Universidade de Aveiro Departamento Electrónica, Telecomunicações e Informática
2018

Diogo Emanuel Morgado Saraiva

IEC 61850 Routable GOOSE and SV profiles over IP Telecom Networks

Utilização do protocolo IEC 61850 sobre redes de telecomunicações IP

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrónica e Telecomunicações, realizada sob a orientação científica do Professor Rui Luís Andrade Aguiar (orientador), Professor Catedrático do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro e do Doutor Cipriano Lomba (co-orientador), Coordenador de Tecnologia e Inovação da EFACEC.

O júri / The jury

Presidente / President

Prof. Doutor Mário José Neves de Lima
Professor Auxiliar do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

Vogais / Examiners comittee

Doutor Rui Lopes Campos
Investigador do Instituto de Engenharia de Sistemas e Computadores do Porto

Prof. Doutor Rui Luis Andrade Aguiar
Professor Catedrático do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

Agradecimentos / Aknowledgments

A entrega desta dissertação marca um fim de um ciclo proveitoso na minha vida, tal como é o percurso académico. Com ele aprendi e cresci, como futuro engenheiro e mais importante, como homem. Estou orgulhoso da escolha de ter envergado nesta área e mais orgulhoso ainda do trabalho que desenvolvi.

Primeiramente gostaria de endereçar especial apreço ao Prof. Doutor Rui Aguiar e ao Prof. Doutor Daniel Corujo pela ajuda e ensinamentos dados durante este último ano.

Em segundo, a toda a minha família e amigos pelo apoio e, em especial, paciência demonstrados.

A todos os meus professores e colegas de curso com os quais me deparei durante o percurso académico.

Às pessoas da EFACEC que se dispuseram sempre a ajudar quando necessário.

E por último, não menos importante até, àqueles que pessoalmente ou remotamente contribuíram para a realização desta dissertação.

Palavras-chave

IEC 61850, IED, GOOSE, SV, WAN, Gateway, Tunnelling, 4G, LTE, Optical Fibre, ADSL

Resumo

Vivemos numa era em que o consumo de energia cresce a um nível quase imensurável, colocando em causa o equilíbrio de um sistema aquando a ocorrência de uma falha na rede elétrica. Isto trouxe a necessidade de a tornar numa rede mais segura e inteligente, surgindo então o conceito de Smart Grid.

O standard IEC 61850 foi inicialmente concebido para operar dentro das subestações, contudo vislumbraram-se outras potencialidades para o aplicar não só dentro, mas também para fora das mesmas. Assente em vários protocolos de comunicação, entre os quais MMS, GOOSE e SV, o IEC 61850 oferece a possibilidade de troca de dados em tempo real, levando a uma redução de custos, um aumento da segurança bem como a interoperabilidade entre os equipamentos dos diferentes vendedores.

Abordando este tema em conjunto com a EFACEC e o Instituto de Telecomunicações de Aveiro, houve uma questão se levantou: Quais são as condições requeridas para a comunicação entre os equipamentos das diferentes subestações?

Esta dissertação de mestrado visa dar resposta a esta questão, assim como analisar outras possíveis tecnologias a utilizar no momento da sua implementação. Realizada essa análise, houve uma tecnologia que se evidenciou, o 4G-LTE. Foi elaborado um estudo com recurso a diagramas temporais de mensagens, utilizando as duas abordagens descritas no standard. Quer utilizando a abordagem por IP quer utilizando a abordagem por VPN, foram testados em ambos os cenários diferentes tecnologias como a ADSL, Fibra Ótica, bem como o já referido, 4G-LTE. De seguida, são analisados os resultados do estudo, obtendo as devidas conclusões para cada uma das abordagens bem como para cada uma das diferentes tecnologias.

Por fim, é analisada qual das abordagens será mais proveitosa a longo prazo, assim como qual o trabalho futuro que deverá ser desenvolvido no que diz respeito às diferentes áreas do standard apresentado.

Keywords

IEC 61850, IED, GOOSE, SV, WAN, Gateway, Tunnelling, 4G, LTE, Optical Fibre, ADSL

Abstract

We live in an era where the consumption of energy rises to an almost immeasurable level, jeopardizing the balance of systems when a power failure in the electrical grid occurs. This brought the necessity of making it safer and intelligent, hence emerging the concept of Smart Grid.

The IEC 61850 standard was initially designed to operate within substations, yet other potential developments glimpsed to operate not only within substations but also to the outside of them. Based on several communications protocols, including MMS, GOOSE and SV, the IEC 61850 provides the possibility of real-time data exchange, leading to cost savings, increased security and interoperability between different vendors' devices.

Addressing this issue, together with EFACEC and the Institute of Telecommunications of Aveiro, a key question emerged: What are the requirements for the communication between devices in different substations?

This MSc dissertation aims to answer this question, as well as to analyse other possible technologies that are suitable at the time of its implementation. With this analysis, a specific technology stands out, the 4G-LTE. A study was conducted recurring to temporal diagrams of the messages, performing with both approaches as described in the standard. Either using the IP or VPN approach, different technologies such as ADSL, Optical Fibre, and the previously referred 4G-LTE were tested. Thereafter, the results of the study are analysed, obtaining this way the conclusions for each one of the approaches, as well for each of the different technologies studied.

Last but not least, it is analysed which of the approaches will be more beneficial in the long-term, as well as the necessary future work that must be developed in the different areas of the presented standard.

Table of Contents

List of Figures

List of Tables

List of Graphics

Abbreviations

ACSI	Abstract Communication Service Interface
AR	Auto-Recloser
ARPANET	Advanced Research Project Agency Network
CB	Circuit Breaker
CDC	Common Data Class
CDMA 2000	Code Division Multiple Access 2000
DL	Downlink
DSLAM	Digital Subscriber Line Access Multiplexer
EDGE	Enhanced Data Rates for GSM Evolution
FDM	Frequency Division Multiplex
GOOSE	Generic Object Oriented Substation Events
GPRS	General Packet Radio Service
GR	Gateway Receiver
GS	Gateway Sender
GSE	Generic Substation Events
GUI	Graphical User Interface
HMI	Human Machine Interface
HSPA	High Speed Packet Access
IEC	International Electrotechnical Commission
IED	Intelligent Electronic Devices
IEEE	Institute of Electrical and Electronic Engineers
IP	Internet Protocol
ISP	Internet Service Provider
LAN	Local Area Network
LD	Logical Device
LN	Logical Node
LTE	Long Term Evolution
MMS	Manufacturing Message Specification
OFDMA	Orthogonal Frequency Division Multiple Access
OHL	Over Head Lines
ONT	Optical Network Terminal
OS	Operating System
OWTT	One Way Trip Time
PON	Passive Optical Network
PV	Parameter Value
RTT	Round Trip Time
SAS	Substation Automation Systems
SCADA	Supervisory Control and Data Acquisition
SC-FDMA	Single Carrier Frequency Division Multiple Access
SCL	Substation Configuration description Language
SCSM	Specific Communication Service Mapping
SI	Session Identifier

SPDU	Session Protocol Data Unit
sqNum	Sequence Number
stNum	State Number
SV	Sampled-Values
TC	Technical Committee
TCP	Transmission Control Protocol
TDM	Time Division Multiplex
TD-SCDMA	Time Division Synchronous CDMA
TR	Technical Report
TSDU	Transport Service Data Unit
UDP	User Datagram Protocol
UL	Uplink
UMTS	Universal Mobile Telecommunications System
VLAN	Virtual Local Area Network
WAN	Wide Area Network
WiMAX	Worldwide Interoperability for Microwave Access

1. Introduction

In the first chapter is presented a quick introduction about smart grids, followed by the problem statement and the objectives desirable to accomplish. Additionally, it is described how the document is structured in each chapter.

1.1 Smarts Grids

A large part of the technologies that we use daily improved a lot in past few years but there is one thing that didn't receive proper attention until recently, the electrical grid. Needless to say that the world electricity demand is increasing every day [1] and it is a concern to promote the best way to save energy as well as not to waste it in processes that can be automated.

Now, there is a need of turning the electrical grid into a smart grid and smart grids came to fight this and many other problems that we could face, such as misuse of electricity, stealing of energy, protect the grid, etc. When all these problems are solved, there will be an efficient, sustainable, economic and secure way to deliver energy to people.

In a smart grid, among the important aspects of a smart distribution system, are the integration of smart control and protection devices. Belonging to smart grids, Over Head Lines (OHL) are lines that are used to transmit power for long distances to the load area and that require a complete and robust protection system. One of the devices that equip these systems are the Auto-Reclosers (AR) which are high voltage circuit breakers. Basically, they are constituted with integrated current and voltage sensors as well as protection relays that are optimized for being used as an overhead network distribution protection asset. Therefore, there is a necessity to develop a modelled control circuit synchronized with the circuit breaker that, in a case of occurrence of any fault, without any human interference, the breaker recloses automatically. There are two types of failures: transient or persistent. In the first one, the AR avoids outage for longer duration at distribution level. However, in the second one, the AR isolates the affected area in the system and avoids outage in the other parts of the system [2].

Additionally, for this goal to be achieved, every grid utility should be able to communicate between each other [3] to balance the electricity generation and consumption in order to keep the power system safe. This

communication has to be reliable and secure. For this reason, the International Electrotechnical Commission (IEC) has introduced IEC 61850 standard to solve the interoperability [4] among different Intelligent Electronic Devices (IED) from different manufactures within a substation automation domain. An IED is a microprocessor based device where all the protective, control and other functions will take place. The main idea of IEC 61850 is to break down the functions of IEDs into core functions called Logical Nodes (LNs). Several LNs can be grouped into a Logical Device (LD) which provides communication access points of IEDs.

1.2 Problem Statement

The topic of this Dissertation was presented by EFACEC Power Solutions to the Institute of Telecommunications of the University of Aveiro. The work was performed at the Institute of Telecommunications of Aveiro, supervised by Professor Rui Luis Andrade Aguiar and with Professor Daniel Nunes Corujo from IT Aveiro. Also present in some meetings from EFACEC were: Rui Dias Jorge, Claudio Silva and Ana Aleixo.

