.status=0
                    db.session.commit()
                new_file = File(version = form.version.data status = 1,
                    dataowner = form.dataowner.data,
                    typemsg=form.typemsg.data,
                    datereference= form.Datereference.data,
                    dateSpan=form.DateSpan.data,
                    serverTimestamp= now.strftime("%d-%m-%Y %H:%M"),
                    data = file_content.read())
                filename = photos.save(file_content , name=form.dataowner.data
                    + form.typemsg.data + form.Datereference.data
                    + form.DateSpan.data + '.')

                db.session.add(new_file)
                db.session.commit()
                app.config['UPLOADED_PHOTOS_DEST']=
                    '/home/pi/projetos/start/Library'
                configure_uploads(app, photos)
                return make_response(render_template('dashboard.html',
                    name= current_user.username,val=len(last_pred),
                        result=last_pred,user=user_search))
        return make_response(render_template('import.html',
            form=form, name= current_user.username))

    def get(self):
        form = ImportForm()
        return make_response(render_template('import.html',
            form=form, name= current_user.username))
```

Listing 4.4: Put Message Server responses

4.3.3 Hardware Chosen

For the Hardware the goal was to make the software in a small and yet powerful enough computer that would allow for the Server to run without latency. Also it had to be easy to manipulate and mould to the imposed specification referred in Chapter 3. For that reason the chosen machine was the Raspberry Pi 3 Model B+, as shown in the Figure 4.13. It was chosen in order to serve as a Proof of concept.

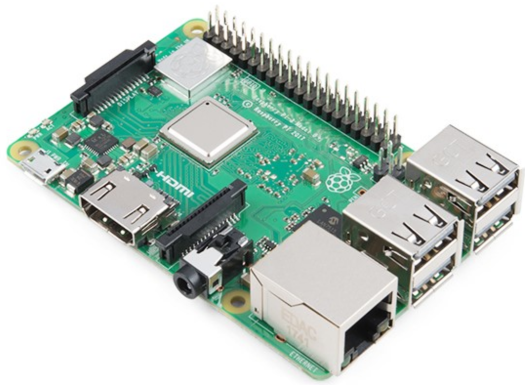


Figure 4.13: Raspberry Pi 3 Model B+

4.3.3.1 Specifications & Benchmarks

At the time of this implementation, the Raspberry Pi 3 Model B+ is the latest in the market. With this it garners a great performance advantage when comparing with the previous model, the 3 [18]. This micro computer has great specifications for its size and well optimised. Its specifications are displayed in the Table 4.1.

Table 4.1: Raspberry Pi 3 Model B+ Specifications [18].

	Specification
SoC	Broadcom BCM2837B0 quad-core A53 (ARMv8) 64-bit @ 1.4GHz
GPU	Broadcom Videocore-IV
RAM	1GB LPDDR2 SDRAM
Networking	Gigabit Ethernet (via USB channel), 2.4GHz and 5GHz 802.11b/g/n/ac Wi-Fi
Bluetooth	Bluetooth 4.2, Bluetooth Low Energy (BLE)
Storage	Micro-SD
GPIO	40-pin GPIO header, populated
Ports	HDMI, 3.5mm analogue audio-video jack, 4x USB 2.0, Ethernet, Camera Serial Interface (CSI), Display Serial Interface (DSI)
Dimensions	82mm x 56mm x 19.5mm, 50g

The more relevant test to take under consideration is the CPU and system performance. This test is meant to measure the CPU's speed while processing a given task, either using one of its cores or several.

Possesses a 1.4GHz CPU, that has a a similar structure to the older model the 3B+'s SoC, System on Chip and houses new features that increase its performance. This is due to the improved packaging and heat-spreader installed in the plaque. A stress test made to all the models in the Raspberry Pi family highlights their performance and resource use, as well as the advantages of multiple core processing, as shown in the Figure 4.14. The stress test used in this graph was the SysBench CPU a software specialised in these kinds of speed and stress tests [19]. In this graphic is displayed all of the different Raspberry Pi models are test results in terms of the time taken to process a given set of tasks by SysBench. The time the CPU's take to process the data is measured in seconds. In the later models of Raspberry Pi the option of Multi-Threading is available and it is also tested. It is experienced an increase in performance especially in the later generations. However, it is still slower than the average machine, which get result in this test 20 to 18 seconds [19].

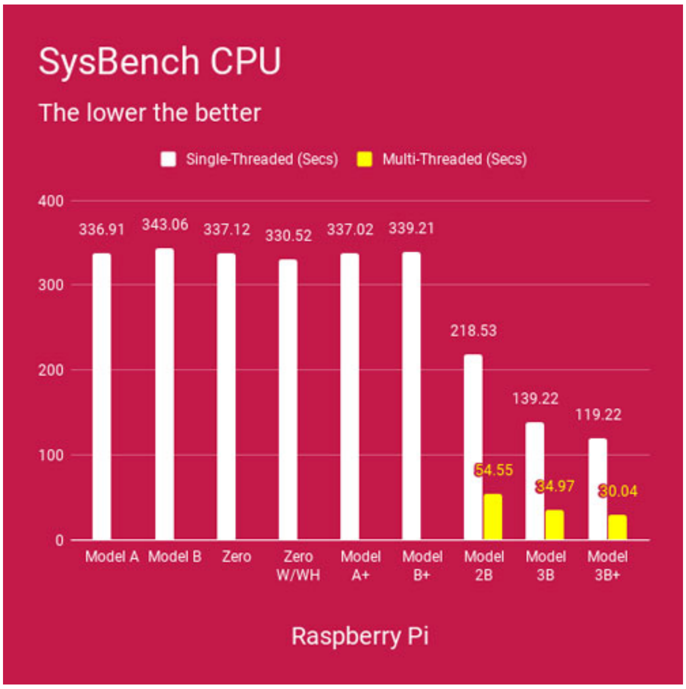


Figure 4.14: Raspberry Pi System Benchmarks [18]

4.3.3.2 Application

The specifications referred in the Subsubsection 4.3.3.1 are pretty sub-par when comparing to a current computer or even a Server so this will serve as a basis test. This is exactly why the Raspberry Pi is a great hardware piece to create the prototype of the Server.

So if the Web-service would successfully work in this machine in terms of information exchange and simultaneous connections, a more powerful machine would also be able to.

The Raspberry Pi is a very versatile machine that can be programmed in different ways [18]. In this micro-computer to be able to operate it the OS, (Operating System) Raspbian was installed. Raspbian is a Linux based OS that is highly optimised for the Raspberry Pi [20]. Installation is straight forward and this OS already has several IDE (integrated development environment) installed. The IDE used to develop the Web-service was Geany.

Besides being a good way to test the Web-service on a micro-computer the implementation on the Raspberry Pi also serves other purposes. For example, by developing in a separate machine it is more obvious to understand the Client and Server dynamic and structure. Also by being two different machines serving as Client and Server it automatically requires a way to connect both of them. Something that might not be emphasised enough if both the Client and the Server resided in the same machine.

Chapter 5

Case Studies & Results

This chapter will be focused on the testing of both the Client and the Server APIs, which APIs were described in Chapter 4, and show the result of them both. The Client section will describes the grids that shall be used to test the API. This description will include the topology of the grid, the elements present and their behaviours in different hours of the day. As for the Server it will touch upon how to access the web-service and its functionalities, giving a specified explanation on how to replicate them step by step. In order to test the Server a stress test was made to evaluate its performance. In the end the results and the methodology of each test will be described.

5.1 Case Studies

To test the Client's API there were two types of grids, a small scale grid and a large scale one. The first serves to tune all the installed functionalities in the API, the large scale grid is to test the API's scalability. For the small scale case, the grid used was based on the IEEE 14 grid, and the large scale has a similar dimension of the Portuguese grid. A 24 hour time-frame was developed in order to capture the progression of the I_{cc} in a typical operational planning period.

5.1.1 IEEE 14 - based

As stated before the grid used for the small scale testing is the IEEE 14 grid. The grid used has a simple structure perfect for the analysis that is intended to be accomplished, the resulting grid can be visualised in Figure 5.1. However, some alterations from the base IEEE 14 grid were made because of the focus on the relationship between TSO and DSO, the grid was modified in order to reflect this. Which means that all 150kV bus bars in the original grid were changed to 60kV ones. To simplify the area between buses 4,7 and 8 two transformers were kept between buses 8 and 4 and a simple line between 8 and 7. This simplification is the reason for the grid only having 13 buses.

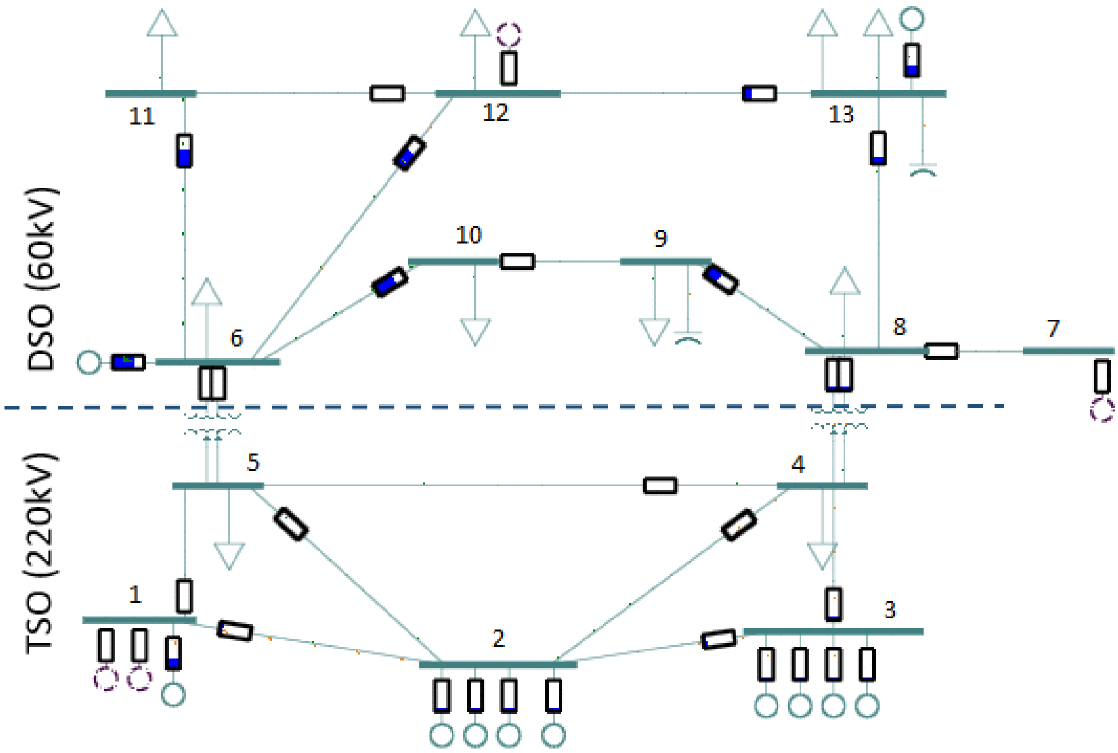


Figure 5.1: IEEE 14 Grid.

5.1.1.1 Grid Topology

With the IEEE 14 based grid defined as the topology used, the grid's elements can be defined. This encompasses the line, transformer, bus, load, and generation characteristics and their rational placement in the assigned place. For this grid's elements the used profiles that represent the values enunciated previously (transformers, lines, loads and generation) were based on real data retrieved from REN [21] [22]. Afterwards, this data was adapted to the IEEE 14 gird. In this section the information regarding all the grid's element characteristics is developed and explained, it is further detailed in Annex II.

Typical values for the 60kV and 220kV lines considered. In table 5.1 is shown the adapted values for each voltage level. The first set of lines are the ones with a base voltage of 220kV and the second set for 60kV. The same approach was used for the modelling of the transformers. Typical values used are shown in Table 5.1. The line values in the Table represented by R, X and B refer to the values of Resistance, Reactance and Susceptance present in each line (measured in p.u.), these characterise the amount of power loss when energy crosses each line. The values R, X and B are dependant on the Length of the line itself (measured in Km). To the Line is also attributed a power rate (measured in MVA), this defines the amount of apparent power that is allowed to travel in a line at a given time. As for the transformer the power rate is also defined (measured in MVA), as well as the short circuit impedance which defines the amount of power loss in the transformer.

Table 5.1: Line and Transformer Characteristics.

Line	R(pu)[a]	X(pu)[a]	B(pu)[a]	length(Km)	Rate(MVA)
Line 1-2	0.0192	0.1056	0.1548	120	316
Line 1-5	0.04	0.22	0.3225	250	316
Line 2-3	0.036	0.198	0.29025	225	316
Line 2-4	0.032	0.176	0.258	200	316
Line 2-5	0.03168	0.17424	0.25542	198	316
Line 3-4	0.03104	0.17072	0.25026	194	316
Line 4-5	0.00768	0.04224	0.06192	48	316
Line 6-10	0.08322	0.23553	0.01288	52	60
Line 6-11	0.10704	0.30293	0.01656	67	60
Line 6-12	0.05450	0.15427	0.00843	34	60
Line 7-8	0.03719	0.10526	0.00575	23	60
Line 8-9	0.03535	0.10006	0.00547	22	60
Line 8-13	0.11313	0.32019	0.01750	71	60
Line 9-10	0.08037	0.22745	0.01243	51	60
Line 11-12	0.08363	0.2367	0.01294	53	60
Line 12-13	0.11658	0.32970	0.01802	73	60
Transformer	X(%)			Rate(MVA)	
Transformer 4-8	10			150	
Transformer 5-6	10			150	

[a] $|S_{base}| = 100MVA$, $|U_{base}| = 220kV$, $|U_{base}| = 60kV$

In order to illustrate the possible evolution of I_{cc} and S_{cc} through a time frame the following types of generation and loads were considered:

- **Bus 1 - Thermal Power Plant.**
- **Bus 2 and 3 - Hydro Power Plant.**
- **Bus 4 and 5 - 220kV substation busses, load of EHV costumers**
- **Bus 6 - 60kV substation bus with Wind powered generation, load of a HV costumer**
- **Bus 7 - Solar power plant**
- **Bus 8 - 60kV substation, load of a HV costumer**
- **Bus 9 - Load of an Industry sector**
- **Bus 10 and 11 - Load of a Residential sector**

- **Bus 12 - Prosumer (Solar) of a residential sector**
- **Bus 13 - Wind powered generation and loads of an Industry sector**

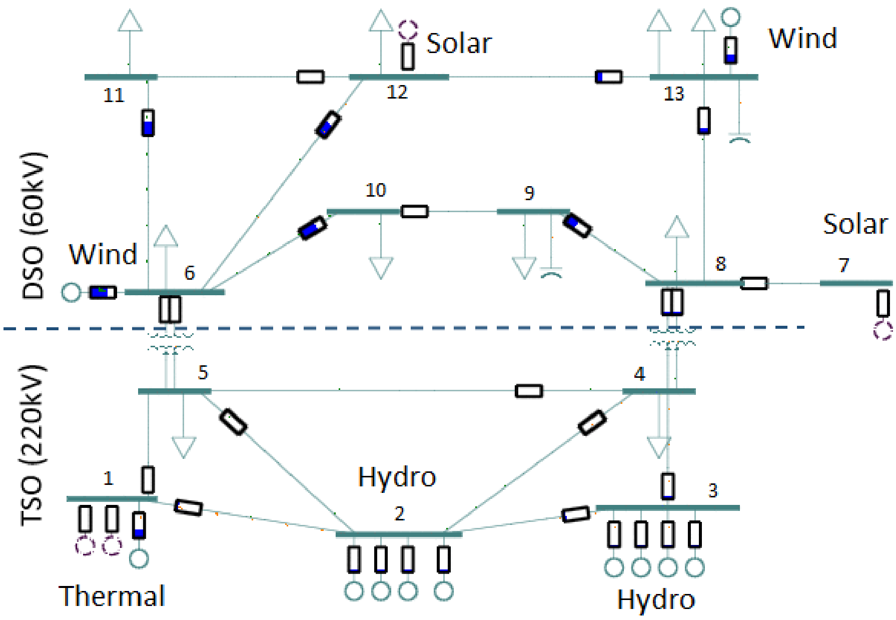


Figure 5.2: IEEE 14 Grid, with discriminated generations.

The profiles that are going to be presented in the following sections, took in consideration real data cases occurred in the Portuguese energy system. As referred previously there were 24 cases developed with this topology, one for each hour of the day.

5.1.1.2 Energy Production Profiles

The chosen energy production power plants were thermal, hydro, wind and solar which are the most relevant in the Portuguese energy market. To get a representation of a close to real life case the profiles of energy production were retrieved from REN [21]. After retrieving the energy dynamic the values were adjusted to fit in the grid's reality.

- **Thermal Energy Production**

  This energy source comes from the burning of fossil fuels, such as oil, gas and coal as the main components. This production is associated with bus 1, the reference Bus, it has 3 machines of 80MW for a Power plant with a total production of 240 MW of active power available. By having three machines there is some control in order to accompany the need of more production by the loads.

- **Hydro Energy Production**

  This type of resource is one of the main producers of energy in Portugal. Their inclusion is important since a good portion of the country's energy is due to Hydro power. Normally, Hydro power due to being built on large dams the injection point in the grid usually is in the EHV grid, therefore the TSO.

These in the IEEE 14 grid are represented in the bus bars 2 and 3, these are located in the TSO grid, as shown in 5.3. In terms of the productions profiles they were retrieved from the REN [21]. The two profiles used in specific were from two different dates, one being from January 8^{th} 2017 and the other one from January 7^{th} 2017. Both of these profiles are displayed in the Figure 5.4 below being Figure 5.4a referent to Bus 2 and Figure 5.4b referent to Bus 3. To adapt these productions to the grid 4 turbines were installed in each Bus bar, each 50 MW of power, totalling 200 MW of power produced at the peak of these profiles when the 4 turbines are active.