Considering that both EFACEC and Institute of Telecommunications of Aveiro want to be at the forefront of technology, there was a necessity to make a study about IEC 61850 and a survey about potential technologies where IEC 61850 could be applied, giving the suggestion to use the mobile infrastructure as principal technology to communicate between devices from different substations.

Once IEC 61850 standard defines different time requirements for the messages processed between IEDs, it is important to analyse their operation time under different conditions. In a way to be even more effective, all these tests should be performed under real conditions, using an implementation to catch all the real time delays that a simulator couldn't. Additionally, there is a deficiency of studies regarding the use of real libraries when compared with the use of simulators, so it becomes relevant to explore this aspect.

Until now, IEC 61850 protocols such as Manufacturing Message Specification (MMS), Generic Object Oriented Substation Events (GOOSE) and Sampled-Values (SV), have been used with Ethernet which is a widely used Local Area Network (LAN) protocol for the underlaying communication network. However, a new paradigm is coming and using just LAN isn't enough to fulfil the requirements of an intelligent and complex system. Instead of LAN, it will be needed a Wide Area Network (WAN) technology in a way to realize the communication between the smart meters and data centre. At the same time, Ethernet setup is expensive and non-flexible since cables are needed which is not the ideal for large areas, thus introducing difficulties for applying IEC 61850 in energy distribution networks. If wired may not a solution, since money matters when the implementation comes, it is time to make a research and find a good wireless solution that meets the requirements. IEC 61850 requires low latency and high bandwidth which limits the amount of choice that we could do about the technology that could be implemented.

Considering 4G Long Term Evolution (LTE), a technology that is widely implemented over the Portuguese territory, it seems a good principal support for IEC 61850 once that with 4G-LTE it will be possible to cover almost every part of the territory and to have low latency and high bandwidth. Additionally, this may be an excellent technology since it isn't needed to install any additional network infrastructures for the implementation of IEC 61850 standard. On top of that, its maintenance would be done by the mobile operator, significantly reducing the costs. However, not everything is perfect and there are some drawbacks to consider: base stations will be shared with mobile operator users that could lead, in exceptional cases, to an extra flow of traffic, thus increasing the latency times and reducing the bandwidth available.

In certain cases, wired options can also be valid options when compared with 4G-LTE ones, since not every part of Portugal territory is covered by 4G as referred before. So, in this case, using a wired connection may be cheaper than place a new base station. Considering this, it's still relevant to analyse others wired technologies for these exception cases.

Last but not least, IEC 61850 standard allows two options to make the devices from different LANs communicate between each other, using tunnelling or gateway approach [5]. There is the necessity to analyse the impact of each approach under the same test conditions, in order to promote the best option in terms of implementation.

In terms of contributions to the chosen library, libIEC61850, it will be added a module that will act as Gateway Sender where the GOOSE messages will be encapsulated in UDP payload to test the gateway approach. To the tunnelling one, it will be deployed a VPN Server and connected its clients, allowing all the devices to be in the same VLAN. Additionally, it is important to remark that exits some scarcity of work in the field when it comes to comparing both approaches, which makes it important to contribute with different kind of tests and its respective results.

1.3 Objectives

Tony Robbins said: "Setting goals is the first step in turning the invisible into the visible". After understanding the needs of EFACEC, what technologies will be necessary in the research and the conditions available to deploy all the work, all the condition are gathered to set up the objectives that are desirable to accomplish:

- Be familiarized with the IEC 61850 standard, its network protocols and its technologies with the focus in the messages transportation;
- Research about previous work performed and understand which hardware will be needed to assembly the test network;
- Perform the proper modifications and execute valid network tests with different technologies and mediums;

- Compare the results and conclude among the tests performed if they are valid and which ones may interest EFACEC in terms of business perspective as well as what could be improved in the future.

Last but not least, as student and as person, in the end of the work I hope to acquire valuable knowledge about the topics approached and soft skills, as well as work methodology, that could be useful in the competitive job market.

1.4 Document Structure

The document is composed by 6 chapters where it is described the evolution of the work perforemd. Besides this description about how the document is structured, in the chapter 1, ***Introduction***, it is presented an initial approach to the theme Smart Grids as well as analysed the needs and defined the objectives.

The next two chapters are detailed to acquire a background that will be useful to understand the practical part. In the chapter 2, ***State of the Art***, is approached not only all the network protocols but also some other ICT concepts required. Here, also the IEC 61850 standard is introduced. After, in the chapter 3, ***Technical Description of IEC 61850***, there is a deeper description about the IEC 61850 communication mechanisms as well as its communication stack.

Jumping into chapter 4, ***Design Network Architecture and Implementation***, the practical part starts. Initially, some data provided by EFACEC is analysed and it is presented a short description about the chosen library as well as the hardware used to perform the work. After, there is description about the work performed in order to prepare the devices and the code developed to realize the necessary captures.

In the second chapter related with the practical part, chapter 5, ***Results***, there is initially explained what the procedures are to capture the messages when they are sent by both communication mechanisms. After understanding how both capture procedures are performed, the results for the Gateway approach are displayed in a table to compare the technologies used and then, is made the same thing for Tunnelling approach. A recap and final conclusions are withdrawn, comparing both approaches.

Lastly, in the chapter 6, ***Conclusions and Future Work***, it is an overview about what was achieved and what was not, as well as what could be the next steps to continue the work made until now.

2. Technological Background

2.1 Introduction

In this second chapter, it will be described the concepts that are necessary to solve the problem and accomplish the objectives presented in chapter one. Initially, it is essential to be familiarize with Substation Automation Systems (SAS), which are complex systems composed with intelligent devices to protect the electrical grid using different communication protocols. In the second place, there is a description about the IEC 61850 standard and its related protocols that will be used in the practical part. After understanding the IEC 61850 protocols, there is the description about OSI model in order to notice which layers of IEC 61850 will require more attention. Next, it is presented Ethernet that lays in layer 2 of OSI model and understand how a LAN works, followed by IEEE 802.1Q where is described how Virtual LANs (VLANs) are created. Climbing in OSI model, is introduced the Internet Protocol (IP) and its well know protocols that IP work with, UDP and TCP. Also, they are presente in the IEC 61850 and later, during the practical work, UDP will be widely use. To achieve the other communication mechanism for the messages that travel to outside of the substation, it is necessary understand what a VPN is and its related protocols.

After finishing to understand the structure and its communication protocols, it is required to understand what mediums that are suitable to implement the standard IEC 61850. Here, there are a few things that is necessary to be aware of, such as bandwidth and latency of each wired and wireless technologies.

As soon as wired and wireless technologies are analysed, the 4G-LTE will be highlighted as a potential desirable technology and hence, there are shown the available mobile operators that operate in Portugal. It is made a short description about each one and displayed a table with information about the mobile spectrum and the technologies provided by each one. Then, even more important, is analysed how its network infrastructures are and where its antennas are distributed over the country, to find possible gaps and constraints that are needed to be aware of.

Last but not least, it will be presented the embedded systems, its advantages and disadvantages as well as possible suitable single boards to work with. In the end, a good and solid background is acquired with this state of the art, allowing to proceed safely to the chapter 3 where the standard IEC 61850 will be deeply studied.

2.2 Substation Automation Systems

There are two goals that everyone wants to achieve: green and clean energy. With the demand of energy, power management has become one of the areas that requires more attention over the world [6], since energy is a basic necessity.

Formerly, the power grids were very limited regarding what technologies were used, but nowadays the world assists to the increase and growth of digital communication technologies in terms of performance and reliability. To keep up with this evolution, SAS are based on dedicated software that are embedded in pieces of hardware [7]. Additionally, SAS provide an easy way to control and monitor all the equipment in the substation either locally or remotely. Supervisory Control and Data Acquisition (SCADA) are systems that provide to the users an Human Machine Interface (HMI) that can be used for controlling, monitoring and protection of the devices. Plus, SAS execute this controlling and monitoring in real time, helping this way to maximize the availability, efficiency, safety and data integration, leading to a significantly reduce of costs [6].

Moreover, it is introduced the IEC 61850 standard and its related communication protocols that will be used within a substation and between substations. Therefore, implementing the MMS, GOOSE and SV protocols allow the controlling and monitoring in an efficiency way, turning the system more robust as well as future proof [8]. Yet, not everything is perfect, since at the beginning of the appearance of the IEC 61850, the vendors started to compete between them to sell their products with their own system and methodology. Being aware of this competition, an additional work was done in engineering configurations and implemented some rules that allows multi-vendor interoperability as well as increased network performance and tightened the time synchronization, resulting in an easier testing and validation of the installation of SAS [9].

The inclusion of integrated testing is the most efficient approach to test a SAS due to its capability of evaluating the performance of multiple aspects of the system simultaneously. Together with IEC 61850 and process buses, real-time power system simulators have the possibility to interact with multiple IEDs and at the same time simulate the electrical substation with an high level of precision.