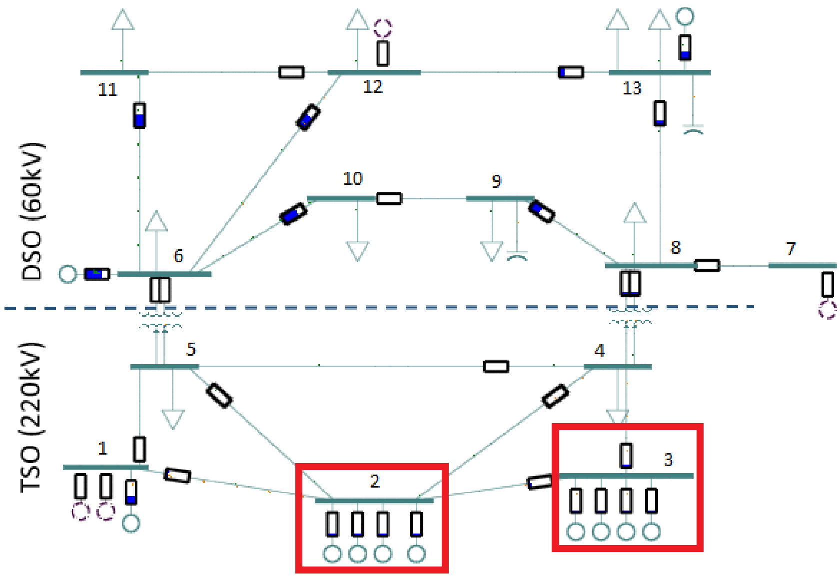
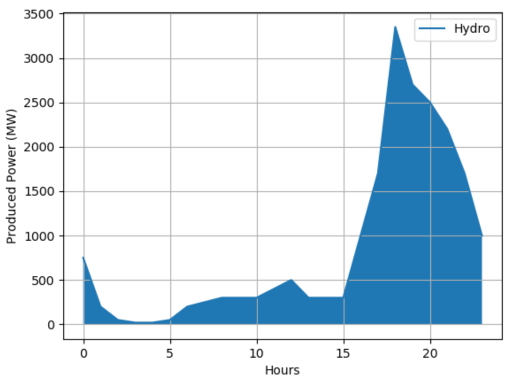
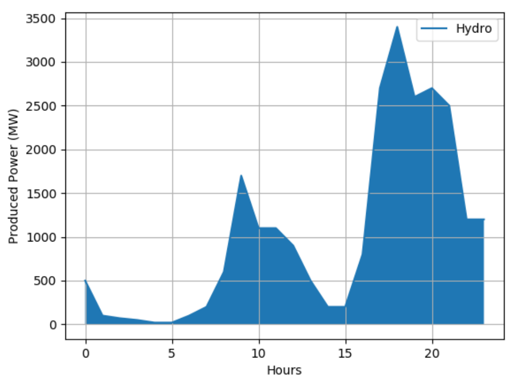


Figure 5.3: Hydro Production Buses in the IEEE 14 grid [21]



(a) Hydro Production for 08/01/2017



(b) Hydro Production for 07/01/2017

Figure 5.4: Hydro Energy Production [21]

- **Wind Energy Production**

  This type of generation is common to produce great amounts of energy, and instead of being connected to the EHV grid this type of energy production usually is connected to the MV or HV grids [23]. Also most of these installations are made on shore where the potential wind power is higher. Since Portugal is a country with a great coast area the use of wind farms is advantageous, as it can produce substantial energy amounts[23]. This explains its importance of this type of production and the reasoning why it is incorporated into the developed grid. Besides that due to the size that each wind turbine needs to be in order to effectively generate electrical energy and the amount of noise produced by it renders it impossible to create a wind farm near a residential area [23].

  As for the implementation in the actual IEEE 14 grid, some measures were made to reflect the closest paradigm to the reality. These included the installation of the wind farms in the HV grid and favouring the proximity with industrial loads. As the Figure 5.5 shows the bus bars that the wind farms are connected to, are bus 6, close to the TSO/DSO border and in the bus 13, associated with two industry sector loads. As for the basis for each bus production the Figure 5.6 show the profiles hat were considered. The wind energy production referent to January 9^{th} 2017, as shown in Figure 5.6a ,is used for bus 6 and the energy profile from January sixteenth 2017, as shown in Figure 5.6b, for bus 13. For the adaptation to the grid the generation was changed to a 160MW generation in bus 6 and 155MW in bus 13.

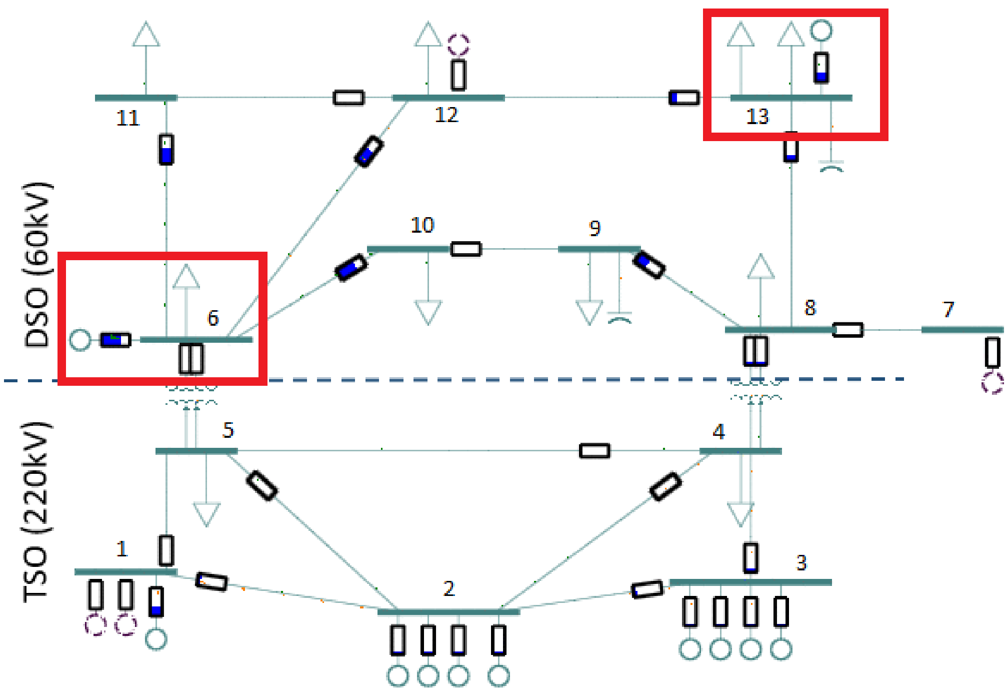
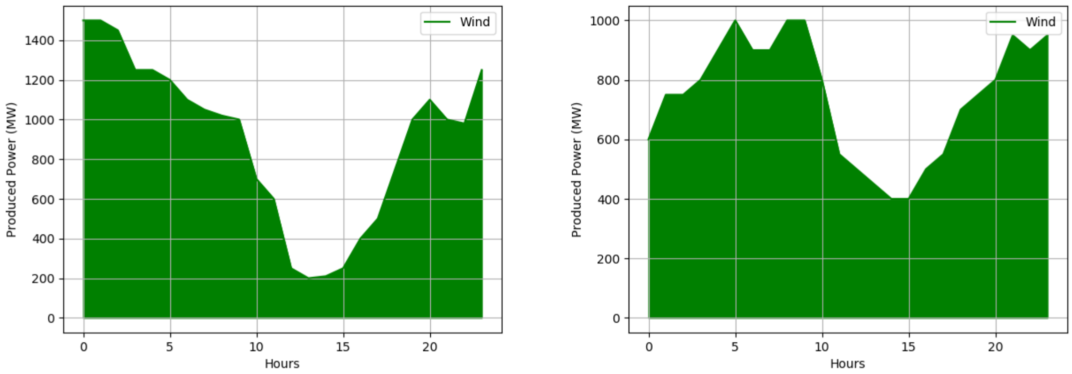


Figure 5.5: Wind Production Buses in the IEEE 14 grid [21]



(a) Wind Production for 09/01/2017 (b) Wind Production for 16/01/2017

Figure 5.6: Wind Energy Production [21]

- **Solar Energy Production**

  Due to Portugal´s sun exposure the solar energy is a viable way of energy production. Since solar power is one of the most common resource that exist on earth, harnessing it brings extreme benefit to the energy landscape [13]. However, the energy production is dependant on more than just the irradiated energy from the sun. There are factors like the weather, time of the year, condition of the photo voltaic panels that determine the output of energy.

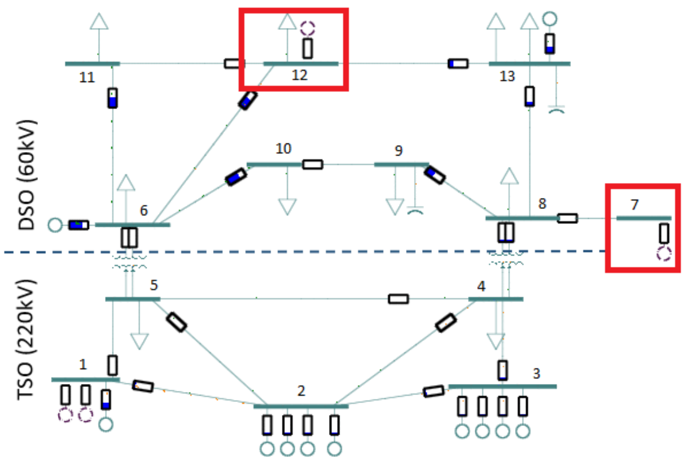


Figure 5.7: Solar Production Buses in the IEEE 14 grid [21]

In IEEE 14 grid there are two instances where solar power plants are installed. One as a actual power plant in bus 7, the other associated with a residential load in bus 12. The position of these two buses in the IEEE 14 grid are represented in Figure 5.7. In the second one the energy production is not considered as a power plant but as a Prosumer, a consumer that also produces some electrical energy. This is to simulate the growing presence of this prosumers in the energy market, either having NZEB

(Nearly Zero-Energy Building) or just with one simple module installed. The energy production profile characteristic of these types of power plants are represented in Figure 5.8, the plants are only active to production in the period where there is sun shine. The solar profile was taken from the REN [21] for January 7th 2017.

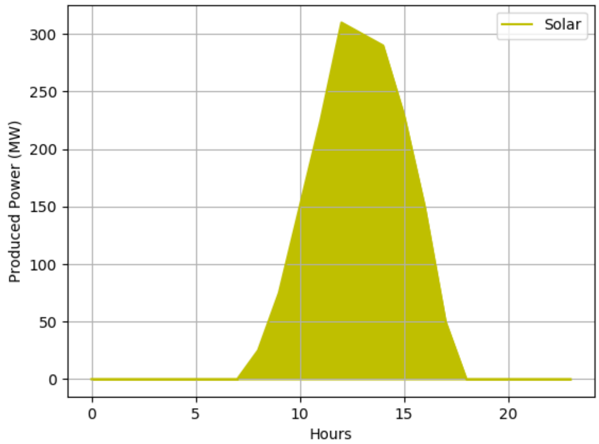


Figure 5.8: Solar Energy Production for 07/01/2017 [21]

5.1.1.3 Load Profiles

What was done with the generation profiles was done to the load profiles as well. The basis of the profiles was retrieved through the REN [22]. The information retrieved has a daily structure where depending on the day presents different profiles. For that there was some screening needed in order to find the load profiles adequate to the intended purpose. As it was detailed above there are two types of loads, Industrial and Residential, both with distinct profiles.

- **Industrial Load**

  The normal profile of an industrial load presents a peak in consumption during labouring hours. In those hours the consumption makes a plateau, and when the labouring period ends there is a swift decline. This type of profile was the standard for the industrial loads for the IEEE 14 grid.

  For the EHV the loads of the buses 4 and 5 have a distinct profile. Since normally the costumers that connect to the EHV grid are from an industrial setting. The bus 4 load profile was retrieved from January 6th 2017 and the one for bus 5 from January 9th 2017.

  As for the HV bus loads the amount of them in the grid are substantially greater than the EHV counterpart. The industrial loads in the HV grid as referenced previously are situated in the buses 6, 8, 9 and 13. As for the EHV loads, each of these loads has a particular daily profile, all of them follow the industrial load standard profile.

For the dates of origin the Bus 6 matches the profile from January 4^{th} 2017, Bus 8 January 5^{th} 2017 and Bus 9 January 3^{rd} 2017. As for the Bus 13 it has two loads, the first one is represented in the Figure 5.9 referent to January 2^{nd}2017, and the second load from January 14^{th} 2017.

For the EHV loads the maximum was established at 20MW, as for the HV ones the maximum is 45MW. This difference between EHV and HV was made in order for the power flow of the grid to follow a cascading effect, going from the TSO grid to the DSO grid.

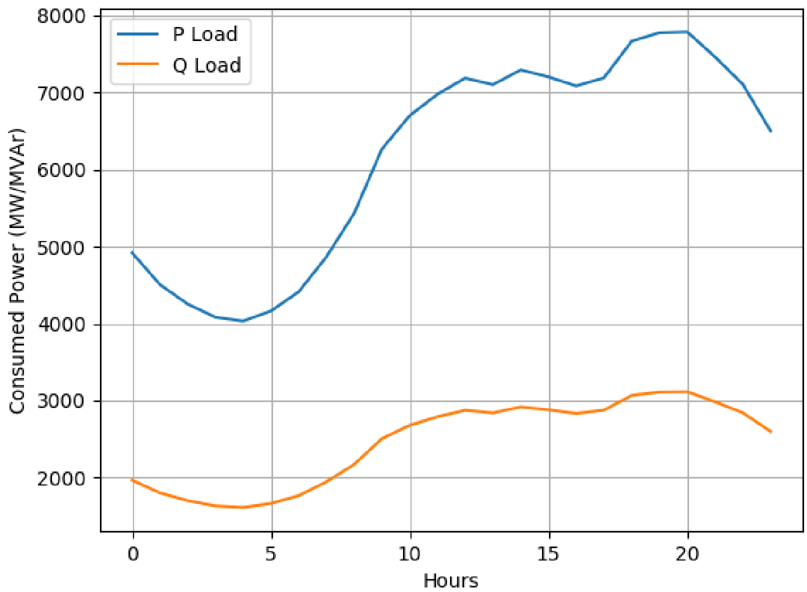


Figure 5.9: Industrial Load Profile from 02/01/2017 [22]

- **Residential Load**

  These types of loads like the industrial ones also have a somewhat defined profile. This is defined by two peaks, one in the morning and one in the evening being the evening one more noticeable. Therefore all buses with residential loads follow this profile.

  These types of loads only exist in the HV grid, the buses that are in this type are bus 10, 11 and 12. To the likeness of the industrial all profiles were obtain from different days from the REN website [22]. Bus 10 was obtained from January 1^{st} 2017 and is the one displayed in Figure 5.10. Bus 11 is referent of January 7^{th} 2017 and Bus 12 from January 8^{th} 2017. In terms of adaptation for the IEEE 14 grid it follows the same scheme as the HV Industrial loads.

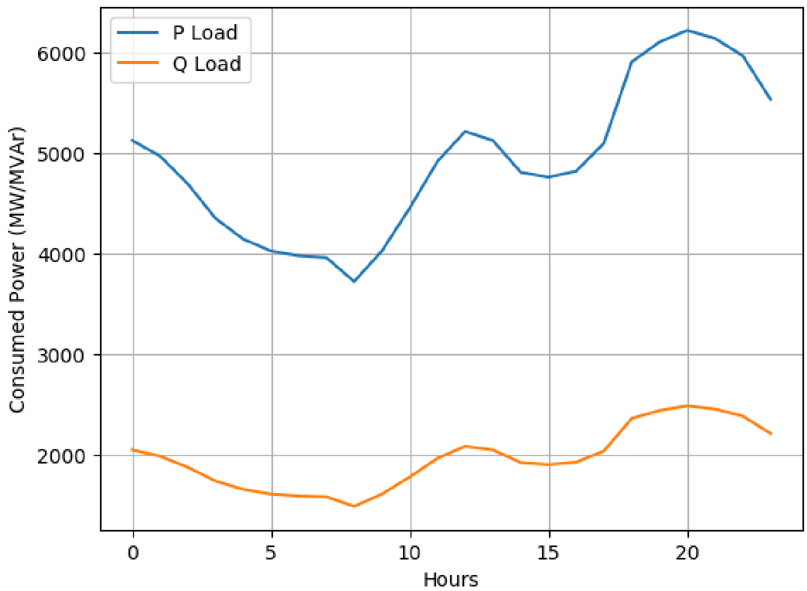


Figure 5.10: Residential Load Profile from 01/01/2017 [22]

5.1.1.4 PSSE Case file creation

The profiles developed in the previous section only took in consideration the active power but the bus loads require also the reactive as well. For that purpose a constant power factor was considered such as $\tan\phi = 0,4$.

To ensure that the retrieved information amounts to a viable case all of the values obtained were from the same time of the year in the same year. This was made also to create a solid set of 24 cases, in this case represent the grid of a winter consumption and production. This detail is important to ensure some structured guideline in the grid's overall dynamic. It would not be correct to have the hydro and wind production profile from winter and the solar from summer due to the differing dynamics along the year.

As for the total sum of the produced energy, it does not fit the profile of all the consumption in the grid. For that reason some control on the produced energy injected to the grid needs to be put into place in order to more accurately fit the consumption profile. And with that a priority in which to turn on or off some generation plants needs to be defined. The grid's production profile in all of its extent requires for it to be greater by a margin, in order to take under consideration any loss that might occur in the lines and transformers present in the grid.

In terms of power injected into the grid the only plants that allow the control of machines operating are the Thermal and Hydro. Both Wind and Solar inject directly the total amount that is being produced without interruption. These two energy sources become the first priority in injecting power in the grid. However, the Hydro and Thermal machines should only be turned on and off for extended periods of time to prevent cases where the machine is turned on for just one hour. This condition is put into place to ensure the longevity and efficiency of the machines involved. After the injection of the Wind and Solar power the next to be connected are the Hydro power generators, and only in last case scenario the Thermal generators are to be turned on. All of this allows some leeway in terms of adapting the production to the consumption, in Figure 5.11 the discriminated planned energy production is displayed. It shows the dynamic of every power plant and the final production profile.

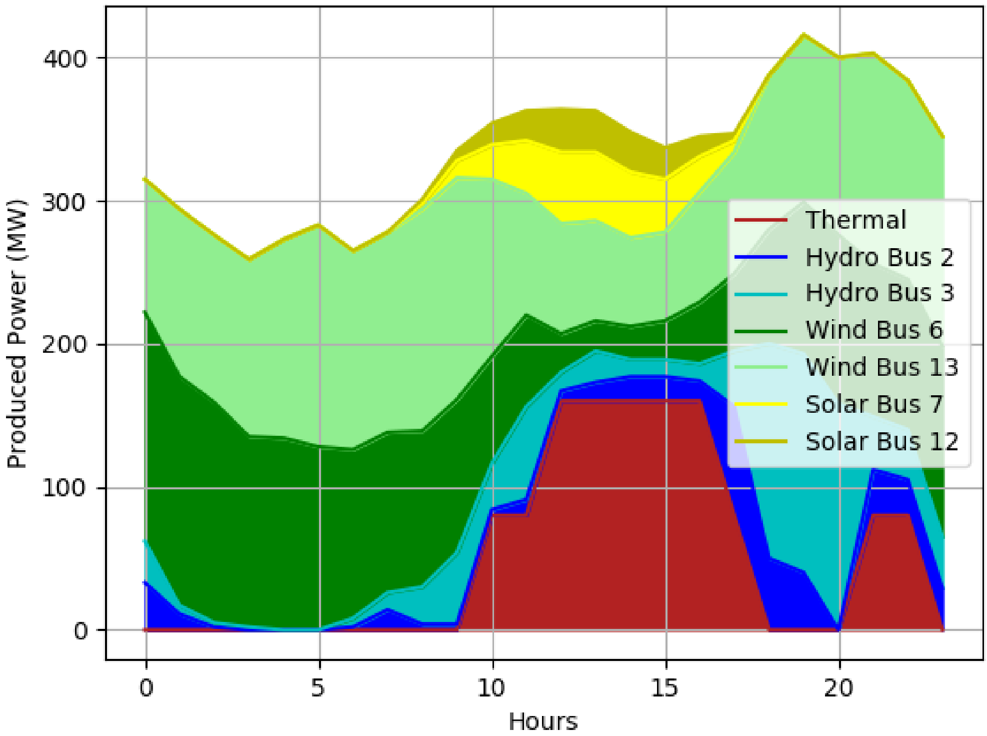


Figure 5.11: Energy Production Profile Diagram

The target of the production profile was to follow the consumption profile and having a little margin for losses that occur in the grid. These losses happen in the lines, transformers and possible other discrepancies. For the amount of this margin it was established at a maximum of 3% of the consumed energy, any higher and the energy would not be effective. In Figure 5.12 is a graph representing both production and consumption. This was the best approximation possible in order to get the values in the margin. However, there are instances were the margin is big and exceeds the 3% mark, there were only two instances out of the 24 that registered this. These instances are in hours 4 and 5 where the margin has a values of 10,6% and 13,8%.

Now that the all the values are adjusted, they can be imported to the PSSE software for the simulation of the power flow. The hourly modelling should represent feasible

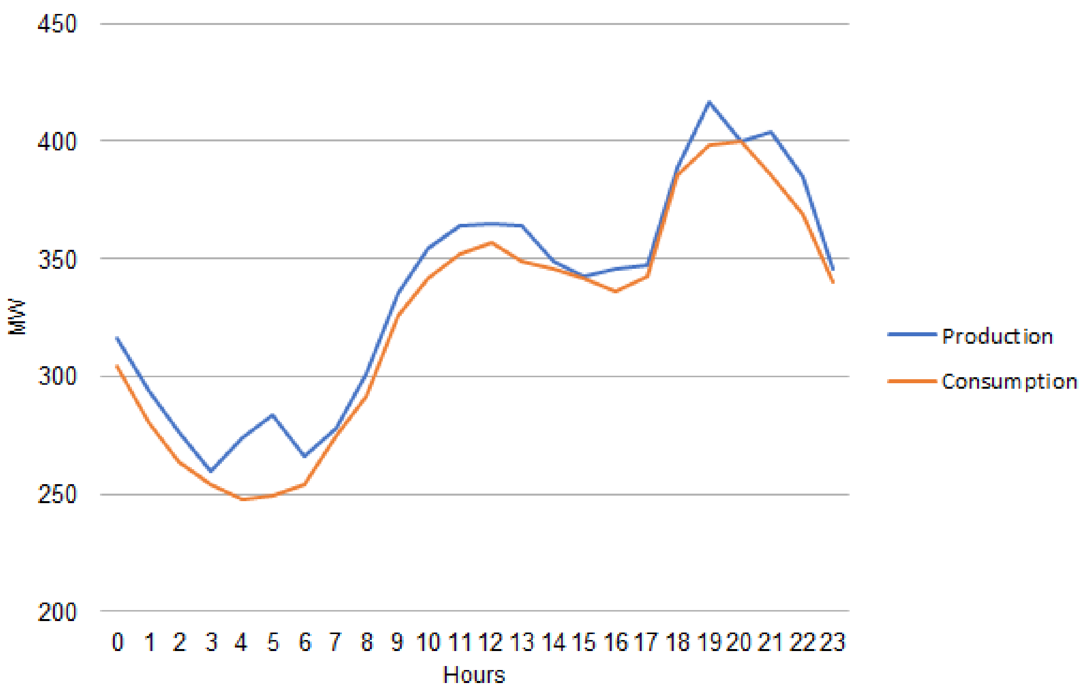
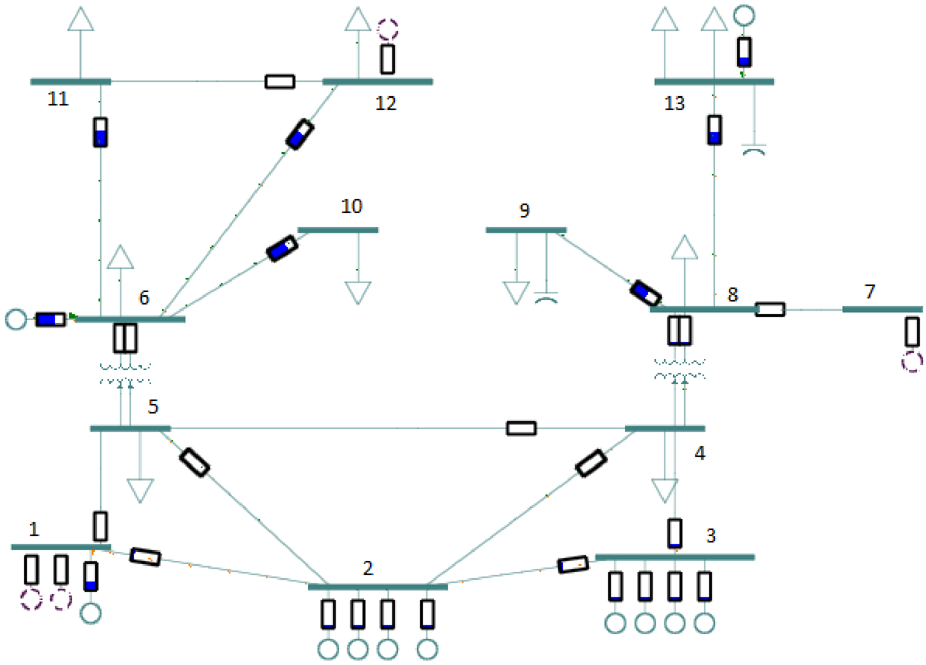


Figure 5.12: Production Vs Consumption of the IEEE14 grid

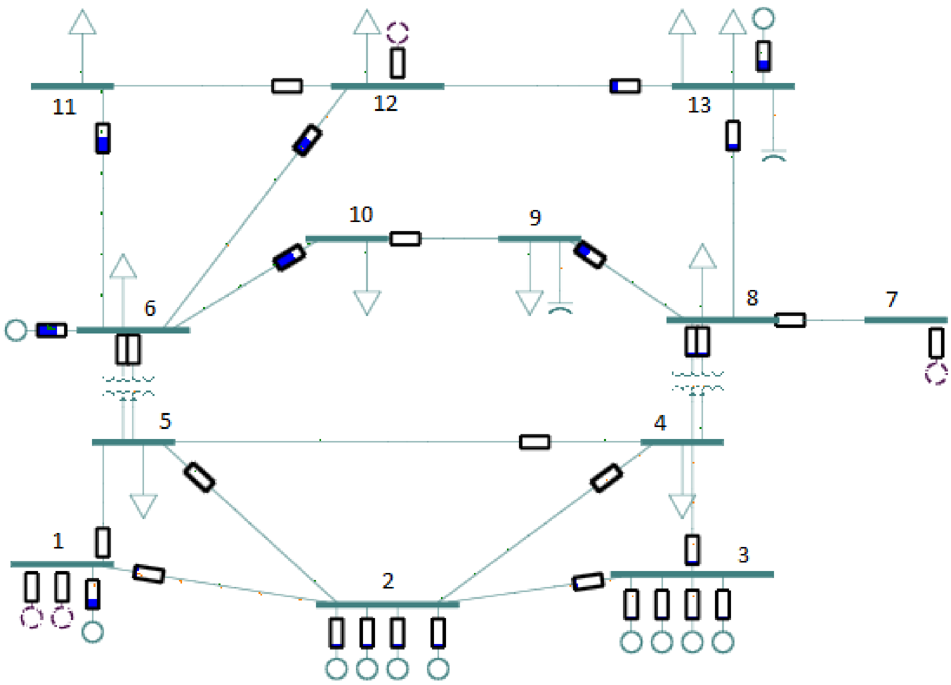
scenarios, thus, not presenting overloads in any of the periods. In those standards there are maximum and minimum values for the voltage in the bus bars and the maximum of power rate in lines and transformers. The range of voltages in the bus bars must be between 1,05 and 0,95 p.u. on each bus. In terms of the rate it only has a maximum stipulated in each line and transformer, as referenced in Table 5.1. If these limits are not crossed the case file is acceptable to carry out through the rest of the process. More information is present in Annex II.

The loss of this energy directly results on the lowering of voltage, as a practical option banks of capacitors were installed in two buses. The first one was the bus 9 where there were more reactive losses and the lowest voltage, The second one was the bus 13 which represents the greatest reactive charge due to being connected to it two loads. The value of each capacitor bank reactive power output in each bus is the same (20MVAr). This addition was not needed in all of the case files, however, they were added in order to have the same structures in all grids.

Besides the 24 hours operational scenarios two additional topological scenarios were developed. One that map a more radial structure and the other is more meshed. In order to create this radial grid two lines were removed the one connecting bus 9 and 10 as well as the one connecting the bus 12 and 13. The two grids that will be analysed are represented in Figure 5.13.



(a) Radial Version of the IEEE 14 Grid



(b) Meshed Version of the IEEE 14 Grid

Figure 5.13: Different Topological Scenarios

5.1.2 Large Scale

For this large scale example a more complete grid was taken from the REN database. This information is provided yearly in their report regarding the Portuguese transmission grid. In this report the detailed characteristics of the bus systems under their responsibility. With this report in mind the PSSE file representing the Portuguese grid for the test of the API was created. This resulted in a good depiction of a real-life case of the Portuguese grid and a good example to test the scalability of the Client API. This grid possesses 1126 bus bars, 300 individual loads, 386 power plants and 1401 branches, categorising as a substantially larger grid than the IEEE 14. For this gird 13 cases were created, however, these do not have the hourly dynamic that the small scale possess. This solution was also incorporated in the EU's project TDX-Assist that has the same goal.

5.2 Server Interface

With the internal Server functions and the messages structure defined previously, now its explored the User's view when interacting with the Server. This will serve as a guide in how to operate and navigate the Server. For that four functionalities will be explored User creation, Get Message, List Message and Put Message. Each one of these represents a requirement imposed by the regulamentation elaborated in Chapter 3. Being the user creation more in order to create a layer of protection of the server, while the other three are to emulate the operations of the 504 protocol.

5.2.1 User Creation

In order to access the Server there is a need for the user to log in with a usable user in the database. For a first time user there is a need to create a new user entry in the database. First the user needs to access the main page of the Web-Service, as illustrated in Figure 5.14, in order to create a new user entry the highlighted button should chosen.

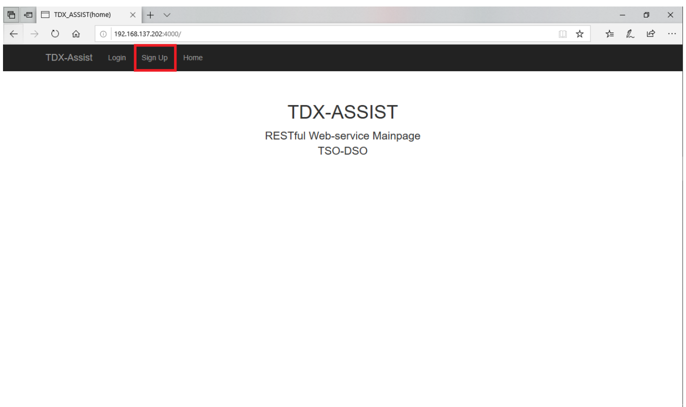


Figure 5.14: Step 1 - Home Page

With this the sign up page should appear this is where the user can create a new user entry. As shown in Figure 5.15 these are the necessary fields that the user must fill, username, email and password. The username can be any possible collection of characters with a minimum of 4 and a maximum of 15. All user names must be unique with that there can not be two equal user entries with the same username. In the email field the user must input a valid unique email address with a maximum of 50 characters. Finally the password field, as other websites, is protected while the user types the password, this field must be between 8 to 80 characters long. To ensure that the password stays private all the input of the password in the database is encrypted. After filling all the fields pressing the submit button will, if all fields are properly filled, create a new user entry.

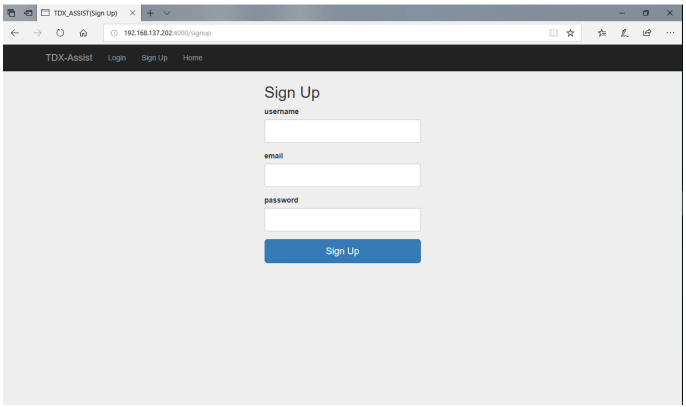


Figure 5.15: Step 2 - Sign up Page

After the creation of the the new user entry the log in can be completed by inputting the username and the password. The Log in screen is shown in Figure 5.16, after passing this page the user can access the actual Web-Service functionalities. These three pages that are displayed here are the only one in the Web-Service that do not require the user to be logged in.

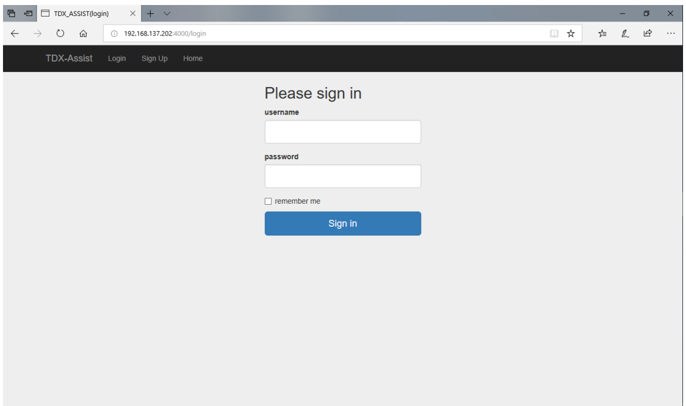


Figure 5.16: Step 3 - Log in Page

5.2.2 Get Message

After accessing the main page the 504 functionalities can be used, firstly the Get message function. To reach this functionality the user should press the Get name in the Dashboard, as shown in Figure 5.17.

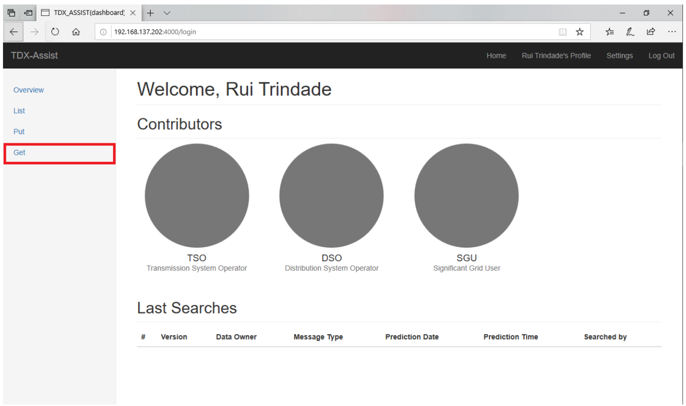


Figure 5.17: Step 1 - Access Get Message Page

After pressing the link the Get page appears with a new form to fill. All the fields of the form must be fill in their entirety in order to find a specific file entry and download to the user's machine the XML file in the searched file entry. If the user inputs the correct information the file is downloaded. However, if the information provided by the user is not contained in the database as the user presses submit the page will reload and all inputs will be turned blank.

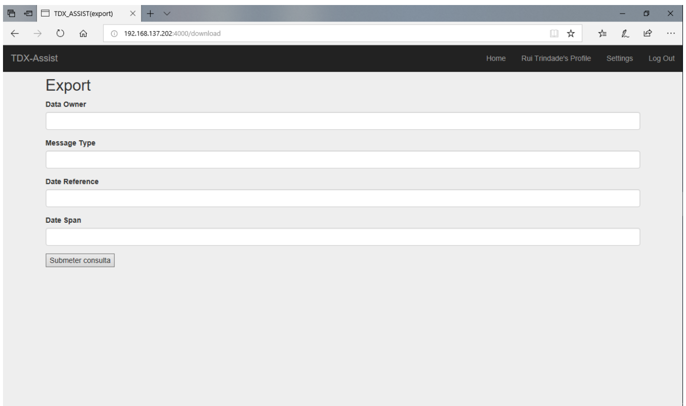


Figure 5.18: Step 2 - File Entry Inquiry

5.2.3 List Message

In order to search the database for the active files in it the List functionality is what the user must rely upon. To reach that functionality the List link in the Dashboard must be interacted with, as highlighted in Figure 5.19.

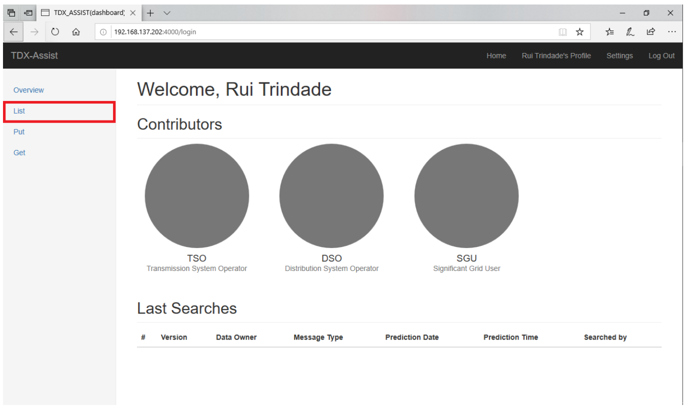


Figure 5.19: Step 1 - Accessing List Message

After accessing the search page the user is prompted by another form. However, this one is different from the one in the Get Message since none of the fields are mandatory. This results in the user being able to search one specific active file entry or a broader scope within the active file pool. The Figure 5.20 shows the form and the usable search topics, Date of Reference, Server Time stamp, File ID, Data Owner and Message Type.

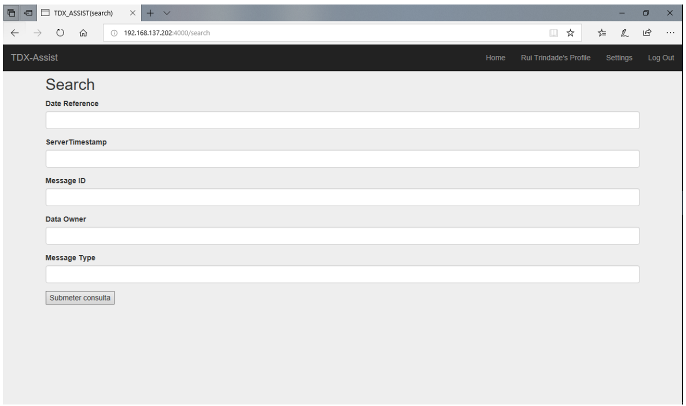


Figure 5.20: Step 2 - File Search Inquiry

After pressing submit if there are search results they are promptly displayed as shown in Figure 5.21. The displayed data in the table is the relevant information defined in the 504 plus some extra ones, these being number of search, file version, Data Owner, Message Type, Prediction date, Prediction time and the User that issued the search. Although this displays multiple search results they are limited by 100 since that is the size of array that contains the searched information.