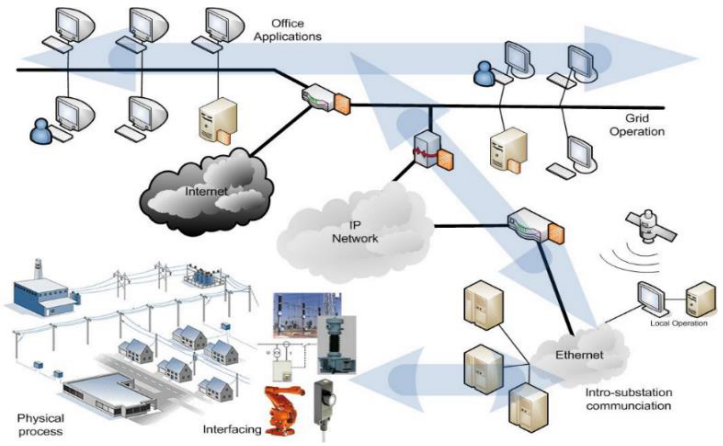


Figure 2.1: Illustration of Substation Automation System [99]

2.3 IEC 61850

2.3.1 Introduction

IEC 61850 is part of the IEC Technical Committee 57 (TC 57) [11] architecture for electrical power systems and it was initially developed for communication in SAS by the Institute of Electrical and Electronic Engineers (IEEE) using IEDs, in 1980-90s [12]. As time went by, these old devices that usually were only capable to use one function, were replace with new ones, capable to perform multiple functions. Unfortunately, these multiple functions lead to other problems: the impossibility to track what logical functions were doing what and where exactly they were located, resulting in unwanted problems. In Figure 2.2, it is shown the evolution from hardwire proprietary protocols until nowadays using the standard IEC 61850.

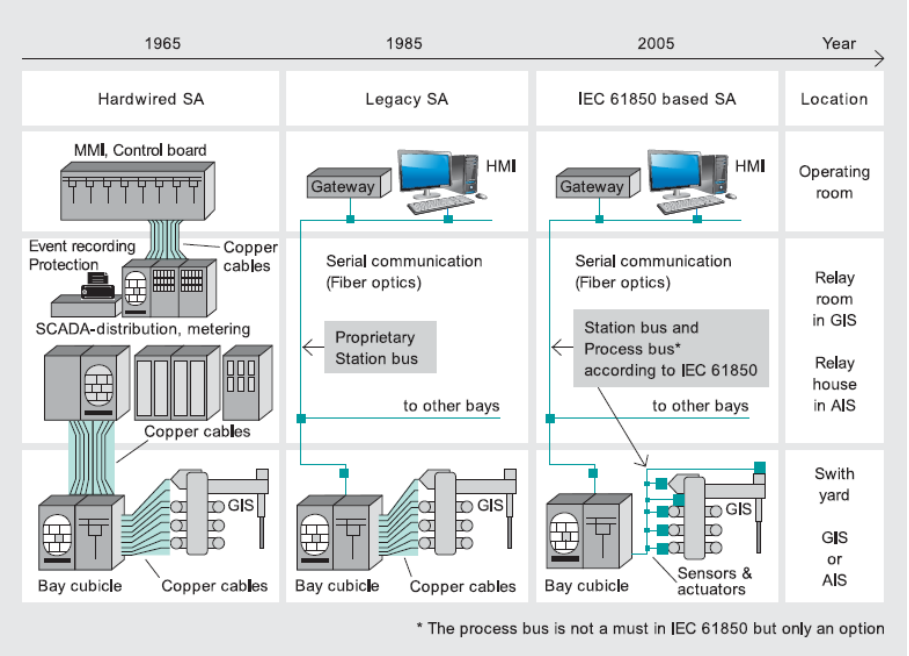


Figure 2.2: Evolution of Substation automation (SA) architecture [21]

In the beginning, each IED has its own proprietary protocol and, when combined multiples IEDs from multiples vendors an obvious problem occurs, it was not possible to have interoperability among them. Interoperability means that, all the devices in the same network or communication path could share information and commands between them, with no problem even if they aren't from the same vendor. Therefore, IEC 61850 came to solve this problem and allow clients to use different devices from different vendors [13].

Despite the initial scope of IEC 61850 being focusing in communications within substations, rapidly the vendors noticed that there are benefits to incorporate other domains such as Distributed Energy Resources and

Hydroelectric Power Plants. In Figure 2.3, it shows that IEC 61850 can also be used to support communications from control centres, SCADA, to the RTUs and DERs [14].

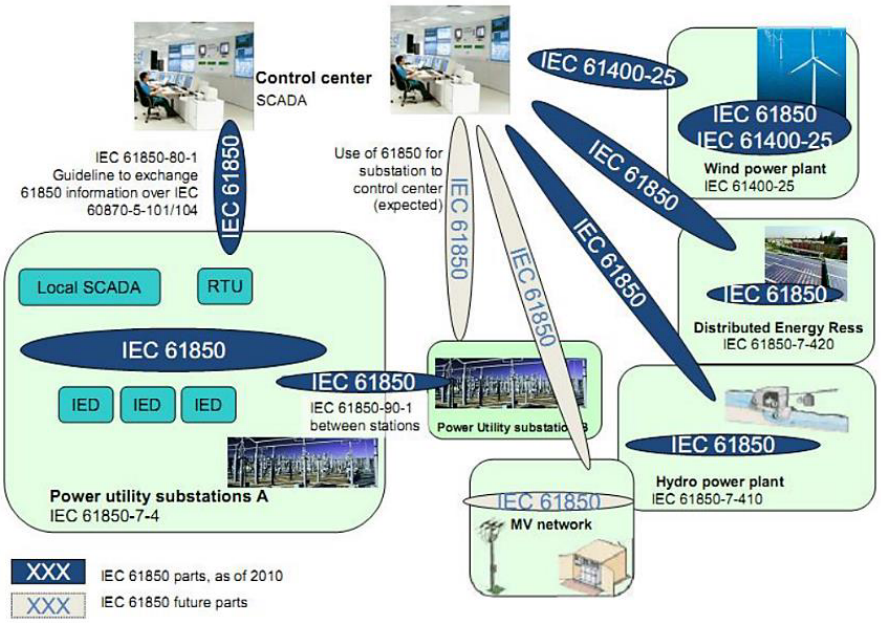


Figure 2.3: IEC 61850 applications [95]

2.3.2 IEC61850 Documents

IEC 61850 defines the various aspects of the power utility and, in Appendix G, there is the Table 7.1 that shows how the standard is divided.

Table 7.1 is a long and extensive table composed with parts that belong to IEC 61850. It's composed mainly with 10 parts, however TC 57 continues to review old parts as well as publish new ones. Plus, there is a short description about each part [15] since the IEC 61850 is complex and it's important to be aware of each part:

IEC61850-1: The 1st part is a general overview of the standard including the history, philosophy and introduce the others 10 parts of the standard as well.

IEC61850-2: The 2nd part is dedicated to the terms and definitions used within a SAS and that will be applied in the remain standard.

IEC61850-3,4,5: These three parts refer to the general requirements, involving network and communications of the system. Quality, reliability, maintainability, availability, security and environmental conditions are referred.

IEC61850-6: The 6th part is described the Substation Configuration description Language (SCL), which is used to write the IED configuration.

IEC61850-7: The 7th part is a longer part composed with different topics. They include the basic communication structure and definition of abstract services and data objects, referred as Logical Nodes. Abstract definitions allow to make the mapping of the data objects and services to any other protocol that will meet the data and services requirements.

IEC61850-8: The 8th part is where the abstract data object and services will be mapping into MMS. GSSE/GOOSE are present in this part

IEC61850-9: The 9th part defines the mapping of Sampled Values into ethernet frames. Both parts, 8 and 9 are pretty import and will be more detailed further since they are the communications aspects of IEC 61850 that will be used.

IEC61850-10: Last, the 10th part approaches the number of techniques to perform conformance tests.

In addition to these 10 parts, there are others that will required also attention like IEC61850-90-1, IEC61850-90-2 and IEC61850-90-5. So, the focus of this document will be mostly the standards in the next table:

Reference	Publication	Title
IEC 61850-8-1	2011	Specific communication service mapping (SCSM) - Mappings to Manufacturing Message Specification MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3
IEC 61850-9-2	2011	Specific communication service mapping (SCSM) - Sampled values over ISO/IEC 8802-3
IEC 61850-90-1	2010	Use of IEC 61850 for the communication between substations
IEC 61850-90-2	2016	Using IEC 61850 for communication between substations and control centers
IEC 61850-90-5	2012	Use of IEC 61850 to transmit synchrophasor information according to IEEE C37.118

Table 2.1: Important parts of IEC 61850 used over the document

To sum up, parts IEC 61850-8-1 and IEC 61850-9-2 explain how the communications within substations works, such as GOOSE and SV. On the other side, the parts IEC 61850-90-1, 90-2 and 90-5 refer to the communications between substations and substations and control centres, translating into communications to outside of the substation.

2.3.3 Data Modelling

Standardizing a data model and date exchange is completely mandatory in order, as referred before, to guarantee interoperability between devices from different manufactures. The IEC61850 data model will be described in the following sectors [16].

2.3.3.1 Physical Device

As the name suggests, the Physical Device (PD), that could be an IED, is an hardware piece that is connected to the network and has its own network address that allows it to be identified from the others PDs in the same network.

2.3.3.2 Logical Device

LD is defined as "entity that represents a set of typical substation functions" and is composed by different LNs that conceptually belong together [17], as illustrated in Figure 2.4.

2.3.3.3 Logical Node

A LN is a function of the group of functions that continues the set of functions of LD. On other words, in the same part of 7-2, is defined as "entity that defines a typical substation function".

Here, different applications are performed inside each LD such as control, protection, monitoring and measurement. The ones represented in Figure 2.4 are Measurement Units, identified as MMXU. If there would be a Circuit Breaker (CB), it would be identified as XCBR.