When the user returns to the dashboard after a search there are some differences in the output of this page. These differences can be spotted in the lowest part of Figure 5.22, where the output of the search is displayed. This display shows the last 100 searches made by all users, this functionality was added to prove that all users are connected to the same platform.

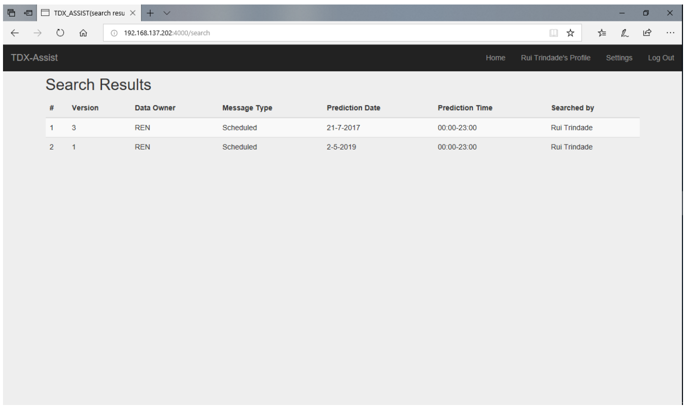


Figure 5.21: Step 3 - File Search Output

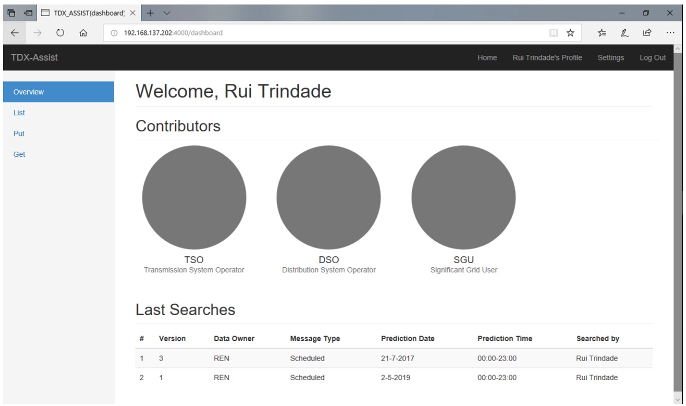


Figure 5.22: Step 4 - Output of search in the Dashboard

5.2.4 Put Message

Finally the last functionality, Put Message creates a new file entry in the database. Like the other two the user to gain access to the page needs to interact with the link represented in Figure 5.23.

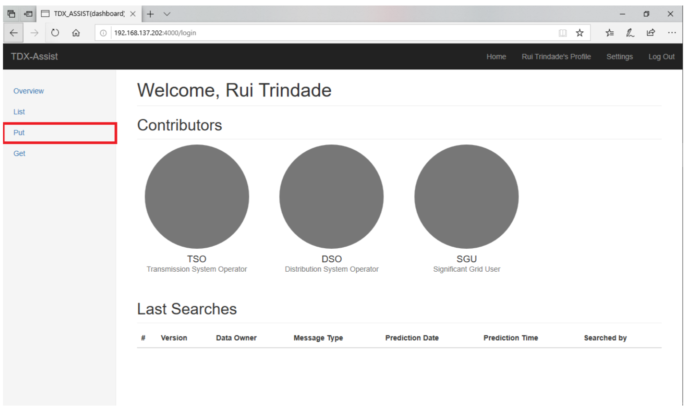


Figure 5.23: Step 1 - Accessing Put Message

When the Put Message page appears, shown in Figure 5.24, the form where the user can input the characteristics of the file that is going to be imported. All the characteristics are required to be filled in order to create the file entry. The user must input the Version, Owner, Prediction date and time of the file as well as the actual XML file. When the user fills all the fields and presses the submit button the Web-Service automatically sends the user to the Dashboard.

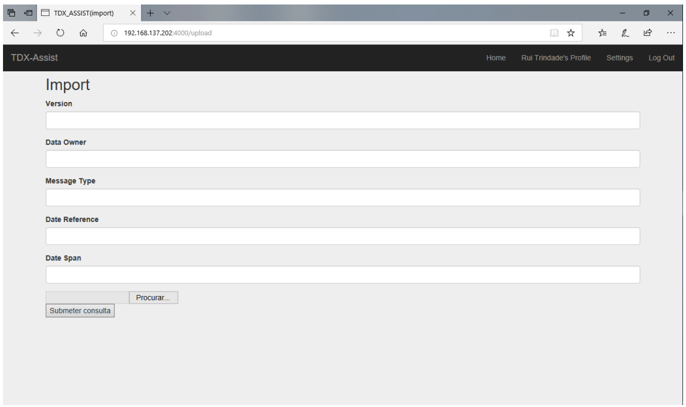


Figure 5.24: Step 2 - File Import Inquiry

5.3 Results & Validation

After discussing each of the API's characteristics and resources used for testing them, its time to create a environment where the they can be tested and the results can be verified. As with the separation of Client and Server the testing will be done separately. This is made in order to test the performance of both separately and then analyse if the results of the tests are satisfying enough for Client and Server to work together.

5.3.1 Client API

For this test the objective is to lay out the results obtained from the API and making a profile of the short circuit currents. The results will focus on the previously mentioned grids, i.e. the small scale (standard and radial IEEE 14) and the large scale (adapted Portuguese grid). In terms of validation of the output values retrieved from the API will be compared with the PSSE output of on one of the case files.

5.3.1.1 Client API Output

After running the API, as explained in Section 4.2, there are two possible outputs (Excel and XML). The used output for the analysis and validation of results is the excel due to its more interactive UI. The excel output of the standard IEEE 14 grid is represented in Figure 5.25. For the analysis ran to get this output it encompasses all the buses and uses all the 24 cases, although it can not be viewed in the Figure 5.25. This file is composed of two parts the header and the body of the file.

	A	B	C	D	E	F	G	H	I
1	Last Update	09/08/2019							
2									
3	Created By	Date/Time Created							
4	TSO	09/08/2019 00:22							
5									
6	Valid From	Valid To							
7	12-12-2018 00-(	12-12-2018 23-00							
8									
9									
10									
11	BUS NAME	Type	00:00	01:00	02:00	03:00	04:00	05:00	06:00
12		Icc3Ph Total without contribution from DSO [A]	2055	1883	1874	2034	1554	1029	1663
13		Contribution from EHV (MAT) [A]	2055	1882	1874	2034	1554	1028	1663
14		Contribution from HV (AT) connected [A]	0	0	0	0	0	0	0
15	BUS_1	Short Circuit Power [MVA]	783	717	714	775	592	392	633
16		Icc3Ph Total without contribution from DSO [A]	3053	2577	2562	3012	1867	996	2035
17		Contribution from EHV (MAT) [A]	3053	2577	2562	3012	1867	996	2035
18		Contribution from HV (AT) connected [A]	0	0	0	0	0	0	0
19	BUS_2	Short Circuit Power [MVA]	1163	981	976	1147	711	379	775
20		Icc3Ph Total without contribution from DSO [A]	2456	2047	2038	2435	1307	793	2115
21		Contribution from EHV (MAT) [A]	2456	2047	2038	2435	1307	793	2115
22		Contribution from HV (AT) connected [A]	0	0	0	0	0	0	0
23	BUS_3	Short Circuit Power [MVA]	935	780	776	927	498	302	805
24		Icc3Ph Total without contribution from DSO [A]	2547	2260	2240	2493	1716	1070	2017
25		Contribution from EHV (MAT) [A]	2389	2132	2118	2350	1620	972	1954
26		Contribution from HV (AT) connected [A]	180	176	175	176	171	168	145
27	BUS_4	Short Circuit Power [MVA]	970	861	853	949	653	407	768
28		Icc3Ph Total without contribution from DSO [A]	2684	2406	2382	2620	1876	1187	2131
29		Contribution from EHV (MAT) [A]	1811	1543	1539	1800	1036	345	1293
30		Contribution from HV (AT) connected [A]	903	911	894	870	868	841	858
31	BUS_5	Short Circuit Power [MVA]	1022	916	907	998	714	452	812
32		Icc3Ph Total without contribution from DSO [A]	9036	8220	8134	8801	6606	4384	7382
33		Contribution from EHV (MAT) [A]	5668	4904	4894	5643	3388	1156	4230
34		Contribution from HV (AT) connected [A]	3524	3546	3479	3394	3360	3229	3252
35	BUS_6	Short Circuit Power [MVA]	939	854	845	914	686	455	767

	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
11	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00	19:00	20:00
12	2044	1728	1751	1765	2042	2114	2653	2807	2806	2099	2504	1684	2101	1612
13	2044	1728	1751	1765	2042	2114	2653	2807	2806	2099	2504	1684	2101	1612
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	778	658	667	672	778	805	1010	1069	1069	799	954	641	800	614
16	3031	2120	1997	2017	2617	2165	3121	3567	3564	2143	2791	2044	3148	1810
17	3031	2120	1997	2017	2617	2165	3121	3567	3564	2143	2791	2044	3148	1810
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	1154	807	760	768	997	824	1189	1359	1358	816	1063	778	1199	689
20	2445	2047	1738	1750	2519	1482	2375	2773	2772	1471	1720	1898	2504	2089
21	2445	2047	1738	1750	2519	1482	2375	2773	2772	1471	1720	1898	2504	2089
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	931	780	662	666	959	564	904	1056	1056	560	655	723	954	796
24	2502	2242	2120	2145	2622	2176	2821	3055	3052	2147	2503	2065	2642	2021
25	2397	1968	1834	1825	2277	1858	2469	2699	2694	1840	2152	1893	2471	1855
26	176	296	310	339	360	339	370	370	367	314	352	179	185	179
27	953	854	807	817	999	829	1074	1164	1162	818	953	786	1006	770
28	2640	2358	2264	2284	2731	2400	3000	3216	3212	2364	2728	2203	2804	2147
29	1781	1358	1276	1337	1783	1390	2068	2285	2284	1371	1807	1308	1859	1151
30	878	1006	989	948	949	1047	946	943	939	1022	930	907	944	1001
31	1005	898	862	870	1040	914	1143	1225	1223	900	1039	839	1068	818
32	8890	8256	7982	7969	9277	8396	10033	10609	10593	8258	9259	7660	9398	7538
33	5574	4358	4128	4287	5578	4423	6365	6943	6943	4363	5646	4191	5754	3700
34	3424	3941	3878	3682	3699	4056	3688	3683	3667	3957	3624	3488	3652	3842
35	923	857	829	828	964	872	1042	1102	1100	858	962	796	976	783

Figure 5.25: Excel Output of the Client's API

The header as shown in Figure 5.26 possesses relevant information regarding the creator (Created by), date and time of creation (Date/Time Created), the date of the last update (Last Update) and the time frame of the analysis (Valid From, Valid To). The information regarding the creator is defined by the Python script, and the last update as well as the date and time of creation are defined by the machine running the API. The time frame is the only parameter in this header defined by the user, as shown in Chapter 4.2.

	A	B
1	Last Update	30/08/2019
2		
3	Created By	Date/Time Created
4	TSO	30/08/2019 22:14
5		
6	Valid From	Valid To
7	11/09/2019 00:00	11/09/2019 23:00
8		
9		

Figure 5.26: Header of the Excel Output

As for the body of the file the it contains all the information that is retrieved from the API regarding the short circuit values. As shown in Figure 5.27 The information is divided by bus, and each bus has four different values attached to it. These values are, the total 3 phase short circuit current and power (measured in Ampere and MVA) as well as the current contributions from HV and EHV (measured in Ampere). The Ampere measurement is chosen to remove the presence of decimal numbers that can prove to be problematic in their use with different software(some software uses , others .). The values for each bus are also divided by the case file that they were taken from, for example in the Figure 5.27 there are 5 columns that are from 5 different cases that represent 5 different hours (0 to 4 AM).

11	ID ISCP	Type	00:00	01:00	02:00	03:00	04:00
12		Icc3Ph Total without contribution from DSO [A]	2055	1883	1874	2034	1554
13		Contribution from EHV (MAT) [A]	2055	1882	1874	2034	1554
14		Contribution from HV (AT) connected [A]	0	0	0	0	0
15	BUS_1	Short Circuit Power [MVA]	783	717	714	775	592
16		Icc3Ph Total without contribution from DSO [A]	3053	2577	2562	3012	1867
17		Contribution from EHV (MAT) [A]	3053	2577	2562	3012	1867
18		Contribution from HV (AT) connected [A]	0	0	0	0	0
19	BUS_2	Short Circuit Power [MVA]	1163	981	976	1147	711
20		Icc3Ph Total without contribution from DSO [A]	2456	2047	2038	2435	1307
21		Contribution from EHV (MAT) [A]	2456	2047	2038	2435	1307
22		Contribution from HV (AT) connected [A]	0	0	0	0	0
23	BUS_3	Short Circuit Power [MVA]	935	780	776	927	498

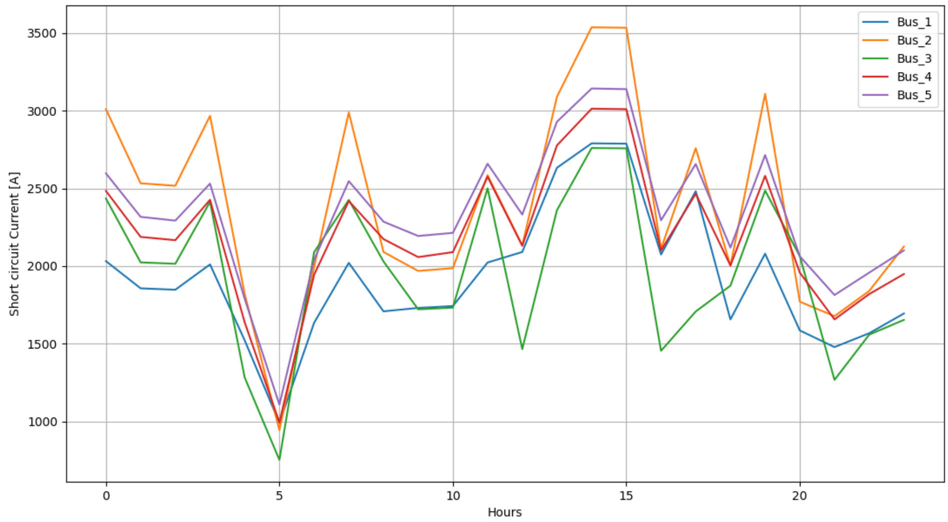
Figure 5.27: Body of the Excel Output

As for the structure of the XML output was displayed in Listing 4.1 the output is similar, the only difference being the amount of bus bars and cases displayed. The XML is the integral part of the communication process, its contents must be structured in such a way that obeys the language standards, these being its validation. A valid XML is one that is structured like the DTD (Document Type Definition) structure, when it does the XML can be correctly interacted with. The resulting XML file needs to pass through verification software in order to ensure that it possess a valid and acceptable structure, to be correctly processed later. The software used to verify the code was a XML editor named Liquid Studio 2018 - Community Edition, in where the XML file was inserted and analysed revealing that its structure is according to the DTD standard.

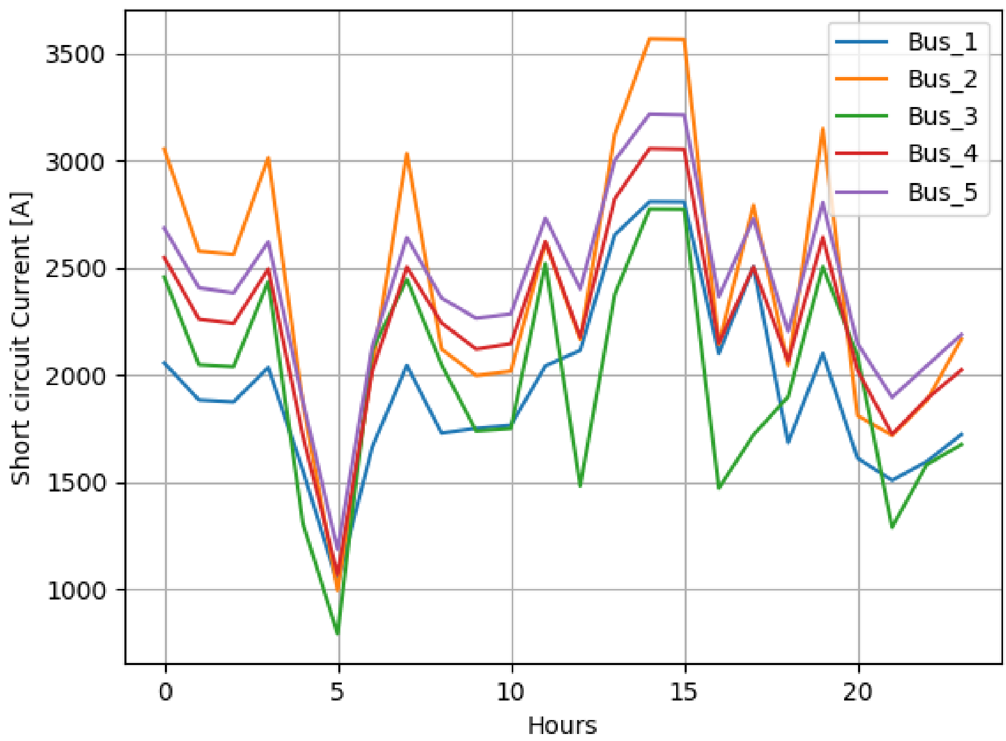
Now that all the values are calculated the three phase short circuit profile of the created grid can be displayed for the both the radial and the standard grids. In order to allow the analysis for each type of grid two different graphics were made, one for each bus voltage (220kV and 60kV). For the 220kV bus bars the Figure 5.28a shows the profile from radial topology represented and the standard by Figure 5.28b.

Regarding the different scope of values the 60kV profiles had to have a different graph separate from the 220kV one. This is shown in Figure 5.29 where the standard and radial are represented by Figure 5.29b and 5.29a respectively.