Additionally, each LN is decomposed into data objects and data attributes [18]:

- **Data Objects:**
  - Inside of each LN are different data objects. They can belong to various categories such as common data information, status info, settings, measured values and controls. Plus, they are derived from a Common Data Class (CDC) which serves as predefined building blocks for creating larger data objects.
- **Data Attributes:**
  - Each data object has several data attributes such as:
    - stVal – Status value
    - q – Quality
    - t – Timestamp
    - d – Description

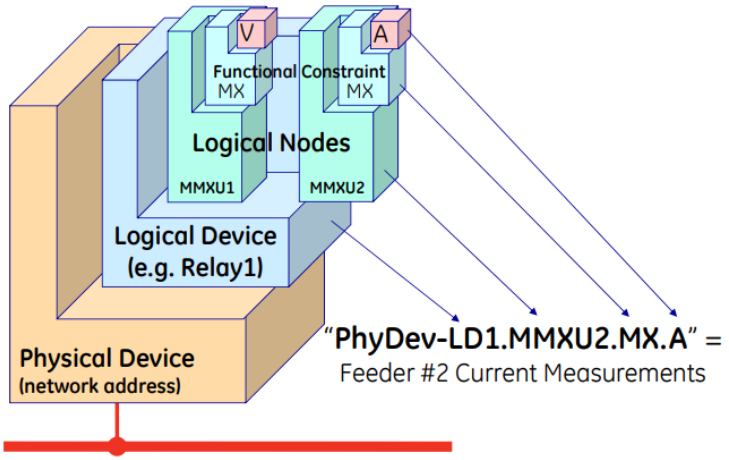


Figure 2.4: IEC 61850 Data Modelling [93]

2.3.4 Mapping to Communication Protocols

IEDs functions are constantly delivering data, being this data organized in a standardized structure through object models. Thereupon, Abstract Communication Service Interface (ACSI) is a virtual interface to an IED that provides abstract communication services since connections, variables access among others, including the ones used for exchanging the information represented by Data and Data Attributes. Therefore, no matter how ACSI is defined, it's independent of underlaying protocol. At the same time, IEC 61850 also provides a Specific Communication Service Map (SCSM) that maps ACSI models into real protocols, delivering this way a pleasant symbiose [15].

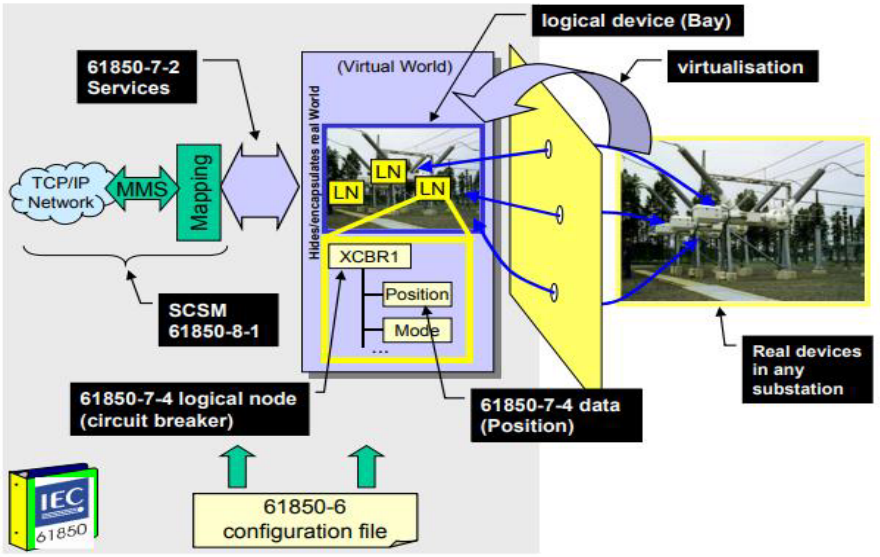


Figure 2.5: Mapping from the real to the virtual world [94]

2.3.5 Manufacturer Message Specification

MMS is an international standard, ISO 9506, that was developed to allow the delivery between networked devices and computer applications as well as supervisory control information [19]. In other words, MMS was developed to standardize and optimize the automation of the industrial processes.

An unquestionable favourable point of the MMS is that it was designed to be vendor-independent, granting the interoperability among devices from different vendors. Also, due to this reason, IEC 61850 is highly influenced from MMS standard [4]. As shown in Table 2.1, in the 8th part of IEC 61850, explains the mapping of ACSI to MMS, providing norms for the mechanisms and rules necessary to implement the objects and services of the ACSI by the MMS ones.

As more manufactures start to produce more devices with the same propose, it was needed to achieve interoperability between them in a way than could be possible to have devices from different manufactures in same system. Hence, this lead to a necessity of development of one standard. MMS is an international standard that was published for the first time in 1988 and embraced in general automation systems. Later, it was revised, getting this way an optimization and started to include communications among computers and intelligent devices of all kind. MMS protocol and services are projected to perform over compliant OSI and UDP/TCP communication profiles [20].

2.3.6 GOOSE

Looking to one of the most important parts where the GOOSE is referred, it is important to notice that in part 7-2 [16], ACSI defines two groups of communication services:

- Peer-to-Peer model
- Client-Server model

On the first one, Peer-to-Peer model it is used for Generic Substation Event (GSE) services for protection proposes. These GSE messages are associated to time critical activities, which requires a fast and reliable communication between IEDs.

The second model rests in a Client/Server mechanism, supporting this way the distribution of same GSE information for more than one device, through broadcast/multicast services. GSE model include two services:

- Generic Object-Oriented Substation Event, GOOSE, which is used to exchange of a wide range of possible common data.
- Generic Substation State Event, GSSE, which provides the capability to transmit state change information

With focus on GOOSE, these messages are exchanged in same LAN through multicast or broadcast. As consequence of some of them are time critical messages, they require a special attention since it is needed to understand if the time requirements are fulfilled, and more important, if these messages are involved in protection functions, which will give them high priority.

Finally, it is important to notice that these messages are present in Application layer, Data-Link layer and Physical Layer of OSI model [21], as illustrated in Figure 2.6.

2.3.7 Sampled Values

As the name suggests, Sampled Value services are defined for the fast and cyclic transmission of sampled raw data that is measured by the equipment of the substation. Current and voltage are the most common transmitted data and they are transmitted, like the GOOSE messages, in the same LAN. They will be processed later, in a manner to offer better functions like protection and control [22].

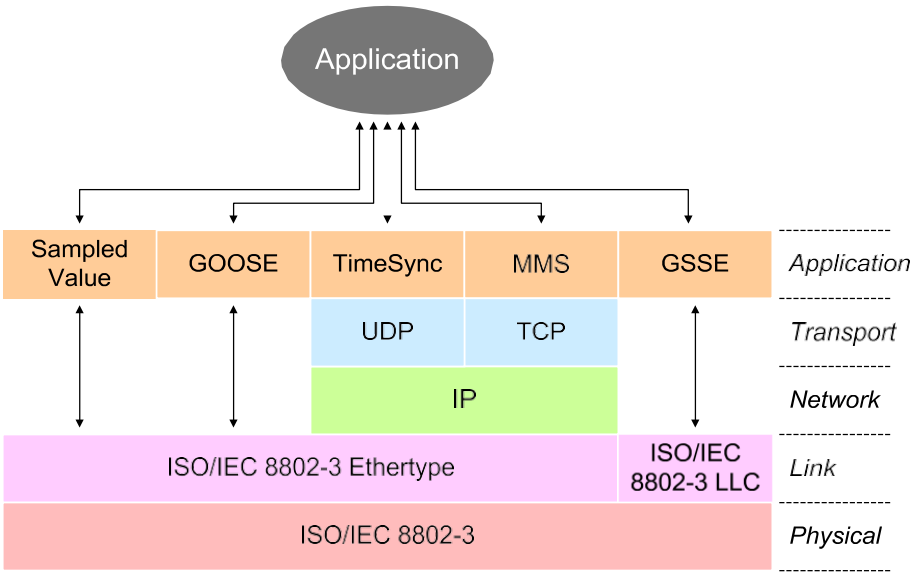


Figure 2.6: Mapping to real protocols in IEC 61850 [21]

2.4 OSI Model

As result of a new will to become the communication services open, in the middle of decade 1970, began by OSI the process of definition of a new standard for architecture of communication networks in way that allows the communication between different equipments [23].

Later, in 1996, its current form was published and organized in seven layers where the model specifies the functional requirements for each layer, without specifying or restricting the protocols to be used in order to achieve interoperability. The seven layers are shown in Figure 2.7: Physic (1), Data-Link (2), Network (3), Transport (4), Session (5), Presentation (6), Application (7). Lower layers, more concretely, layers 1 and 2, have as objective the transmission of data through the physic way that connects the various nodes in network. Upper layers have some roles like: routing messages until its destiny, stablish a connection among the final users and the communication system [24].

Two important books, several papers [23] [24] [25] [26] [27] as well as some websites were consulted, where could be withdrawn information about OSI model. Below, there is a brief description about each layer.

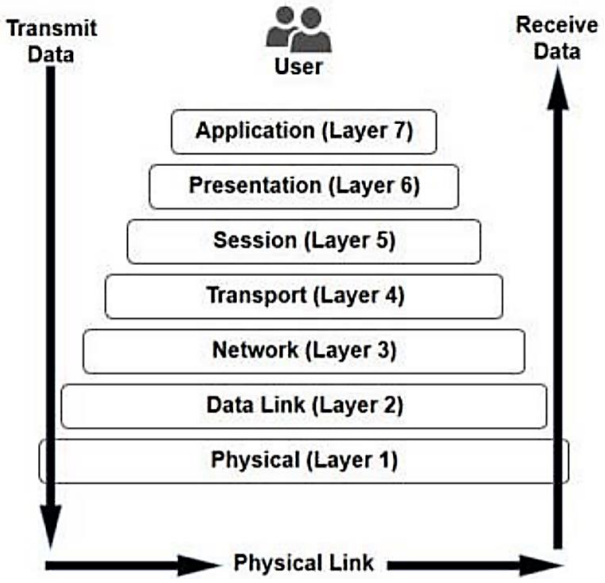


Figure 2.7: OSI Model [27]

2.4.1 Physical Layer

Physical layer is the first layer and defines the electric and physical specifications of devices. More important, defines the relation between a device and a transmission medium, such as copper or optic fibre cable. Electro-mechanical components handle the transmission/receiving of raw data in the form of bits (0/1). Bit control is also done in physical layer as well as the definition if it is simplex, half or full duplex.

2.4.2 Data Link Layer

Data Link Layer is responsible for setting up links over the physical network and, at the same time, has the capability to correct errors that happen in physical layer. It's divided into:

a) MAC layer – responsible for controlling the access of the communication medium and grant permission to transmit data from devices, in order to prevent data collision.
b) LLC layer – responsible for performing error checking, frame synchronization and flow control.