With these results there are some conclusions that we can extrapolate, the influence of EHV and HV, the difference between radial and meshed grids, the profile of the I_{cc} as well as the influence of RES and other factors in I_{cc}. Starting with the difference in base voltage it is seen by the Figure 5.28 that all of the buses in the EHV grid follow a similar profile and have approximately the same results. Having the maximum I_{cc} on Bus 2 at hour 15 with values of over 3,5kA in both cases. As for the HV grid, as evidenced by Figure 5.29, that there are differences in the values of I_{cc} throughout all the buses in this grid. However, despite this difference they still have similar profiles, having peaks and

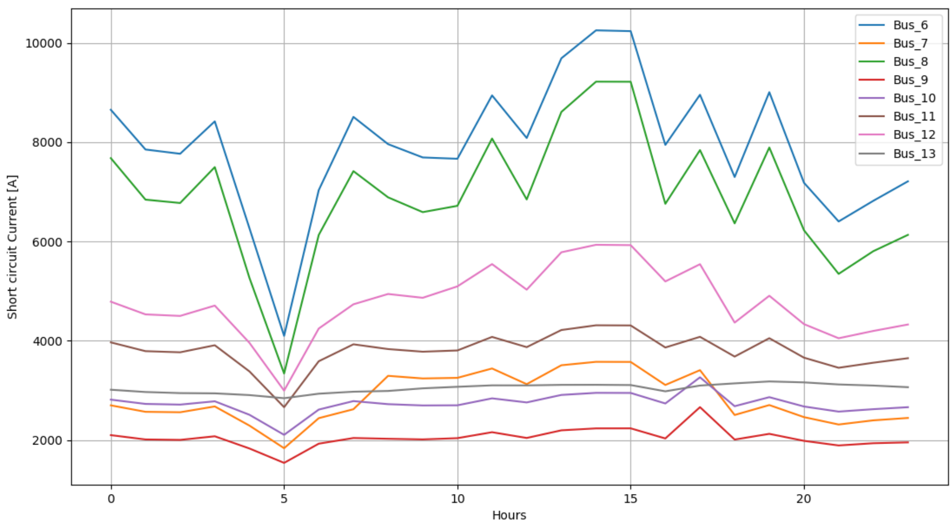


(a) Radial EHV three phase fault Output Chart

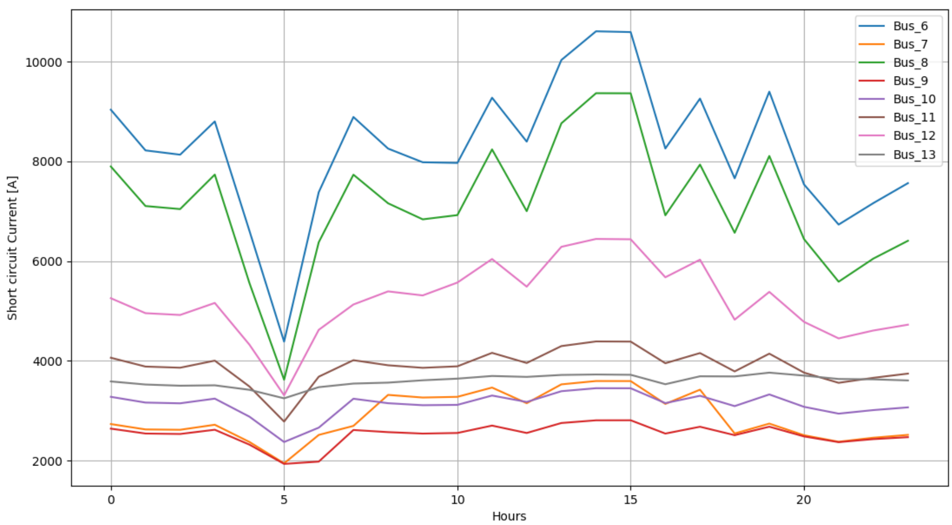


(b) Meshed EHV three phase fault Output Chart

Figure 5.28: EHV 3ph fault Output Chart



(a) Radial HV three phase fault Output Chart



(b) Meshed HV three phase fault Output Chart

Figure 5.29: HV 3ph fault Output Chart

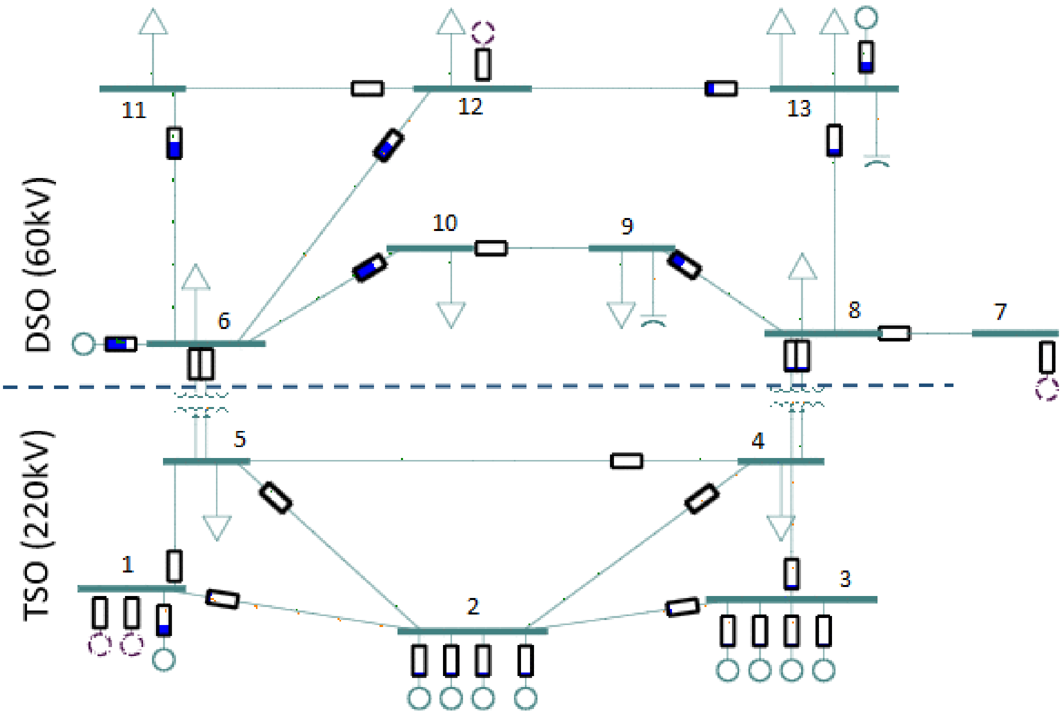


Figure 5.30: IEEE 14 Grid.

slopes in the same places. As for the I_{cc} values in this voltage level is substantially higher than the EHV counterpart, having a maximum of over 10kA. This difference between HV and EHV value can be attributed to the difference in the base current, as detailed in Chapter 2.1.3. As the Figures illustrate the range in currents in the HV grid is larger than in the EHV.

For the difference in topology of the grids there were some detected in both Figure 5.28 and 5.29. In both Radial topologies there is an offset that reduces the value of I_{cc} in the whole graph. For example in hour five of Figure 5.29 the value of the Meshed topology is 2kA and in the Radial topology the value at this hour is lower, this continues throughout the rest of the profile and also on the EHV buses. This difference in this case is subtle since the difference between topologies and the complexity of the overall grid are not that great. This is due to a constant difference in the equivalent impedance of grid to calculate the I_{cc}, as explained in 2.1.3. As seen in both Figures the maximum values of I_{cc} does not reach the limit imposed by the Portuguese TSO and DSO (25kA for the TSO and 31kA for the DSO, values elaborated in Chapter 3.2.2). But by seeing the difference in these two topologies if a more complex grid were to be analysed the values of I_{cc} would increase.

The overall profile of both the HV and the EHV are somewhat similar having slopes and peaks in the same locations. This can be attributed to the functioning of the generation throughout the whole 24 cases. This means that with more machines connected in each bus the more the I_{cc} rises. Since the energy generation is intrinsically entwined and dependant with the consumption we can assume that the profile can change when the consumption changes. If we look to Figure 5.31 and compare with the I_{cc} a similar profile can be seen, as they both share similar peaks and slopes (5h for slopes and 10h and 18h for the peaks). To corroborate this assumption of the machine interference in the I_{cc} their active periods are displayed in Annex II.

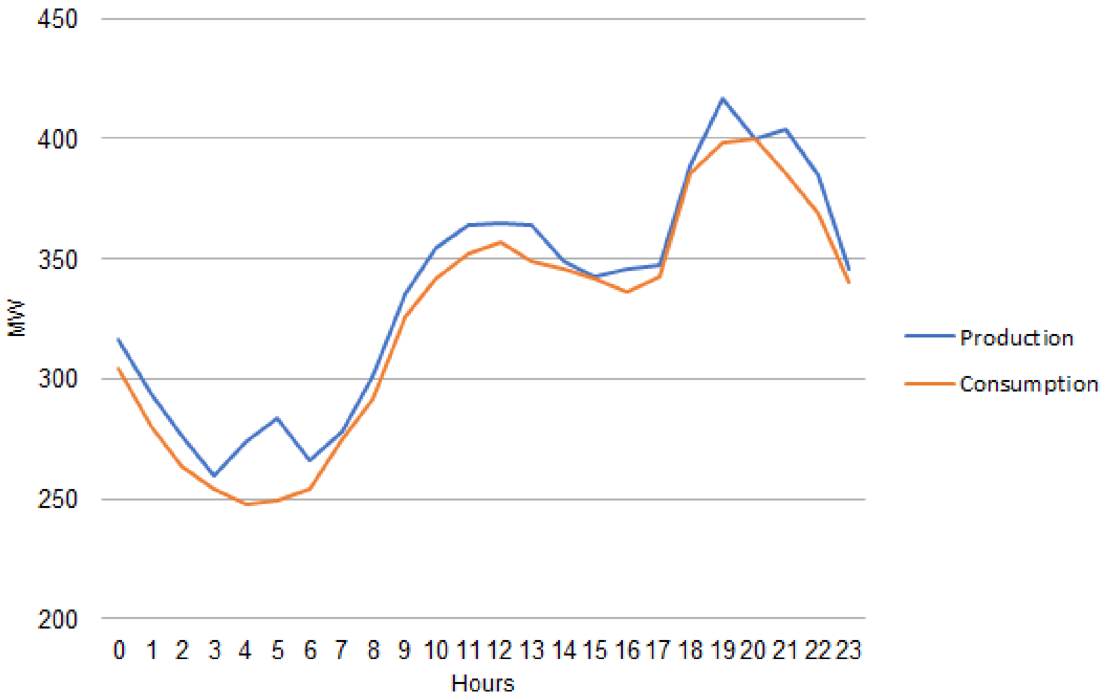


Figure 5.31: Production and Consumption profile

As established in Chapter 2.1.3, the I_{cc} rises when either the base voltage of the grid is lower or the equivalent impedance of the grid from a given bus is also lower. The voltage difference was already touched upon previously, but the influence of the grid topology can also be seen in all the graphics. When analysed the higher values of I_{cc} are located in buses that have more branches or machines connected to it (bus 2 in EHV and buses 6 and 8 in HV). This is given by complexity of the grid in a certain point, but can also be applied to buses that have RES connected to it. For example the bus 7 although being only connected by one line it is not the bus with the lowest I_{cc}. That belongs to bus 9 and at certain hours of the day bus 10, that each of them has two branches connected to them. This phenomena is derived of the RES connected to bus 7 and when this energy source is injecting energy to the grid (during the day time) the I_{cc} rises.

After analysing the IEEE 14 grid it has time to test the API on a large scale grid to know if it can withstand a substantially larger grid as defined in Chapter 5.1.2. In order to make the interface easier for the user it was just slightly changed from the version explained in Chapter 4.2. This easier interface with the user incorporates two options for the bus subsystem definition, these were either all buses or the excel bus subsystem input. This was put into place since the individual bus option would make the GUI extremely overburden with buttons and some of the buses were not being displayed. As for the TSO, DSO and TSO/DSO border since this proved to be not that practical or useful was scrapped. For these reasons the API just has the two options for the Bus subsystem definition. In terms of the outcome the 13 grids were ran in full to test the validity and time latency of the script. In terms of latency the API does take more time to process the larger grid than the IEEE 14 example.

5.3.1.2 Comparing with PSSE Outputs

For the verification of the actual short circuit values the chosen way to compare the performance of the API, since there is no other API similar to this one developed at the time, is to compare directly with the PSSE output. This is to ensure that the values that are being used for the API are valid and correct.

As for the calculation of the actual short circuit currents by the PSSE software the method applied was based on the standard fault current calculation. In the Figure 5.32 shows the option chosen for the analysis on the PSSE software.

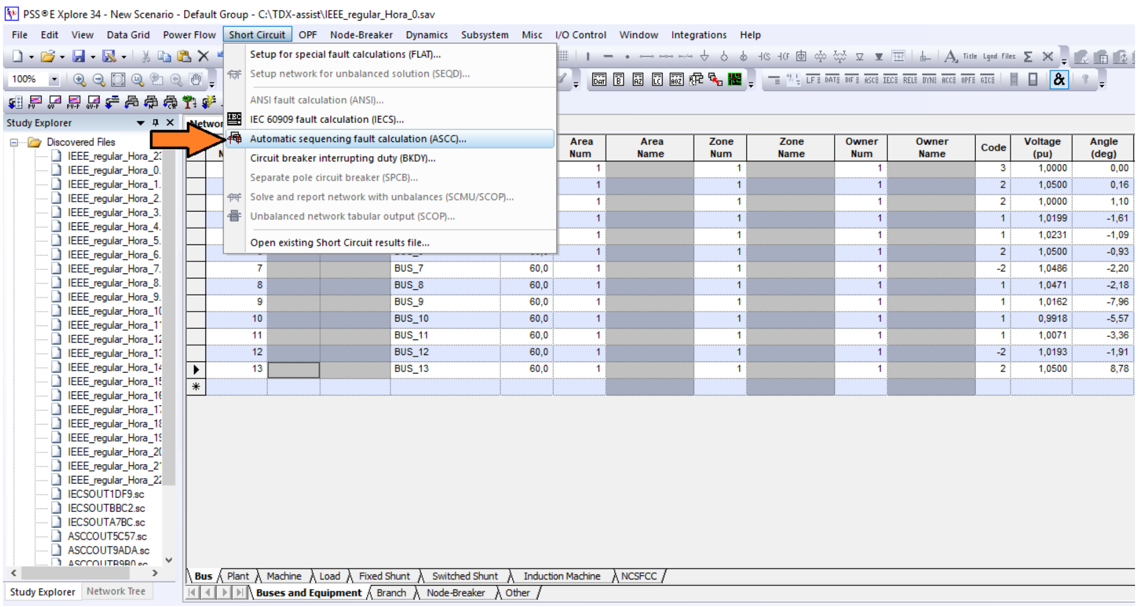


Figure 5.32: Short Circuit Analysis

The output of this PSSE function is partly shown in Figure 5.34, this only shows the output of two of the chosen bus bars in the Meshed grid. In this case the chosen buses were Bus 4 and 5 in the Meshed grid (highlighted in Figure 5.33), which in the IEEE 14 are border buses. This means that they possess contributions from both the HV and EHV grids. These buses are ideal for the test seeing that they are influenced from both grids, which allows to test if the API processes the information correctly. The output of the PSSE shows the analysed Bus (displayed in yellow), the total value of the short circuit current (highlighted in red) as well as the separated values of the contributions (green for EHV and blue for HV). In this output all the phases of the current are displayed and each one of them is characterised by two values of polar coordinates. In order to get the total value of the contributions a vectorial sum of all the currents needs to be done, as for the total value of the current that does not need to be done since it is directly given. For comparison with the PSSE values the API's output of the same two buses is shown in Figure 5.35.

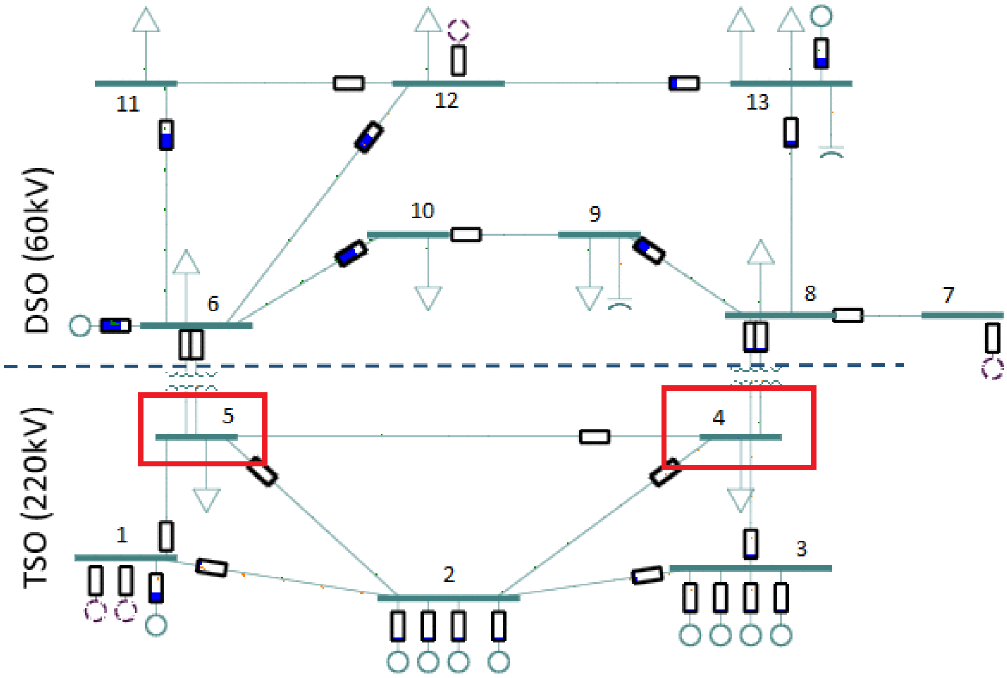


Figure 5.33: Buses Analysed (Bus 4 and 5)

When comparing the values of the total current with the one retrieved from the API, they are the same. When summing up the currents of each voltage level the values also match to the ones given by the API. These values prove the validity of the values retrieved from the API.

The cases shown here were referent to the IEEE 14 grid, however, the values were also verified for the large scale grid. this was to ensure the process can manage any type of grid for analysis. Even with the larger grid the API still returns the correct values for both the short circuit contributions.

5.3.2 Initial Server Test

To test the performance of the server it was submitted to a defined regiment in order to assess its performance. In order to evaluate the server's performance, the latency of the request/response reaction and CPU usage was monitored. To execute the test a local network was set up to ensure a safe and uninterrupted experiment. For this network it was used the the hotspot wireless functionality of Windows 10. This ensures that only

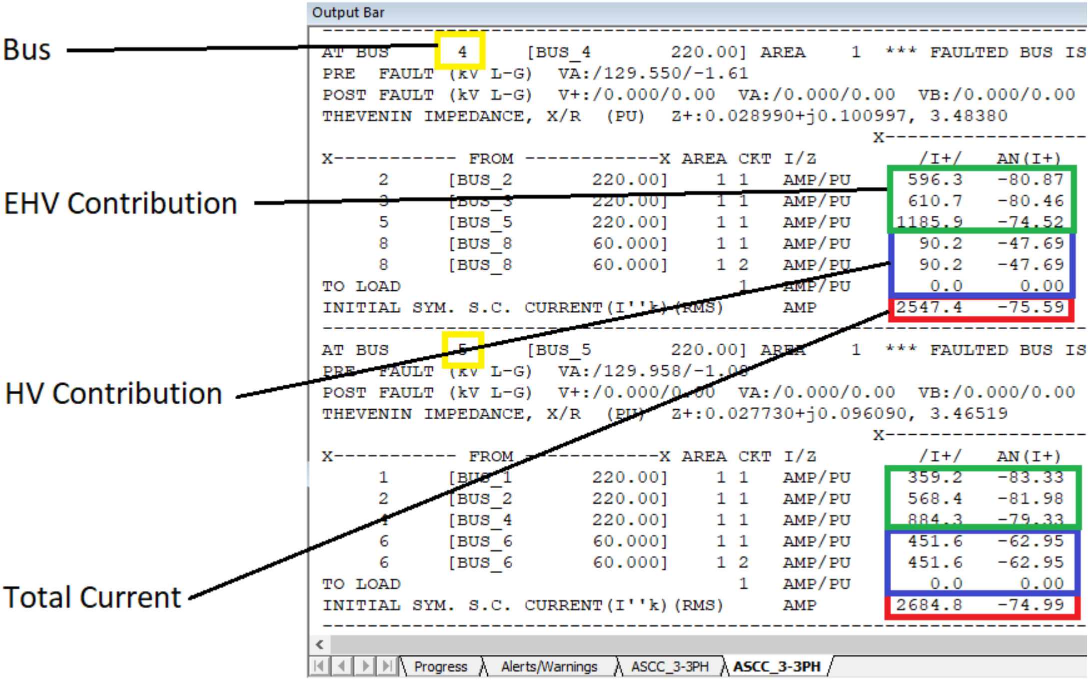


Figure 5.34: PSSE Output

ID ISCP	Type	00:00
	Icc3Ph Total without contribution from DSO [A]	2547
	Contribution from EHV (MAT) [A]	2389
	Contribution from HV (AT) connected [A]	180
BUS_4	Short Circuit Power [MVA]	970
	Icc3Ph Total without contribution from DSO [A]	2684
	Contribution from EHV (MAT) [A]	1811
	Contribution from HV (AT) connected [A]	903
BUS_5	Short Circuit Power [MVA]	1022

Figure 5.35: API output of Buses 4 and 5

the machines connected to this network can use the web-service. Although the maximum amount of machines that can access this network is limited, allowing only 8, this can prove to be more than enough to test the server. The number of machines used in the test were 5 with a stable connection with the network. The test consisted of 10 minutes and in each 2 minutes a new task established, in order to check the server performance and reaction. The 5 tasks consisted in the following steps:

1. User Creation - The machine operators were told to create several users in two minutes.
2. Put Message - After logging in with a user the operators were told to upload to the server several times a file that was previously provided to them. The files provided were retrieved from the Client's API, each one of them is different from all the others.
3. List Message - After uploading files to the database the users were told to search freely through said database during the two minutes.
4. Get Message - For this task the users were told to use the Get message function to retrieve a file from the database.
5. Free Usage - These two minutes were dedicated for the users to use any and all the features tested before freely.

The tasks were specifically made to test the web-service and the database access. The ability to write to the database with tasks 1 and 2, querying the database with task 3 and retrieving information from database in task 4. Task 5 is to test the simultaneous different types of accesses to the database.

During the test no issues occurred, all the interactions with the server were successfully made. The database entries are correctly accessed and saved in the proper locations. In terms of latency of the request/response the amount of time spent is almost none. All the resources and functionalities are available to the user in fraction of seconds. This still happens in the peak of usage of the service and no evidence of any slow down by the server was detected. Finally the CPU performance during the the whole test never exceeded 5% of the Raspberry Pi's total processing power. This was constantly monitored during the entirety of the test reaching a peak in the final task of the test. The result of this test is a positive outcome for the Proof of Concept.

Chapter 6

Conclusions

In this Chapter an overview of the implementation results explored on this Thesis are summarised, as well as the future work planning.

6.1 Results Overview

There is a definite push to comply with the European standards regarding the production and consumption of energy from renewable sources (Wind, Solar, Hydro). This may result in instability in the grids that deliver energy across the country. To minimise such instability, channels of communication between system operators need to to be implemented as well as the information definition conveyed through those channels. This communication needs to meet regulations and specifications established by the European Union regarding data exchange. to achieve it, a RESTful architecture to develop a Web-service to exchange the data between System Operators and a Client API that arranged the data from a certain number of grids and returned usable information to share with the neighbouring grids was implemented.

The implementation required to be developed to the highest standards not only to comply with EU regulations but also to ensure rigorous testing made it possible to port it and use it with real data cases.

6.1.1 Client API Validation

The tests made to the API were successful in demonstrating the functionalities and validate the usefulness of the platform. The values that were retrieved from the grid were properly managed and allocated into both the XML and the Excel files. The form and content of both file outputs was verified to make sure they could be used in the rest of the implementation's structure. Content validation, that encompasses all the values retrieved form the case files, was achieved by comparing them with the PSSE output values regarding the same used values. With the form of the files only the XML fits into the category since the excel does not require to adhere to any to any universal file structure just the one defined for this use by this thesis. The excel file includes a filter that highlights the instances where a certain bus exceeds a certain amount of I_{cc} at a given time.

With all the values verified and the outputs being created properly the API was subjected to a test using cases with substantially higher complexity, which means more elements in the grid itself. For this test some examples based on REN's yearly report regarding the Portuguese transmission grid were used to test the scalability of the script to larger case files. The test was successful and responded with the accurate values and it is currently being used for other case files that represent the actual real data case files.

After testing and validating the output values from the Client API, there was an analysis of these values in regard of the influence of the grid's characteristics. This analysis was made in order to rationalise the different values of I_{cc} and S_{cc}, by identifying their causes. The causes identified that alter these values were base voltage (EHV vs HV), complexity of the grid (Meshed vs Radial) and the associated generation of the grid (Distributed vs Concentrated). These grid's characteristics have influence in the profile of each bus I_{cc} and have a similar profile to the load diagram of the whole grid. With the values of I_{cc} defined there is also a direct influence in the dimensioning of the bus protections.

6.1.2 RESTful Server Validation

The RESTful web-service was developed in compliance with European standards either by the used data exchange protocol or the inert characteristics of the web-service itself regarding privacy and the conditioning of access. These standards were kept by the structure of the RESTful Architecture, also by implementing a login feature to the Server and restricting the access to web-service to authorised users in a local network. The chosen hardware was specified in Chapter 4.3.3, with preliminary data on a small computer indicating thus if the Server can be implemented in this machine it can also be scaled-up to an actual server hardware.

The Server after being developed was tested to assess the performance and eventual bugs that might be in the structure. A stress test was run, detailed in full in Chapter 5.3.2, that focused on the performance of both the web-service response and the Server CPU. At the end of the test the web-service responded to each request instantly and the strain and usage done to the Server CPU is minimal, with a maximum of 5% of the total CPU usage during the whole test.

6.2 Future Work

Although findings are encouraging, future work is required to scale-up the platform. To achieve it, some key features still need to be further developed such as:

- Processing of the information in the XML file and add it to a PSSE grid.

  The solution developed dealt with the processing of the information of a given grid retrieving the usable information converting into two types of output files and the interaction with the RESTful Server. However, there is still the information treatment of the XML file after being retrieved from the server. The information in the XML file needs to be retrieved, incorporated in a grid file and then it is processed to send a new XML file to the Server.

- Implementation of the Web-service in a larger scale Server.

  This implementation used a Raspberry Pi as the Server hardware, nonetheless, the Server to function with the real data from the DSO and TSO requires a different hardware and allocation of the Web-service and database to this new equipment.

- Redesign the graphical interface of the Web-service pages.

  A web-service interface with a more user-friendly outlook for the regular usage of all the parties involved.

- Implement in the Web service the functions of Overview, Settings and Admin and Account Verification.

  There were planned functionalities for the web-service that would act as add-ons to the Server itself. These focused on the processing of information of the XML and the user registration. These were not the main topic of the thesis therefore they were not developed.

  First, the overview function would serve as a statistic and graphical display of the information in a XML file. To access this functionality the user would press the Overview button in the Dashboard as shown in Figure 6.1. After this the user would have to choose one file, such as the Get Message, this would give the user the option to choose the a bus from the file. Finally after the choice is made a graphic of the I_{cc} and the TSO and DSO I_{cc} limitations is made. This would make it easier to view the profile of a certain bus's I_{cc}.

  Other functionalities are more user-focused in terms of allowing access to the platform and having an Admin account. The settings page would have the specific information regarding the user (username, e-mail,change password and searches) and this would be the personal page of that actual user. As for the Admin account this would be able to give access to new users and remove old ones this functionality. This is to reduce the amount of users that are authorised to work in the platform. This functionality works by giving to the Admin account the privileges to verify other accounts giving or removing access to certain functionalities. All these functionalities would be accessed via a button highlighted in the top right corner of the Figure 6.2.

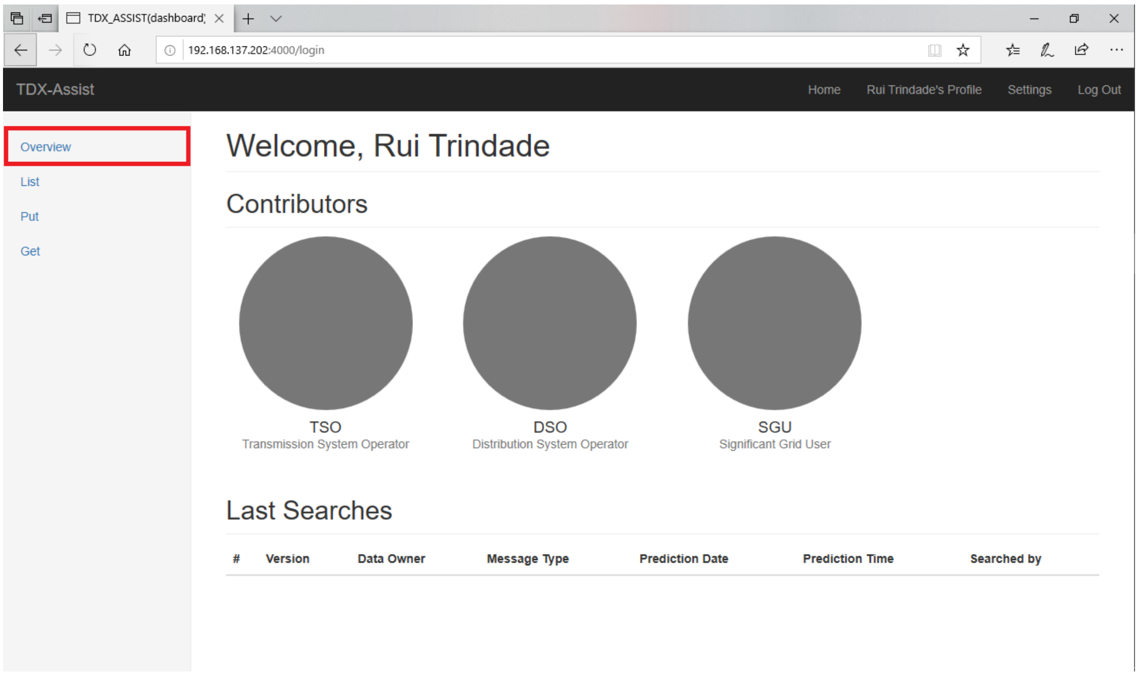


Figure 6.1: Dashboard Overview Selection

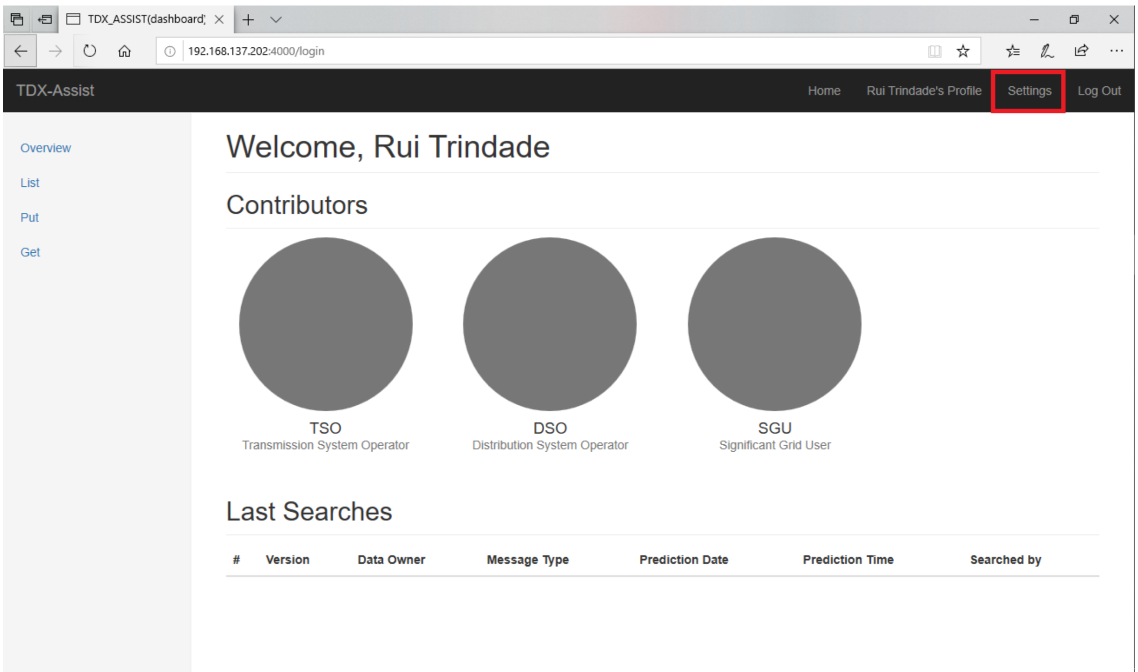


Figure 6.2: Dashboard Settings Selection

6.3 Submitted Article

During the development of this thesis an article, based on this topic, was written and submitted to an IEEE conference. The conference in question was the IECON 2019 held in Lisbon, Portugal in October 2019. This conference covers contemporary industry topics ranging from electronics, controls, manufacturing to communications and computational intelligence. Given the adequacy of topics the article was submitted for consideration. It was titled "Implementation of RESTful web-services as a platform for exchanging information between grid operators",a derivation from the title of this same thesis. The full article is shown in Annex I.

The submission was made to test the relevance of the methodology used in this work under the scrutiny of peer-review process. The article was accepted for publishing. According to the reviewers, the methodology was the strong point of the article.

Bibliography

[4] *IEC 60050 - International Electrotechnical Vocabulary - Welcome*. URL: http://www.electropedia.org/ (visited on 01/23/2019).

[1] *Universidade de Évora e REN criam "ilhas de biodiversidade" sob postes de alta tensão - Rádio Campanário*. URL: https://www.radiocampanario.com/ultimas/regional/universidade-de-evora-e-ren-criam-ilhas-de-biodiversidade-sob-postes-de-alta-tensao (visited on 2018).

[2] J. Paiva. *Redes de Energia Elétrica*. 4th ed. IST Press, 2015.

[3] M. Neves. *Redes de Energia Eléctrica / Produção e Transporte de Energia*. Ed. by U. Departamento de Engenharia Electrotécnica Faculdade de Ciências e Tecnologia.

[5] C. J.W. H. Z. Yun L. Junjie. "A Framework Research of Power Distribution Equipment Condition Monitoring Cloud Platform Based on RESTful Web Service." In: *2017 IEEE Conference on Energy Internet and Energy System Integration (EI2)*. 2017.

[6] P. R. N. Singhal U. Sakthivel. "RESTful Web Services Composition & Performance Evaluation with Different Databases." In: *2017 International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT)*. 2017.

[7] P. P.N. B. K. Murthy A. Kumar. "Implication of Restful web services in Bigdata Analytics." In: *2nd IEEE International Conference on Computational Systems and Information Technology for Sustainable Solutions 2017*. 2017.

[8] J. T.A.R.-C. S. Segura J. Parejo. "Metamorphic Testing of RESTful Web APIs." In: *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL.44, NO.11, November 2018* (2018).

[9] P. H.C.M.-E. S. A. Archip C. Amarandei. "RESTful Web Services – A Question of Standards." In: *2018 22nd International Conference on System Theory, Control and Computing (ICSTCC)*. 2018.

[10] J. M.Y. R. G. Serme A. Oliveira. "Enabling Message Security for RESTful Services." In: *2012 IEEE 19th International Conference on Web Services*. 2012.

[11] E. E. G. CEDEC EDSO. *TSO – DSO DATA MANAGEMENT REPORT*. 2018.

[12] E. F.N.O.E.-R.E.E.E.E. I. G. Taylor BUL. *State of the Art - TSO-DSO Interoperability, Coordination of Transmission and Distribution data eXchanges for renewables integration in the European marketplace through Advanced, Scalable and Secure ICT Systems and Tools*. 2017.

[13] *WEO*. URL: https://www.iea.org/weo/ (visited on 02/05/2019).

[14] F. R. R. Alves. *State of the Art and Gap Analysis on TSO-DSO Data Exchange in Portugal*. Tech. rep. Centro de Investigação em Energia REN-STATE GRID, R&D Nester, 2017.

[15] I. E. C. (IEC). *Framework for energy market communications - Part 504: Utilization of web services for electronic data interchanges on the European energy marquet for electricity*. Tech. rep. IEC, 2015.

[16] REN. *Dados Técnicos '17*. Tech. rep. REN, 2017.

[17] E. F.N.-O.E.R.E.E.E.E. I. G. Taylor BUL. *TSO-DSO Use Cases Agreed models, Use Case list, and Use Case description in UML, Coordination of Transmission and Distribution data eXchanges for renewables integration in the European marketplace through Advanced, Scalable and Secure ICT Systems and Tools*. 2018.

[18] *The MagPi Magazine*. URL: https://www.raspberrypi.org/magpi/raspberry-pi-3bplus-specs-benchmarks (visited on 08/01/2019).

[19] *Sysbench*. URL: https://wiki.gentoo.org/wiki/Sysbench (visited on 08/31/2019).

[20] *Raspbian*. URL: https://www.raspbian.org (visited on 08/01/2019).

[21] *Centrodeinformacao.ren.pt*. URL: http://www.centrodeinformacao.ren.pt/PT/InformacaoExploracao/Paginas/EstatisticaDiariaDiagrama.aspx (visited on 05/19/2019).

[22] *Mercado.ren.pt. (2019)*. URL: http://www.mercado.ren.pt/PT/Electr/InfoMercado/Consumo/Paginas/Verif.aspx (visited on 05/19/2019).

[23] M. Molina and P. Enrique Mercado. "Modelling and Control Design of Pitch-Controlled Variable Speed Wind Turbines." In: Apr. 2011. ISBN: 978-953-307-221-0. DOI: 10.5772/15880.

Annex I

Annex 1 - Developed Paper

Implementation of RESTful web-services as a platform for exchanging information between grid operators

Rui Trindade
Departamento de Engenharia Electrotecnica
Faculdade de Ciências e Tecnologias, UNL
Almada, Portugal
rm.trindade@campus.fct.unl.pt

Francisco Reis
R&D Nester
Centro de Investigação em Energia REN-State Grid S.A.