2.4.3 Network Layer

Unlikely the data link layer, the network layer is capable to send packets from one node to another one connected in a different network. This layer also provides the functional and procedural means to transfer variable length data sequences. The network layer not only operates with routing functions but can also realize fragmentation, message reassembly and further deliver error reporting. For example, a very well known function, is IP addressing.

2.4.4 Transport Layer

Transport layer is responsible to guarantee data delivery between two or more networked devices as well as to perform some functions like reliability control of the communication link via flow control, segmentation/desegmentation and error control. A key feature of transport layer rests in separation of physical layers (1 to 3) and application layers (5 to 7). So, it is needed to make the connection among these two important parts jointly with the determination of service class. Additionally, important protocols like UDP and TCP are used in this layer.

2.4.5 Session Layer

Session layer is accountable for establishing, maintaining and terminating the session between two or more networked devices. Even if communication fails, network devices restart the transmission of data from the last mark received by the receiving device.

2.4.6 Presentation Layer

Also referred as the syntax layer, it has responsibility to convert the data format received by the application layer into common format to be used for transmission. In other words, it makes the conversion between application and network formats and vice versa. Other important functions present in presentation layer are data translation, data encryption/decryption and as said before other protocols conversions.

2.4.7 Application Layer

To conclude, the last and top layer of the OSI model, application layer, works with application or programs in a way that allow the interaction between machine-user. Everything in this layer is related with software and some protocols that are acquainted are, for example, HTTP, SMTP, FTP POP3, IMPAP, BitTorrent, DNS and ICMP.

2.5 Ethernet

After understanding the OSI model, it is necessary to understand the technologies that are available and are recommended to use in IEC 61850. Ethernet is one of them and GOOSE and SV messages use ethernet frames for their transmission

Ethernet belongs to the data link layer of OSI model and it's the most used technology for area networks, such as LAN and WAN. It's divided in two main parts for transmission of data: the packet and the frame. Among others, the frame contains the payload, addressing information about the physical-MAC address of sender and receiver, VLAN and priority tagging, information about quality of the service and error correction information. About transmission speeds, they are indeed pretty good since they usually vary between 100Mbps and 10Gbps, which give enough bandwidth to work with.

2.6 IEEE 802.1Q supporting Virtual LAN's

A LAN consists in the use of Ethernet devices where they communicate between them using layer 2 protocols. With the capability of IEEE 802.1.Q standard to create Virtual-LANs, a VLAN identifier is present in Ethernet packet, called VLAN tag. This VLAN tag is composed with 32 bits, being divided this way [28] [29]:

- **Tag protocol identifier**
  - Uses 16 bits and it is set to 0x8100 hexadecimal value in order to identify the frame as an IEEE 802.1Q tagged frame,

- **Tag control information**
  - It also uses 16 bits and it is divided in 3 subfields
    - Priority code point – 3 bits to identify the priority
    - Drop eligible indicator – 1 bit indicates if the frame is eligible to be dropped in the presence of congestion
    - VLAN identifier – 12 bits to identify VLAN. 0x000 and 0xFFF are reserved and cannot be used, leaving 4094 possibilities

Being aware of each field allows to understand some of the functionalities of VLANs. With the VLAN identifier, it can divide logical systems into logical subnets. Different subnets can be ruled differently, regarding which protocols are used.

Another thing that has to be pointed is that devices in different VLANs cannot communicate between each other by default, as if they were not in the same LAN. And, considering this, it comes others important topics and advantages such as extra security, network segmentation and simplified network management [30]. Important to remark that VLANs can be tunnelled through layer 3, which allows connect devices that aren't in same physical location, communicate if they were due to be in same VLAN.

VLANs are approached in point 7 of IEC 61850-90-1, allowing that the packets being isolated by its VLAN ID. This is an improvement in terms of traffic velocity, security and protection since it separates the critical messages with high-priority from the low priority ones [31].

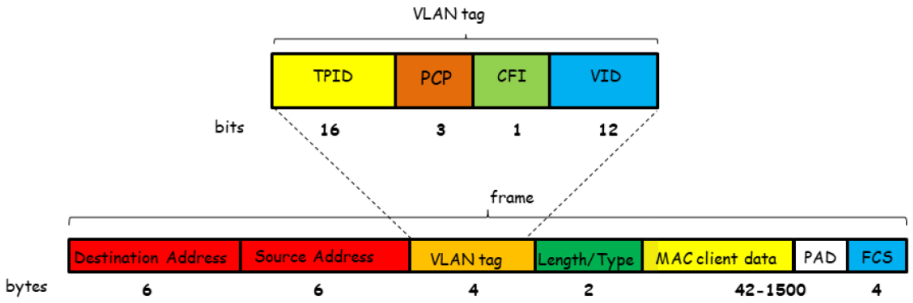


Figure 2.8: IEEE 802.1Q Frame [29]

2.7 Internet Protocol

As well know, IP is the principal communications protocol allowing routing data across network boundaries. This protocol belongs to the network layer of the OSI model but, it's almost a two layer protocol since it collaborates most of the time with User Datagram Protocol (UDP) and Transmission Control Protocol (TCP). The process of transmission of data settles in, IP receiving datagrams and giving them the IP address of the

sender and the receiver [32]. Routing is the process of choosing the best path and, even if there are multiple packets of the same message, they can be sent by different paths.

2.8 UDP and TCP

After understanding the OSI model, it is necessary to understand how UDP and TCP work since they will be used in part of this work. Either protocols are present in IEC 61850 and they can be used as the transport protocols for outside of substation, commonly named as R-GOOSE or R-SV. Thereby, both protocols belong to the Transport Layer of OSI model and their simplified structure is displayed below, in Figure 2.9.

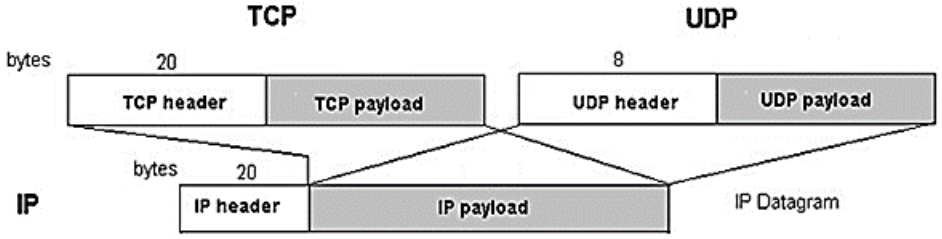


Figure 2.9: Simplified TCP/UDP-IP Frame [92]

2.8.1 UDP

The UDP is a simple protocol, designed by David Reed in 1980 and formally defined in RFC 768 [24]. Using a simple connectionless communication model, no prior communications are required in order to set up communication channels or data paths. As result, UDP doesn't send acknowledgments even that data doesn't reach the receiver where it can be concluded it doesn't offer a reliable data transmission. Running on top of IP, unlike it, it offers port numbering and helps to distinguish among different user requests and thus, also referred as UDP-IP.

Contrarily to TCP, UDP does not offer congestion control and data retransmission and for this reason, it needs less bandwidth and lower latency. In short, it's a good compromise for network applications where latency matters and some packages can be dropped like gamming or VoIP [33]. Also, in GOOSE, latency is a big requirement and will be shown why UDP will be the best choice.

Another interesting characteristic is that UDP can be configured to retransmit again, make the UDP usable in applications that are vulnerable to data loss. Of course, in the end this will lead to some redundancy but for GOOSE this is ideal since some messages are critical for proper operation of SAS.

2.8.2 TCP

TCP was developed in 1969 as a resource for an experimental project called ARPANET (Advanced Research Project Agency Network). Several years later, in 1990, the TCP was finally standardized. TCP is one of the most important protocols nowadays [34] and it's a bit the opposite of the UDP in certain aspects. It's a connection oriented protocol, which means that it needs set up and maintains the connection until both applications involved in transmission finish the data exchange. Then again, contrary to UDP, TCP performs congestion control and a non-error data transmission once it retransmits the lost packages. Additionally, this leads to better reliability comparing with UDP since an acknowledgment of the successful receiving data is received from the receiver [34], confirming that the data was delivered. In addition to that, and since the data could be retransmitted, these procedures will request more bandwidth and induce more latency making it not ideal for time critical applications.

2.9 Tunnelling and VPN

An additional area that is required to analyse is tunnelling and its related protocols. So, VPN is a process that extends a private or local network through a public one as illustrated in Figure 2.10. This process, that allows users to send data across a tunnel created to be safe and secure [35], is called tunnelling and allows users that are connected to different networks to be in the same LAN even if they are physically distanced between each other.

Initially, VPN technologies were developed for professional use, since it was necessary to access documents and order commands from the office even if the worker was not there [36]. Later, with the development of technologies, the emergence of 4.0 industry and IoT, even common users felt necessity to use this technology to control their home appliances or technologies.

In VPN transmission, data frames are encrypted while the protocol used to transport them could be UDP for as fast deliver or TCP thereby ensuring that everything reaches its destiny. Also, there are some types of VPN tunnels that operate over the 2nd and 3rd layer of the OSI model [35]:

- PPTP and L2TP are both protocols that operate in Layer 2, Data-Link Layer
- IPsec can operate as VPN protocol in Layer 3, Network Layer
- OpenVPN [37] is a free and open source software that provides VPN techniques and operates over both layers.

OpenVPN will be the choice in practical part, being deployed a VPN server and connecting its clients as explained in 4.8.4.

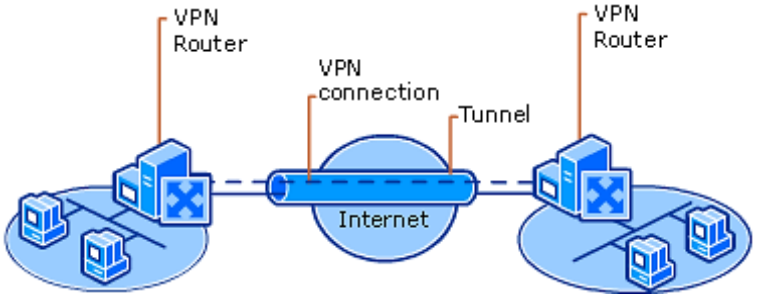


Figure 2.10: VPN example [91]

2.10 Wired and Wireless Technologies

2.10.1 Introduction

After a short description of IEC 61850, it's required to choose the technology where the data will be carried. In other words, if the medium will be wired or wireless and, inside of each one what will be the technologies that can accomplish the time requirements needed for IEC 61850. Looking for the Table 3.1 that is ahead with the time requirements, it is possible to observe that there are messages that don't need the same amount of time to be delivered since they may or may not need to leave the LAN of the substation. For messages that have to travel longer distances, wireless technologies could be cheaper than wired, since longer distances require longer cables, however wireless technologies also have some drawbacks to take in consideration.

2.10.2 Wired

In the Table 2.2 [38], there are three popular wired technologies used to carry data. Considering this table:

- Optical Fibre is the best choice among these three technologies. Theoretically, it achieves the latency and data rate requirements, however it is expensive to deploy.
- X-DSL is also expensive to deploy and could not guarantee both requirements. In comparison with Optical Fibre, Optical Fibre is better in every way, but DSL can use the existing copper lines.
- PLC could be a good alternative, but unfortunately it has a big problem, a noisy channel environment. In this case this is a huge problem, since the wired would be close to high/medium voltage transmission lines, which would deliver a lot of noise to the channel.

Technology	Spectrum	Data Rate	Latency	Coverage Range	Cost	Limitation
Optical Fibre	up to 353000 GHz	up to 40 Gbps	3.34 μs per km	up to 100 km	High	High network deployment costs High cost of terminal equipment
X-DSL	20 kHz – 1 MHz	ADSL: 1-8 Mbps HDSL: 2 Mbps VDSL: 15-100 Mbps	10-70 ms	ADSL: up to 5 km HDSL: Up to 3.6 km VDSL: up to 1.2 km	High	Telecom operators can charge utilities high prices to use their networks. Not suitable for network back haul (long distances result into data rate degradation)
PLC	1-30 MHz	2-3 Mbps	5-7 ms	1-5 km	Low	Harsh, noisy channel environment

Table 2.2: Comparison between Wired Technologies [38]

2.10.3 Wireless

Nowadays, wireless technologies are widely used since, they are more convenient to users and they don't need a real physical medium to carry data, turning them cheaper when compared with the wired ones. Thereupon, there is the Table 2.3 [38] that shows a several wireless technologies that have to be analysed in order to see if they are suitable or not for IEC 61850 requirements. At this point, such as before, latency and data rate are both requirements that are necessary to look for.

- Starting from the bottom the last two technologies, NB-IoT and LoRa are automatically discarded, since both latency times are over a second and IEC 61850 protocol cannot afford such delay.
- On the other hand, ZigBee and Z-Wave have a very low latency however, the bandwidth available is not enough to accomplish the requirements, making them not suitable to transfer data in IEC 61850.
- Wireless Mesh also looks unappropriated since it is expensive to implement and its network management is complex, which could lead to network delays, resulting in higher latency times.
- Finally, there are two technologies that could be suitable in terms of latency and data rates, WiMAX and WLAN 2.4GHz, also known as Wi-Fi. After analysing WiMAX, it is easily affected by adverse weather conditions, the number of devices connected in same base station and it's easily interfered with other wireless signals [39].

So, at first selection, Wi-Fi is the one that looks more suitable from the Table 2.3 to use with IEC 61850, however doesn't accomplish another important requirement: an infrastructure that is already implemented, which would permit to reduce deployment costs.

To fulfil this requirement, Mobile operators already have an infrastructure installed on the field, to serve its mobile users. Naturally, it is necessary to understand if the latest mobile technology, in this case, 4G-LTE is suitable for IEC 61850. Below, is shown another table, Table 2.4, where are compared both 4G-LTE and Wi-Fi, norm IEEE 802.11ac.

Technology	Spectrum	Data Rate	Latency	Coverage Range	Cost	Limitation
ZigBee	2.4 GHz-868-915 MHz	250 kbps	15 ms	30-50 m	Low	Short-range
WLAN	2.4 GHz	2-600 Mbps	3.2-17 ms	100 m (indoor)	Low	Power consumption might be too high for many smart grid devices
Z-Wave	2.4 GHz-868-908 MHz	9.6-40 kbps	100 ms	30 m (indoor) 100 m (outdoor)	Low	Short range Low data rate
Wireless Mesh	Various	Depending on selected protocols	Depending on selected protocols	Depending on deployments	High	Network management is complex
WiMAX	2.5 GHz, 3.5 GHz, 5.8 GHz	up to 75 Mbps	10-50 ms	10-50 km (LOS) 1-5 km (NLOS)	High	High cost of terminal equipment Weak diffraction ability
LoRa	868-915-433 MHz	0.3-50 kbps	Average 2 s	3-8 km (urban) 15-22 km (rural) 15-45 km (flat)	Low	Low data rate
NB-IoT	900-1800 MHz	Uplink: < 250 kbps Downlink: < 230 kbps	Less than 10 s	<35 km	Low	Latency insensitive

Table 2.3: Comparison between Wireless Technologies [38]

Technology	IEEE 802.11ac	LTE
Signalling	OFDM	DL: OFDM UL: SC-FDMA
Peak data rate	≈ 7Gbps	DL: 300 Mbps UL: 75 Mbps
Typical latency	50 µs-10 ms	5-30 ms

Table 2.4: Comparison between Wi-Fi and 4G-LTE [40]

2.11 Mobile Networks

Nowadays, people communicate with each other without having be physically present, using voice calls, SMS or any other software present in the device used to communicate, like Skype, WhatsApp, Facebook Messenger or Telegram, among others.

However, the story of mobile communications started a long time ago [41], in 1865 with Maxwell equations developed by James Maxwell. In the same century, in 1887, Heinrich Hertz create the oscillator and in 1895 Guglielmo Marconi developed a transmitter with a long wavelength and high power transmission for telegraph signals. Turning the page, in 1907 Lee De Forest created the vacuum tube capable of transmitting higher frequencies until 1.5MHz.

In the early 1920s, the use of radio communication system emerged, where the Titanic disaster had a great impact. Later, with World War II, radio communications become essential in combat. In 1947, it was possible

to use a cellular network that consisted in a simple set up, using a base station, thus providing signal in a surround area where calls could be made and a terminal to make them.

Until nowadays, it's used base stations that cover large geographical areas, thus allowing the continuous handover. These structures have some advantages such as the possibility to reuse the spectrum, the need of lower transmission power and they are flexible, being possible to be installed in different places. Nevertheless, they have some limitations when compared with wired connections, such as latency and intercell interference [42]. As illustrated in Figure 2.11, the evolution of each generation takes usually 10 years each. It is shown the evolution since 1G to 4G, being each one briefly describes bellow.

2.11.1 1st, 2nd and 3rd Generation

The 1st generation of cellular communications arose in the 80s years. It was very limited, since it used analog signal [42] and the voice quality was generally poor.

In next decade, in 1991, the 2nd generation appeared and was a big improvement in comparison with 1G, since it brought the digital era with more coverage and capacity than 1st generation. Additionally, it was possible to send SMSs and MMSs [43] and brought for the first time data services. Global Systems for Mobile Communications were deployed in Europe and used in more than 200 countries, being the most dominant system used. It supports transfer data rates up to 22.8kb/s [44] and operates in the 900MHz/1800MHz and 850Mhz/1900MHz, depending the part of the globe as illustrated in Figure 2.12.

Ten years later, in 2001 General Packet Radio Service (GPRS) was created with the objective to upgrade the GSM system, giving higher data rate via packet switching. Considering this upgrade, it was possible in theory to have, with a perfect channel, data rates until 171.2kb/s [44] but it was usual that data rates oscillated between 10 to 115kb/s with an average of 40kb/s [44]. Enhanced Data Rates for GSM Evolution (EDGE) also surged with rates between 8 and 60kb/s. However, these data rates weren't enough and better quality of service and low latency was required, leading to the appearance of Universal Mobile Telecommunication System in Europe and Japan as well as Code Division Multiple Access 2000 (CDMA-2000) in the U.S.A. Universal Mobile Telecommunications System (UMTS) supports both Time Division Multiple (TDM) and Frequency Division Multiplex (FDM) and includes two standards: Wideband-CDMA and High Speed Packet Access (HSPA) [43]. A different approach is Time Division Synchronous CDMA (TD-SCDMA), also developed in Europe. It's important to notice that 3rd Generation Partnership Project, a corporation of 6 regional standards groups, was responsible to control the evolution and migration of GSM, 3G and LTE systems.