Lisbon, Portugal
fa.reis@fct.unl.pt

***Abstract*—The main focus centres around a proposed implementation of a Web-Service based information exchange platform. The information sharing will be rooted on the relation between System Operators, in order to increase and expand the scope of their operational coordination levels in various time frames. This shall be accomplished in two parts, the first is the Web-service specifications and the communication between System Operators. The second addresses one specific potential application that corresponds to Ssc and Isc (Short Circuit Power, Short Circuit Current) during a 24 hour period. For the Web-service, the developed platform is based on RESTful Architecture. For the grid analysis the PSS/E tool in together with Python language are the used in this research. This framework was tested using IEEE 14 based bus network**

***Index Terms*—RESTful, Python, Short-Circuit, TSO, DSO, 504 Protocol**

I. Introduction

A. The need to coordinate and exchange information

Renewable energy sources contribution in distribution networks has reached a ultimate high in recent years. Despite the clear environmental benefits, the use of such energy sources has created new challenges in Power Systems operation, mainly due to their uncertain behaviours. As a result, better coordination between System Operators is of utmost importance and as such information exchange platforms need to be designed and specified. In order to increase the level of coordination information exchange needs to be designed and specified. In that regard EC, ACER and CEER released statements and protocols to increase the interactions between TSOs and DSOs (Transmission System Operators and Distribution System Operators) [1].

There is pressure in European Union (EU) to meet some goals in the form of the 20/20/20 energy objective directive. That dictates the reduction by 40% of the emissions greenhouse gasses in comparison with 1990, 27% of overall renewable energy consumption and a reduce in 27% of energy consumption by 2020 [1]. Showing that renewable energy usage expansion is a reality in the next couple of years.

By improving the communication between the System Operators enables the prediction of the grid operating point. This can be both a scheduled information or in Real-Time information, allowing the anticipation and contingency planning of all influenced grids [2].

As the energy production moves to more sustainable systems, like renewable energy, such as the planning and function of the energy grid changes. This transition has a high impact on the System Operators since there are more and more energy producers injecting power to the grid [1].Although the growth rate of energy production through renewable sources in the European Union might be modest when comparing to the rest of the world, there is a clear move from traditional energy supplies and their impact on System Operators can not be disregarded. [3]. Countries such as Denmark, Portugal and Spain are successfully integrating these kinds of energy sources [1].

In the context of TSO and DSO, Transmission and Distribution System Operators, coordination some BUC, Business Use Case, were identified. These deals with scheduled and Real-Time information in order to predict and safeguard the grid [2]. To illustrate the need for increased TSO and DSO coordination in operational planning time frame (1 day to 1 week) a specific BUC will be presented. This BUC relates to the three phase short-circuit calculation at the interface bus between the System Operators.

II. Use Case Short-Circuit

A. Current Practices

In Portugal the current method of transmission information is particularly stable and reliable [1]. The Data exchange between EDPD and REN (Energias de Portugal, Redes Energticas Nacionais) is based on an ICCP(Inter-Control Center Communications Protocol), this connection is only used for Real-Time information. The TSO and DSO possess a SCADA system (Supervisory Control and Data Acquisition) connected to the ICCP,that enables the Real-Time information monitoring and analysis of the data exchanged. The ICCP also facilitates SGUs (Significant Grid Users) connection to the system and sharing their readings thus allowing better predictions [1]. The Figure 1 illustrates the overall operation of the system. The

Figure I.1: Paper Page 1

ICCP links the TSO and DSO, ensuring an accurate, reliable and fast exchange of real-time data between them [1].

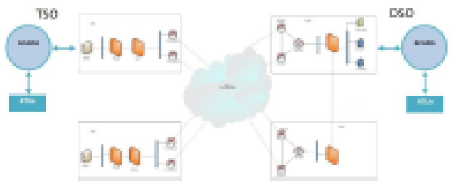


Fig. 1. Architecture between TSO and DSO for real time information exchange [1].

There are plans for the grid development in the Portuguese Electrical grid infrastructures plan (PDIRT) [4]. These plans, are bound by legislative obligations and rules of the electrical sector. In them there is a particular section addressing to the the expected short circuit currents. This is achieved by defining a maximum and a minimum short circuit current range allowed as depicted in Figure 2 [4].

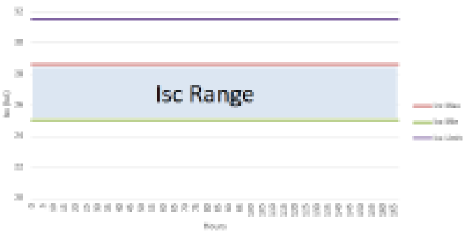


Fig. 2. PDIRT Isc values and range for the grid [4].

The objective is to capture the dynamic behaviour of the Isc. If one calculates on an hourly basis, taking into account the generation scheduled and topology, the short circuit at nodal level. It obtains a pattern similar to the on illustrated in Figure 3. Clearly the pattern is connected with the corresponding load diagram [4].

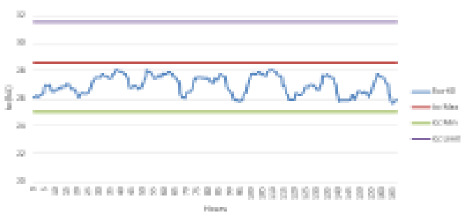


Fig. 3. Hourly dynamic compared with the PDIRT short circuit currents [4].

III. Suggested Solution

A. *RESTful Web-Services*

REST or Representational State Transfer is an architecture that enables the communication client-server through the internet or a local network. Defined by Dr.Roy Fielding, REST is a architecture with a style described as a "distributed hypermedia application" [5]. In REST the state of the application represents the client's state, or a session state. And the resources used by the web-service are categorised as hypermedia in the used browser [5]. REST is neither a standard nor a technology but a framework that has defined constraints. If a REST service follows all the ones defined bellow the service can be called a RESTful web service [6].

- **Client-server interaction**
  There is a split between the Client or user interface and the server. The client cannot be developed separately from the server, the server only acts when called upon by a request-response [5] [6] [7].
- **Stateless**
  The server does not store the state of the client, the client possesses the state of the application. Allowing to be up-scaled and the possibility of concurrent computing with ease, since it does not require to manage the state information [5] [6] [7].
- **Cache usage**
  The contents referred to a request made to the server are present in the communication chain. In a way preventing the client repeating the same request, reducing the system inertia and its load rate, increasing the overall productivity [5] [6] [7].
- **URI Structure**
  The URI or Uniform Resource Identifiers is the standardised structure that RESTful uses to defined the requests sent. It allows a easier interaction with the server improving its visibility [5] [6] [7].
- **Layered system**
  This architecture can be segregated into independent layers, each one with a specific purpose. Therefore a single layer only interacts with those in its vicinity. This layered system, acts in a predictable way and also reduces the risk of coupling functionality [5] [6] [7].
- **On-demand code**
  To allow a better interaction with the system, the client can actively download and execute code. This can be favourable because it allows to add new application codes giving a great expansive and evolutionary capabilities [5] [6] [7].

The RESTful web service data exchange uses a request and response approach, using HTTP requests. These have their equivalent as RESTful requests, POST/CREATE, GET/READ, PUT/UPDATE, DELETE, or more commonly known as CRUD [5]. This request is sent to the server using the URI which when it reaches there identifies the requested entity/resource. This could be an Image, Text, XML, JSON

Figure I.2: Paper Page 2

file depending on the request or scope sent by the client [8]. These interactions are represented in the Figure 4 below.

- CREATE/POST - Request that handles the adding of a resource or collection into the server. Derived by the successful reception of the server, it returns a affirmative or negative response [7] [8].
- READ/GET - Request that retrieves a given resource or collection from the server. Returns the targeted resource if successful or a negative response if unsuccessful [7] [8].
- UPDATE/PUT - Request that alters already existing information in the server system. If successful returns an affirmative response, if there is no file in the Database returns a negative response [7] [8].
- DELETE - Request that removes resource or collection from the server. If successful returns an affirmative response, if there is no file in the Database returns a negative response. In the case of a collection it also deletes all the resources appended to it [7] [8].

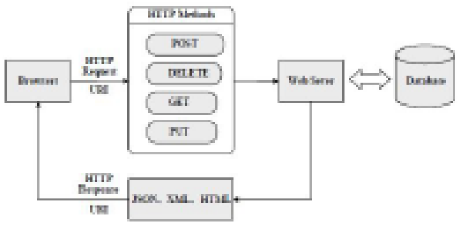


Fig. 4. RESTful response-request [5].

1) 504 Protocol: This protocol (IEC 62325-504) is part of the IEC framework (International Electrotechnical Commission) for energy market communications, more specifically the use of web-services for data exchange [8]. This framework is supposed to function in near real-time, in a secure way and it can be applied as the solution to other types of integration problems outside it's scope [9]. The 504 has some base requirements that need to be met in order for it's effective implementation [9]. As most of them encompass the minimal technologies and methods to be used for the base 504 to work, but they can have upgraded versions [9].

- List messages - Definition that queries the database for a entry, that the user defines, and displays the information on screen [9]. This can be achieved trough a GET request.
- Get messages - Finds a certain file entry in the database and imports the XML file to the client's computer [9]. This can be achieved trough a GET request.
- Put messages - Saves to the database a specific XML file provided by the client along side the information specific to the file [9]. This can be achieved trough a POST request.

As seen in this protocol the only type of requests needed are the GET and POST, simplifying the server functionality. All of them have certain dynamics in term of information that enters and leaves for each request. In the table I there is all the information exchanged in each action.

TABLE I
504 Service Requirements

	List		Get		Put	
	Req	Res	Req	Res	Req	Res
Message ID	O	M	C	O	O	-
Message Version	O	M	C	O	O	-
Internal Code	C	M	C	O	O	-
Status	C	O	O	O	O	-
App Time frame	C	M	O	O	O	-
Server Time	C	M	O	O	O	-
Message Type	O	M	O	O	O	-
Data Owner	O	M	O	O	O	-
File	-	-	-	M	M	-

*M-Mandatory, O-Optional, C-Choose

B. Python API

In order to access the PSS/E in a more programmable way the Python API (Application programming interface) was used to allow a more autonomous interaction with several case files. Python was chosen as the programming language due to the existence of a PSSE library, exclusive to Python language. This facts would prove effective in order to analyse all the pretended aspects of the case files. The developed script is divided into three different groups:

1) **User interface**
This is where the operator decides the extent of the analysis. The user may choose the area of the grid in which the sc calculation will be performed. Either changing the grid topology or defining the bus system that will be run. In terms of the definition of a bus subsystem this brings the possibility of choosing to analyse all busses or just the ones defined by the user.
2) **Grid Analysis**
Where the main grid analysis takes place. It cycles through 24 different cases and queries for relevant information.This constitutes the main body of the Python code. The results are Thevenin impedance, base voltage and symmetric three phase fault current of each Bus.
3) **Information Treatment**
Organises the information obtained and arranges it into readable formats. This takes place after the Grid Analysis and grabs the relevant information presented in strings and structures converting them to an Excel and a XML File both with different functions. Relevant information refers to the fault power and currents . As for the different files, the excel is for a more visual representation of the values allowing for the creation of graphics. And the XML file is the chosen file format to be transferred through the server.

Figure I.3: Paper Page 3

IV. System Test

To test the developed RESTful platform a client server framework was used together with an IEEE 14 based bus network. Both the grid and server were based on existing technologies and standards and expands upon them.

A. Grid Topology

For the grid characteristics the IEEE 14 standard was used as a basis, as represented in Figure 5. All the values applied to the scenario are based on real cases.

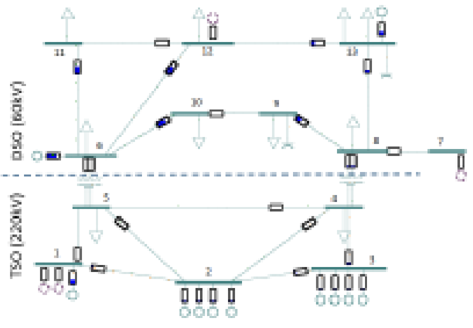


Fig. 5. IEEE 14 Grid.

In this network both transmission and distribution are represented. Load and generation profiles were developed based on real life profiles. These are represented in Figures 6 and 7 as the solar and wind generation profiles and the Industrial and Residential load, this information was obtained through REN's public information.

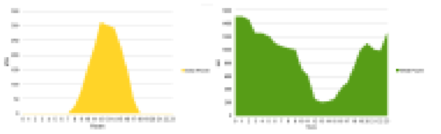


Fig. 6. Real Characteristics of Solar and Wind Energy Generation. [10]

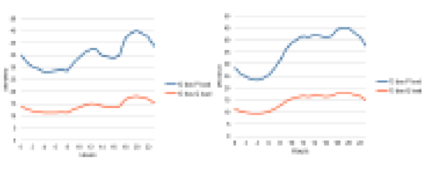


Fig. 7. Real Characteristics of Residential and Industrial Loads. [11]

Bus loads and generation are added in a way to mimic an actual grid. The types of plants added to grid were Thermal, Hydro, Wind and Solar, as follows.

- Bus 1 - Thermal Power Plant
- Bus 2 and 3 - Hydro Power Plant
- Bus 4 and 5 - 220kV substation busses, load of a EHV costumers
- Bus 6 - 60kV substation bus with Wind powered generation, load of a HV costumer
- Bus 7 - Solar power plant
- Bus 8 - 60kV substation, load of a HV costumer
- Bus 9 - Load of an Industry sector
- Bus 10 and 11 - Load of a Residential sector
- Bus 12 - Prosumer (Solar) of a residential sector
- Bus 13 - Wind powered generation and loads of an Industry sector

B. Server Topology

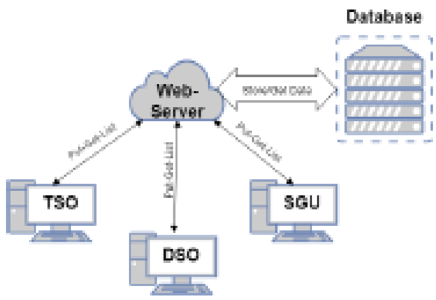


Fig. 8. Proposed Model.

In the server there were a couple of requirements to meet for it to be within the specified parameters. These being the 504 protocol specifications and the European standards for information transfer between Service Operators. So the choice of the language for the server must be versatile enough to meet the requirements. As the Figure 8 above exemplifies, the clients, either TSO, DSO or SGU can access the web server in the local network. This web server in turn, depending on the request made by the client, interacts with the physical database.

For that the language chosen for the implementation of the server was Python. As Flask is a Python library that enables the necessary key-points for the server setup and it is very easy to develop on. As for hardware used the choice was the micro computer Raspberry Pi 3 B+. This decision was made in order to prove that the platform works is light-weight and can be scalable for increased interactions.

The Flask REST and RESTful library are very easy to implement this makes the server development pretty straightforward. However, the user interface can be quite tricky for newcomers for that purpose a dashboard was created to allow a easier access and navigation. This Dashboard acts as the main Hub for the user it shows the possible operations available to the user, Search, Download file and Upload file.

Figure I.4: Paper Page 4

As described by the 504 protocol the interactions with the server must have specific requests and responses. These are implemented into this service as the operations described before have their parallel with the 504. Search is the List Message, Download file is Get message and Upload is the Put message. The development of these operations had from the beginning the 504 protocol in mind therefore are in line with the requirements.

With all this ease of access there has to be a control in who is allowed to operate the Dashboard. For that a user must be created and authenticated in order to access the Dashboard functions. This is made possible by the implementation of the Flask-Login functionalities. Besides the login requirement the server can only be accessed through the localhost therefore must be done locally in the same network. All the information about the user and the files in the server are stored in the server own local database.

C. Results

With the testing we could get the fault current in each bus in each hour of the day. These are shown in the graphics below as Figure 9 refers to the 220kV busses and Figure 10 the 60kV ones. And as it can be seen there a difference in current values between the busses with 220kV and 60kV. This coupled with the high contrast between minimum and maximum values proves that there are a lot of swing factors in the grid. These factors can be caused by the difference in load throughout the day resulting in adapting the energy generation to the consumption. Or the different operating cycles of each producer, for example solar energy having a high impact during the day or wind at night.

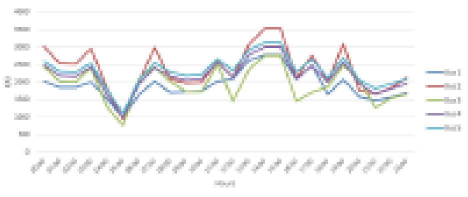


Fig. 9. Fault Currents of 220kv busses

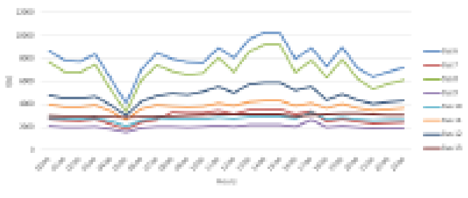


Fig. 10. Fault Currents of 60kv busses

Another impacting result is that the highest contrast in the fault currents is in the border busses between HV and EHV. This shows the relevance of sharing this information between TSO and DSO. Because if each side consider the other as a passive grid the retroactive effects can be dire. In the case of considering the neighbouring grids as active and underestimate the short-circuit current there can be damaged equipment and shortages. In the other hand if it is overestimated there would be a great deal of unnecessary resources spent.

V. Conclusions

A web-based platform was developed to comply with the requirements for data exchange between TSO and DSO. A specific application was used in what concerns the exchange of information, related to symmetric three phase faults. This was achieved for a time frame of 24 hours ahead (operational planning).

As for the RESTful web-service, the tests were successful as a small computer was able to withstand multiple requests at once. This is promising because if the service's hardware can be upgraded to more powerful specifications allowing it to be able to run even more requests. As for the speed of processing since the files that are transmitted are relatively small the requests are instantaneous without latency. In the case of expanding the scope and frequency of use of the server since the REST architecture is well structured, there will be no trouble.

References

[1] G. Taylor, State of the Art - TSO-DSO Interoperability, Coordination of Transmission and Distribution data eXchanges for renewables integration in the European marketplace through Advanced, Scalable and Secure ICT Systems and Tools, BUL, EDF, FRAUNHOFER, NESTER, OFF, ENTSOE, REN, EDPD, EIMV, ELES, EG, INESC, 2017

[2] G. Taylor, TSO-DSO Use Cases Agreed models, Use Case list, and Use Case description in UML, Coordination of Transmission and Distribution data eXchanges for renewables integration in the European marketplace through Advanced, Scalable and Secure ICT Systems and Tools, BUL, EDF, FRAUNHOFER, NESTER, OFF, ENTSOE, REN, EDPD, EIMV, ELES, EG, INESC, 2018

[3] J. Paiva, Redes de Energia Eltrica, IST Press, 2015, 4th edition

[4] PDIRT Plano de Desenvolvimento e Investimento da Rede Nacional de Transporte, REN, junho 2017

[5] Z. Yun, L. Junjie, C. Ji, W. Hua, A Framework Research of Power Distribution Equipment Condition Monitoring Cloud Platform Based on RESTful Web Service, 2017 IEEE Conference on Energy Internet and Energy System Integration (EI2), 2017

[6] N. Singhal, U. Sakthivel, P. Raj, RESTful Web Services Composition & Performance, Evaluation with Different Databases, 2017 International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT), 2017

[7] A. Archip, C. Amarandei, P. Herghelegiu, C. Mironeanu, E. Serban, RESTful Web Services A Question of Standards, 2018 22nd International Conference on System Theory, Control and Computing (ICSTCC), 2018

[8] K. Murthy, A. Kumar, P. Palagi, N. Bhasaha, Implication of Restful web services in Bigdata Analytics, 2nd IEEE International Conference on Computational Systems and Information Technology for Sustainable Solutions 2017, 2017

[9] International Electrotechnical Commission, IEC TS 62325-504 Technical Specification, IEC, 2015

[10] Centrodeinformacao.ren.pt, (2019) PREs. [online] Available at: http://www.centrodeinformacao.ren.pt/PT/InformacaoExploracao /Paginas/ED_PREs.aspx?Data=07-01-2017 [Accessed 19 Apr. 2019].

[11] Mercado.ren.pt. (2019). SIMEE - Verificado. [online] Available at: http://www.mercado.ren.pt/PT/Electr/InfoMercado/Consumo /Paginas/Verif.aspx [Accessed 19 Apr. 2019].

Figure I.5: Paper Page 5

Annex II

Annex 2 - IEEE 14 Grid Characteristics

In this Annex the characteristics of the IEEE 14 based grid used in Chapter 5. This Annex will highlight the the topology and all the values of the structures involved in all the 24 cases developed. These involve values that are common to all cases (lines and transformers) and those who change with each case (generation and load). In Figure II.1 the IEEE is displayed.

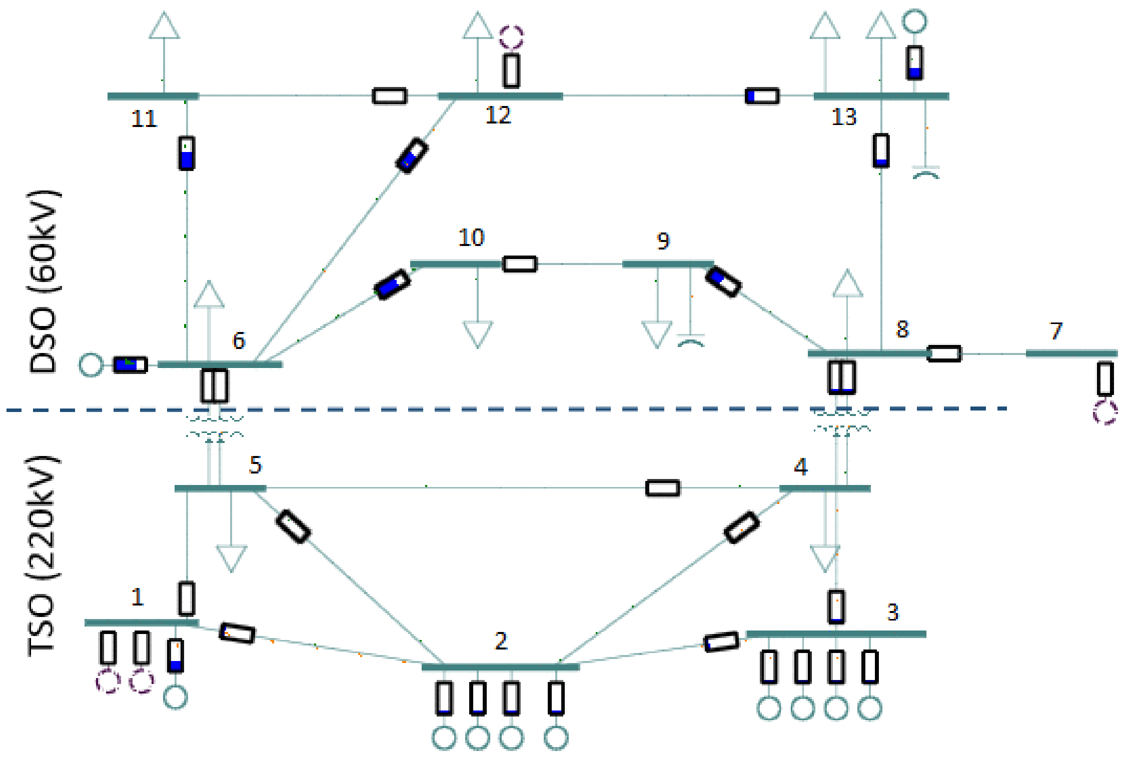


Figure II.1: IEEE 14 Grid.

II.1 Line and Transformer Characteristics

In terms of lines the values that were used as the basis are dependent on the base voltage of the lines (60kV and 220kV) and the fact that they are simple lines. The values that were adapted from the lines standards were the resistance (R), the reactance (X), the susceptance (B), and the power rate. The first three the standard values came in (pu/Km) and the power rate in (MVA). The next step was to define the length of the lines in order to calculate their actual values. Again using the real life as an example, the lines of the TSO are substantially greater in length than the ones from the DSO. The attributed length values as well as the resulting line characteristic are displayed in Table II.1. As referenced in section 2.1 the 60kV lines possess a greater impedance rate (pu/Km) than the 220kV ones, this is a fact to be aware in terms of power flow.

As for the transformer the values taken into consideration for the transformers were the short circuit reactance (X) and their respective power rate. The reactance in the transformers is different than in the lines, where the impedance reveals to be a percentage value of the transformer's power. As expressed in Table II.1 the percentage of the short circuit impedance is 10%, And the amount of power that can travel through it in a given amount of time is 150 MVA.

Table II.1: Line and Transformer Characteristics.

Line	R(pu)[a]	X(pu)[a]	B(pu)[a]	length(Km)	Rate(MVA)
Line 1-2	0.0192	0.1056	0.1548	120	316
Line 1-5	0.04	0.22	0.3225	250	316
Line 2-3	0.036	0.198	0.29025	225	316
Line 2-4	0.032	0.176	0.258	200	316
Line 2-5	0.03168	0.17424	0.25542	198	316
Line 3-4	0.03104	0.17072	0.25026	194	316
Line 4-5	0.00768	0.04224	0.06192	48	316
Line 6-10	0.08322	0.23553	0.01288	52	60
Line 6-11	0.10704	0.30293	0.01656	67	60
Line 6-12	0.05450	0.15427	0.00843	34	60
Line 7-8	0.03719	0.10526	0.00575	23	60
Line 8-9	0.03535	0.10006	0.00547	22	60
Line 8-13	0.11313	0.32019	0.01750	71	60
Line 9-10	0.08037	0.22745	0.01243	51	60
Line 11-12	0.08363	0.2367	0.01294	53	60
Line 12-13	0.11658	0.32970	0.01802	73	60
Transformer	X(%)			Rate(MVA)	
Transformer 4-8	10			150	
Transformer 5-6	10			150	

[a] $|S_{base}| = 100MVA$, $|U_{base}| = 220kV$, $|U_{base}| = 60kV$

II.2 Generation Profiles

As highlighted in Chapter 5.1.1 the generation considered in the grid are thermal, hydro, wind and solar, their location is shown in Figure II.2. The source for the profiles of each energy generation power plant was taken from [21], in different days that were also referred in Chapter 5.1.1. From those profiles 24 values were removed to have one for each case. However, some adaption was needed in order to fit to the used grid. For that only the profile of production was kept and the maximum value of power production was changed to a more suitable one. The maximum values of production of each plant is highlighted in Table II.2.

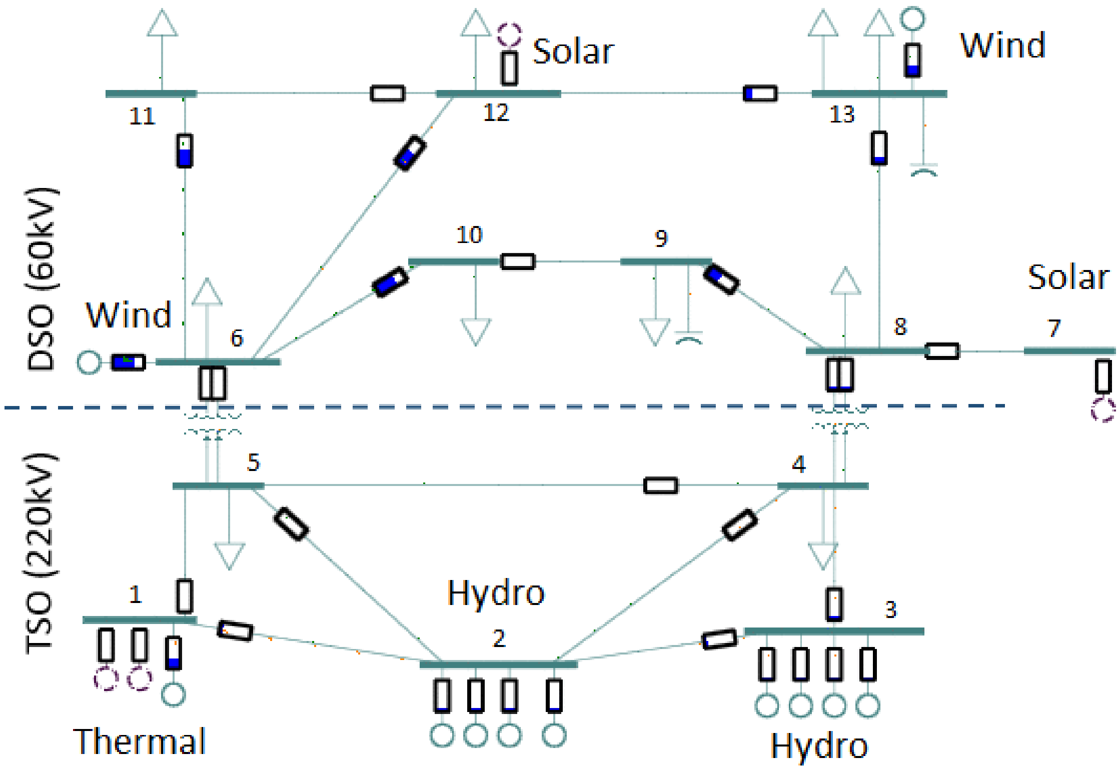


Figure II.2: IEEE 14 Grid, with discriminated generations.

The injection of power for each plant is different and can be separated into two groups, the continuous injection and the controlled injection. The first refers to the solar and wind power plants that when these are producing energy they inject it fully in the the grid. On the other hand the controlled injection is used on thermal and hydro where the produced energy is dependant with the demand increasing or decreasing depending on the grid's consumption. For this control to work productions plants have more than one machine, so these plants can only reach the values described in the adapted profile when all the machines in each plant are operational and working. The thermal production plant has 3 machines that when all are working produces a power of 240 MW, this power is constant due to this production being man controlled. The hydro power plants have 4 machines which in the peak production when all 4 are working produces 200 MW. The values displayed in Table II.2 for the thermal and hydro plants refers to the amount of machines operational, therefore, to get the values of the hydro production profile all values need to be adapted to 4 machines. The Table II.2 refers to all the production values and their respective bus of all the power plants used for the creation of the case files.

Table II.2: Generation Characteristics.

Gen type	Thermal		Hydro				Wind		Solar	
Bus num	1		2		3		6	13	7	12
Hour	MW	num	MW	num	MW	num	MW	MW	MW	MW
0	0	0	33.6	3	29.4	4	160	93	0	0
1	0	0	11.9	4	5.9	4	160	116.3	0	0
2	0	0	2.2	3	3.1	3	154.7	116.3	0	0
3	0	0	0.9	3	1.5	2	133.3	124	0	0
4	0	0	0.6	2	0.3	1	133.3	139.5	0	0
5	0	0	0	0	0.3	1	128	155	0	0
6	0	0	3	1	5.9	4	117.3	139.5	0	0
7	0	0	14.9	4	11.8	4	112	139.5	0	0
8	0	0	4.5	1	26.5	3	108.8	155	4	2.4
9	0	0	4.5	1	50	2	106.7	155	12.1	7.3
10	80	1	4.5	1	32.4	2	74.7	124	24.2	14.5
11	80	1	11.9	2	64.7	4	64	85.3	36.3	21.8
12	160	2	7.5	1	13.2	1	26.7	77.5	50	30
13	160	2	13.4	3	22.1	3	21.3	69.8	48.4	29
14	160	2	17.9	4	11.8	4	22.4	62	46.8	28.1
15	160	2	17.9	4	11.8	4	26.6	62	37.1	22.3
16	160	2	14.9	1	11.8	1	42.7	77.5	24.2	14.5
17	80	1	76.1	3	39.7	1	53.3	85.3	8.1	4.8
18	0	0	50	1	150	3	80	108.5	0	0
19	0	0	40.3	1	152.9	4	106.7	116.3	0	0
20	0	0	0	0	158.8	4	117.3	124	0	0
21	80	1	32.8	1	36.8	1	106.7	147.3	0	0
22	80	1	25.4	1	35.3	2	104.5	139.5	0	0
23	0	0	29.8	2	35.3	2	133.3	147.3	0	0
Max Power(MW)	240		200		200		160	155	50	30

II.3 Load Profiles

Similarly to the generation profiles, the load profiles were based from real case data [22]. Depending on their profiles each load is characterised as either industrial or residential load, in Figure II.3 the loads can be seen associated with each bus (except buses 1, 2, 3 and 7). After choosing the profiles for each bus there needs to happen a adaptation to the grid, equal to what happened with the generation profiles. For that a maximum was chosen and the profile of the grid is maintained relative to said maximum. The maximums established are dependant on the voltage level of the given load, the loads on HV buses have a maximum of 45MW and the loads on EHV buses 20MW. In Table II.4 are the load active power for each HV bus and in Table II.3 for the EHV ones, in both tables all the values for the 24 cases are shown. As for the reactive power associated to each bus, a power factor of $\tan\phi = 0,4$. was used to get the for all cases.

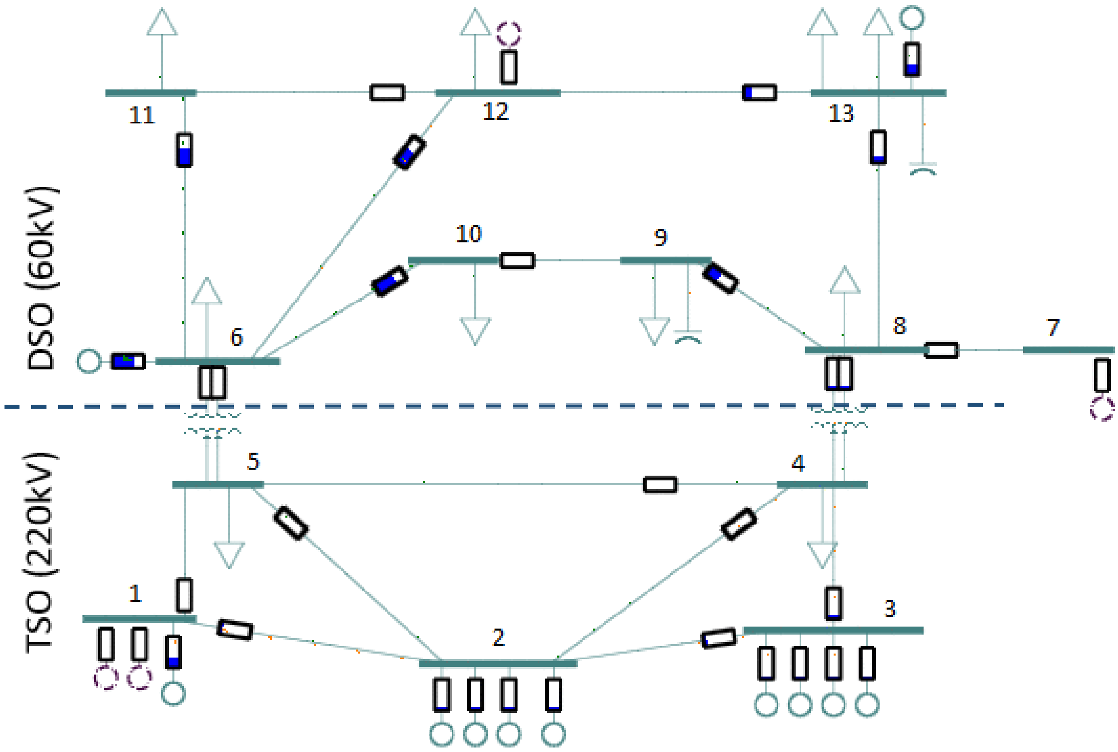


Figure II.3: IEEE 14 Grid.

Table II.3: EHV Load Characteristics.

Bus num	Bus 4	Bus 5
Hour	MW	MW
0	15.3	13.5
1	14.1	12.5
2	13.3	11.6
3	13.1	11.4
4	12.8	11.4
5	12.9	11.5
6	13.5	12.1
7	15.1	14.0
8	16.9	15.9
9	18.5	17.8
10	18.7	18.2
11	19.1	18.6
12	18.7	18.2
13	17.9	17.5
14	18.1	17.9
15	18.1	17.6
16	17.5	17.2
17	17.6	17.3
18	19.5	19.2
19	20	19.9
20	19.9	20
21	19.2	19.3
22	18.4	18.3
23	17.1	16.7

Table II.4: HV Load Characteristics.

Bus num	Bus 6	Bus 8	Bus 9	Bus 10	Bus 11	Bus 12	Bus 13	
Hour	MW	MW	MW	MW	MW	MW	MW	MW
0	33.6	33.6	33.2	37.1	37.3	35.2	28.4	36.7
1	30.9	30.9	30.4	36.0	34.3	32.2	26.0	33.2
2	29.4	28.7	28.6	34.0	31.8	30.1	24.6	31.2
3	28.1	27.8	28.0	31.5	30.9	29.2	23.6	30.3
4	27.6	27.5	27.2	30.0	30.1	28.1	23.3	29.4
5	28.2	27.8	27.8	29.1	29.9	28.3	24.1	29.3
6	29.0	28.6	28.9	28.8	30.0	28.5	25.5	29.6
7	32.8	32.5	32.4	28.7	31.5	28.9	28.1	30.9
8	36.9	37.0	36.1	26.9	31.2	28.4	31.4	31.1
9	40.7	41.0	40.2	29.1	35.4	31.6	36.2	35.0
10	41.7	41.4	41.6	32.2	37.8	34.0	38.7	37.1
11	42.2	42.1	42.3	35.6	38.3	35.9	40.3	37.7
12	42.2	41.8	42.4	37.7	38.6	37.3	41.5	38.2
13	40.4	40.2	41.3	37.1	38.3	37.2	41.0	37.8
14	41.5	41.0	41.9	34.8	36.8	34.8	42.1	36.3
15	41.3	40.4	41.6	34.4	36.4	34.2	41.6	36.0
16	40.2	39.3	40.8	34.9	35.8	33.8	40.9	35.4
17	40.4	39.8	41.0	36.9	36.8	34.7	41.5	36.3
18	44.0	43.8	44.1	42.7	43.2	41.8	44.3	42.5
19	45.0	45.0	45.0	44.2	45.0	44.1	44.9	44.9
20	44.9	44.9	44.9	45.0	45.0	45.0	45.0	45.0
21	43.0	43.1	43.1	44.4	43.0	43.9	43.1	43.0
22	41.1	41.4	41.4	43.2	41.0	42.3	41.1	40.5
23	37.9	37.8	37.8	40.0	38.6	38.6	37.6	38.0