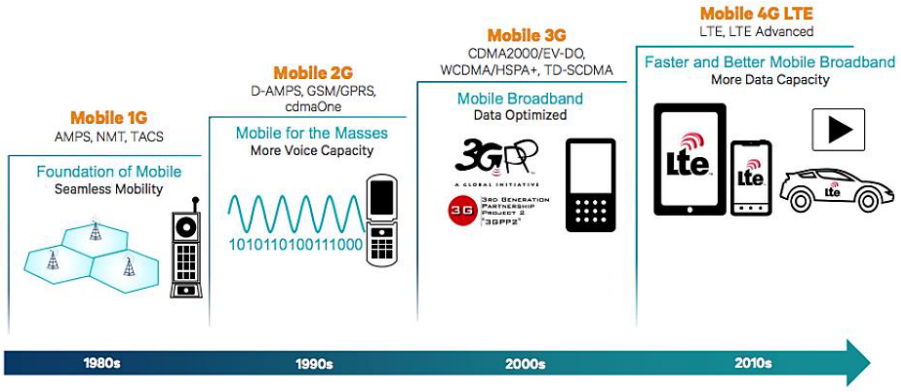


Figure 2.11: Evolution of the Mobile Communication Systems [98]

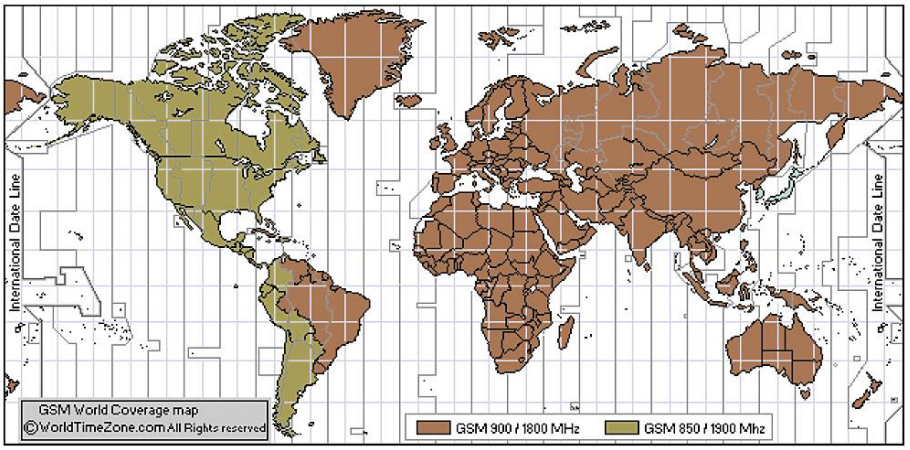


Figure 2.12: GSM Bands Information by Country [87]

2.11.2 4G

In 2007, the first iPhone was launched and other smartphones appeared, requiring even higher data rates and lower latency. As consequence, in 2009, the 4th generation shows up, being marketed as LTE aiming to deliver data rates higher than 100Mb/s and low latency. 4G-LTE brought a big improvement for the system capacity, leading to an increase of the peak data rate over the years. Accordingly to OpenSignal [45], with real performed tests, “mobile operators have elevated the average of 4G speeds beyond 20Mbps, then beyond 30Mbps, and in the last two years, beyond 40Mbps”. However, these values still are a little bit far from the ones that were announced in perfect conditions. Today, 4G-LTE represents more than 2.8 Billion of connections [46].

Unlikely to WiMAX, that uses Orthogonal Frequency Division Multiple Access (OFDMA), at both downlink (DL) and uplink (UP), LTE uses OFDMA at DL and Single Carrier Frequency Division Multiple Access (SC-FDMA) at UL [47]. Looking to the latency and bandwidth values in previous topic, 4G looks like a suitable technology to use in same grids, as illustrated in Figure 2.13.

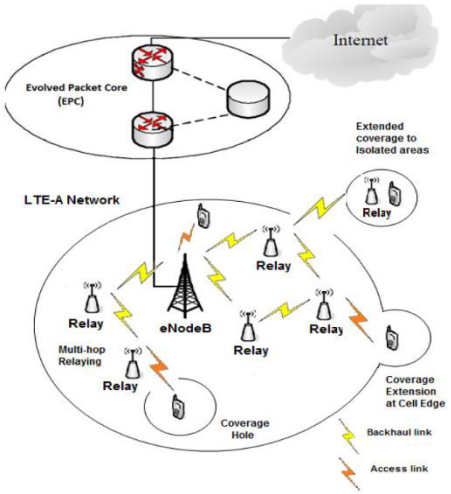


Figure 2.13: LTE-A network applied in smart grids [88]

2.11.3 5G

The new 5G era is around the corner. As seen every day, more and more devices connect through wireless technologies requiring more bandwidth and low latency to the new services that will appear such as video and music streaming, social networking, gaming and other interactive applications. Naturally, the consequence is that is required dozens to hundreds of Mb/s in order to have a good QoS. Therefore, 5th generation is expected to be available in 2020 and it's designed to interconnect billions of small devices with the objective of achieving the minimum data rate of 1Gb/s [48] and 5Gb/s for high mobility users [49].

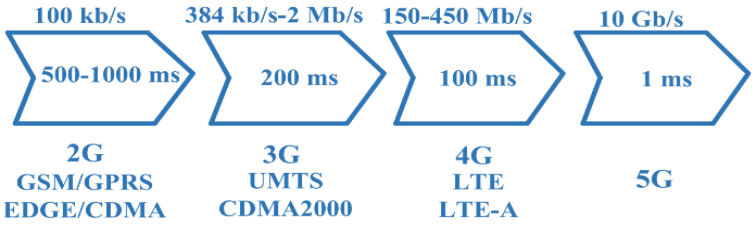


Figure 2.14: Evolution of Mobile Networks [96]

2.12 Portuguese Internet Service Providers

2.12.1 Vodafone

Founded in 16 of September of 1991, Vodafone is a British multinational telecommunications conglomerate, with headquarters in London, England. Its networks are present in 27 countries and it has more than 47 countries as partners [50].

Regarding Portugal, Vodafone bought Telecel [51], starting to use just the name Vodafone since 2001. Nowadays, it delivers to its clients fixed and mobile telecommunications services.

2.12.2 Meo

Meo is a company that belongs to Altice Group [52] that also delivers to its clients fixed and mobile telecommunications services. It started in the beginning of 1990s and since then there are incorporations and separations with other companies.

2.12.3 NOS

More recently, ZON and Optimus, both telecommunications companies joinned in order to create NOS in 2013 [53]. NOS also provides both fixed and mobile telecommunications services such as its competing operators.

2.12.4 NOWO

Formerly Cabovisão, NOWO, it is the 4th bigger TV cable operator. In terms of mobile service, in 2016 was celebrated an MVNO agreement where NOWO uses MEO operator network, allowing them to have 4G speeds over the country [54].

2.12.5 Mobile Technologies and Spectrum

Below, there is the Table 2.5 that shows the technologies and the frequency used by each mobile operator. Looking for the table and each technology, along with the previous research, it is possible to observe that they offer high speed data rates. Both, Vodafone and MEO will be tested with 4G-LTE while NOWO and again, MEO, will be tested through Optical Fibre and ADSL.

Operator	Downlink		Uplink		Band		Technology
MEO	791,0	801,0	832,0	842,0	800	MHz	Technological Neutrality
MEO	950,9	958,9	905,9	913,9	900	MHz	GSM / UMTS / WIMAX / LTE
MEO	1845,0	1865,0	1750,0	1770,0	1800	MHz	GSM / UMTS / WIMAX / LTE
MEO	2149,9	2169,7	1959,9	1979,7	2100	MHz	UMTS
MEO	2670,0	2690,0	2550,0	2570,0	2600	MHz	Technological Neutrality
NOS	811,0	821,0	852,0	862,0	800	MHz	Technological Neutrality
NOS	943,1	950,9	898,1	905,9	900	MHz	GSM / UMTS / WIMAX / LTE
NOS	1825,0	1845,0	1730,0	1750,0	1800	MHz	GSM / UMTS / WIMAX / LTE
NOS	2130,1	2144,9	1940,1	1954,9	2100	MHz	UMTS
NOS	2650,0	2670,0	2530,0	2550,0	2600	MHz	Technological Neutrality
Vodafone	801,0	811,0	842,0	852,0	800	MHz	Technological Neutrality
Vodafone	930,0	935,0	885,0	890,0	900	MHz	GSM / UMTS / WIMAX / LTE
Vodafone	935,1	943,1	890,1	898,1	900	MHz	GSM / UMTS / WIMAX / LTE
Vodafone	1805,0	1825,0	1710,0	1730,0	1800	MHz	GSM / UMTS / WIMAX / LTE
NOS	2650,0	2670,0	2530,0	2550,0	2600	MHz	Technological Neutrality
Vodafone	801,0	811,0	842,0	852,0	800	MHz	Technological Neutrality
Vodafone	930,0	935,0	885,0	890,0	900	MHz	GSM / UMTS / WIMAX / LTE

Table 2.5: Frequency Distribution in Portugal [55]

Note:

- Downlink – communication made from the base station to the device
- Uplink – communication made from the device to the base station

2.13 Portugal Topology

Last but not least, it's necessary to analyse the distribution of 4G antennas over the territory since the latency and bandwidth values only apply in perfect conditions. This will not be the case, due to several reasons like weather conditions, wear of infrastructures, number of devices connected to the base station as well as antennas location.

Using nperf website [56], it was withdrawn for each mobile operator, MEO, NOS and Vodafone, the map of 2G/3G/4G/4G+ cover in Portugal. Looking to the Figure 2.15:

- All three maps look similar in terms of 4G+/4G/3G/2G signal distribution.
- 4G+ covers the principals high populated density areas such as Oporto, Lisbon and south of Algarve.
- Close to the shore, there are better 4G cover signal when compared with the areas that are distanced.
- Even if the maps aren't completely loaded with mobile network results, it is observable that in the interior of the country, there are a lot of more area with 3G tests instead of 4G ones.

Additionally, in Appendix I [57], can be found the localization of the antennas of each mobile operator, although the maps weren't being updated since 2014. Despite that, it's concludable that the number of antennas in coastline or big cities are much bigger than the other location, corroborating the tests shown in Figure 2.15.

This problem may not be exclusively from Portugal territory, since it will happen to other countries that don't have all the territory covered with cell towers, thus leaving some areas with the deficiency of high-speed mobile networks cover. The solution may pass to subscribe different mobile operators allowing to use each other's towers when the coverage is not good enough with the other one. This could be expensive to cover a minimal area however, each case is a particular case and it should studied individually. Even applying a wired connection could be cheaper for a small area than subscribe another mobile operator.

In the end, this is a study that is beyond the scope of this work and should be done later, when it will be the deploying time. Note that these observations were only taking considering Portugal mainland, leaving outside of the study Azores' and Madeira's islands.

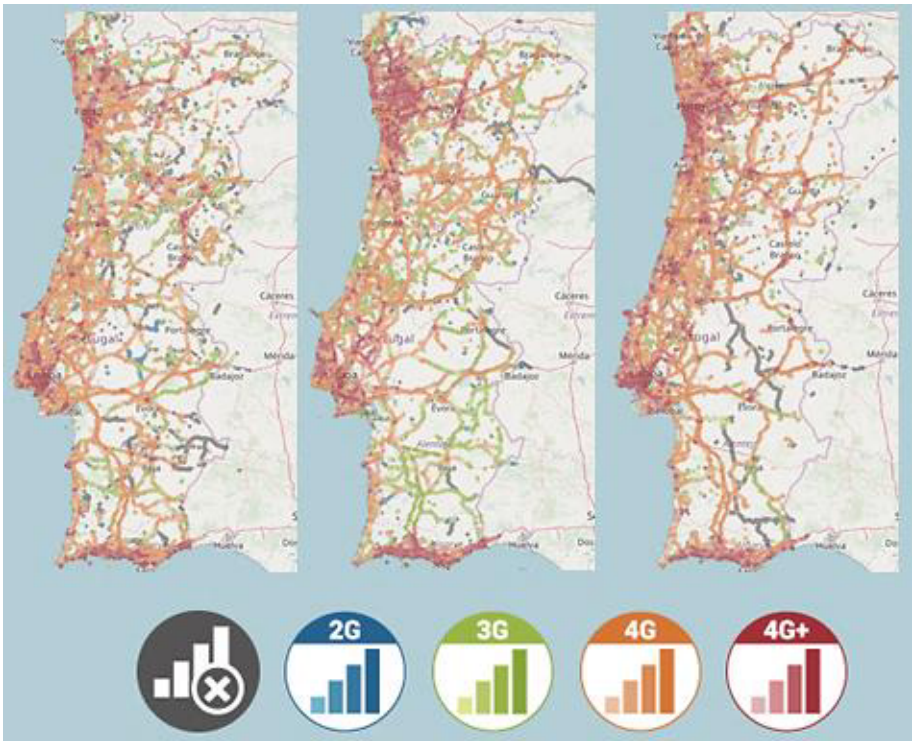


Figure 2.15: Portugal's coverage map to 2G/3G/4G according nperf [56]

Note:

- 1st map – Meo
- 2nd map – NOS
- 3rd map – Vodafone

2.14 Embedded Systems

Changing to the subject, it should be discussed where the work will be deployed. Embedded systems turn out to be a combination of a computer hardware and a piece of software. They differ from personal computers because embedded systems are designed exclusively to perform a particular task [58], making this option better in terms of physical size, processing capabilities to the required needs and, more important, usually cheaper.

Overall, embedded system functionalities are not changed during its use. In order to make a change, it is usually needed to reprogram all the system and there are hundreds, maybe thousands, of these examples in our day life, such as ATMs, house appliances, vehicles, mobiles, semaphores, among others.

Since they are so present nowadays and make engineers' tasks easier, some important characteristics will be remarked:

- Formerly, they were designed just to handle simple and light tasks, having limited functionalities such as switches, digital displays, LEDs, in order to indicate the status of the program, however, nowadays, they are capable to execute tasks with a certain level of complexity, e.g., ATMs [59].
- Theoretically, embedded systems had no user interface since all the data was already incorporated and no human interaction was necessary, e.g. semaphores, but the stronger capability of the new processors brought the possibility to interact with the system, through buttons, keyboards, graphic interface, etc.
- There are several different programming languages or operating systems that have been developed for embedded systems, for e.g., embedded Java, C/C++ or Python [60] and Ubuntu or Raspbian.

Looking for these characteristics, it provides a good platform to develop future work and implement IEC 61850 in real test condition since:

- With an open library, it is possible for the engineer or the student to be familiarized with standard concepts and application without needing, in reality, a real IDE, laboratory or substation.
- Due to its flexibility and mass production, they are cheaper and portable which make them suitable even for small offices.
- Possibility to write, execute and correct the application's code in order to optimize the behaviour before being executed in a real case.

3. Technical Description of IEC 61850

3.1 Introduction

In both previous chapters, it was pointed that the part that will require more attention in this thesis is the transportation however, the application is also important and shouldn't be neglected since there are some points that must be highlighted. Besides what was approached in 2.3, during this chapter are described other communication characteristics such as the retransmission process, the different type messages that are defined in IEC 61850, considerations regarding calculation time, both communication mechanisms that are used to transfer messages over a WAN as well as messages specifications that help to understand the practical work with more certainty and quality when unexpected problems appear.

Looking at the Table 2.1, there are three Technical Reports (TRs) that haven't been approached yet. Some parts of IEC 61850-8-1 and 9-2 were briefly described, however IEC 61850-90-1, 90-2 and 90-5 weren't. In part 90-1, it's described how the information exchange is made between substations, while in part 90-2 is addressed how the information is exchanged between substations and control or maintenance centers and, lastly, in part 90-5 is detailed how to send the synchrophasor data between PMUs, PDC's WAMPAC and control center application [5].

3.2 Retransmission Process

GOOSE and SV are protocols that don't include an acknowledgment procedure for their messages, lacking confirmation if the message was received. On the other side, there is a retransmission process [61] illustrated in Figure 3.1, that ensures every message is retransmitted with these rules: after the change occurs, the interval for retransmission is very short, a few milliseconds, being increased until the maximum value, which is in order of seconds. These parameters can be configured by the engineer, which is very useful because it will depend in the application case and the medium used to transmit data.

More important, with this method is possible:

- To guarantee that a lost message with not compromise the correct functionality of the system.

- In the case of a new device being connected after the first transmitted message, it is fitted to receive the same information as others, due to the content of retransmitted messages being the same information as the first one.
- Redundancy and security, because in case of intrusion, the data can be compared with the other devices in the network.

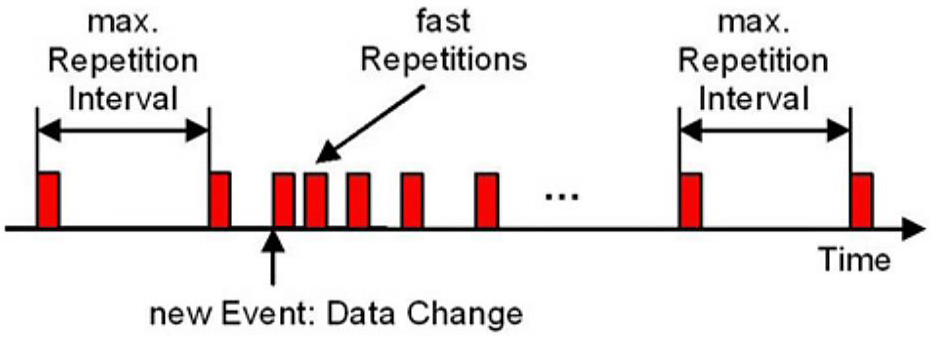


Figure 3.1: Retransmission Process [89]

3.3 Message Types and Performance Classes

In part 5 of IEC 61850 [62], are detailed not only the message types but also the time requirements and its functions. Inside a SAS it is necessary to transfer a panoply of messages, with different content, length, security and transfer time. These messages can be categorized in two independent performance classes: one related with control and protection where are included three performance classes, P1, P2 and P3; and the other one used for metering and power quality applications. In Table 3.1, it is described the characteristics about each message type.

3.4 Transmission Time

Bellow, in Figure 3.2, it's represented the Transfer Time, as variable t, between Physical Devices subdivided as t_a, t_b and t_c. The time t_a corresponds to the Publisher delay which "includes the data segmentation, message packing, message queueing and sent the message to the GOOSE processor". The time t_b corresponds to the network delay which includes "queueing delay of switch and network transmission". The time t_c is the time spend by Subscriber which includes "GOOSE message analysis, data connection and sending notifications to the GOOSE application terminal [63].

From Figure 3.2 can be withdrawn the following equation:

$$t_{transfer} = t_a + t_b + t_c \qquad 3.1$$

Decomposing the time spent at the Publisher communication processor, illustrated in Figure 3.3, it results in Equation 3.2:

$$t_{devicelatency} = t_y - t_x = t_c + t_{application} + t_a \quad 3.2$$

Using both Equations 3.1 and 3.2:

$$t_{transfer} = t_{devicelatency} - t_{application} + t_b \quad 3.3$$

At the end, it is obtained the Equation 3.3. However, unlikely the document where this information was withdrawn [63], in almost every case of the practical work, it will not dismiss the t_b. Detailing more, it will depend what kind of test will be performed:

- If a WAN is used to perform the tests, the t_b will be huge when compared with t_a and t_c, being both ignored.
- If a LAN is used to perform the tests, the t_b will be similar when compared with t_a and t_c.
- In both LAN and WAN, $t_{application}$ will not be consider, since it is removed in calculation, as later are illustrated in Figure 5.5 and Figure 5.9.

Notice that, it is considered the first bullet point when the tests are performed through the internet.

Type	Application	Performance Class	Time (ms)	Protocol	Example Application
1A	Fast messages – Trip	P1 P2/P3	≤10 ≤3	GOOSE	Circuit breaker commands (trip, close, reclose, start, stop, block); states; etc
1B	Fast messages – Others	P1 P2/P3	≤100 ≤20	GOOSE	
2	Medium Messages	-	≤100	Others	RMS values calculated from type 4 messages
3	Low Speed	-	≤500	Others	Alarm, configuration, non-electrical measurements, etc
4	Raw Data	P1 P2/P3	≤10 ≤3	SV	Digital representation of electrical measurements
5	File Transfer	-	≥1000	Others	File of data for recording, etc
6	Time Synchronization	-	Accuracy	Others	IED internal clocks synchronization
7	Control Commands	-	-	-	Based on type 3 message with additional secure procedure

Table 3.1: Message Types and Time Requirements [10] [20] [64] [65]

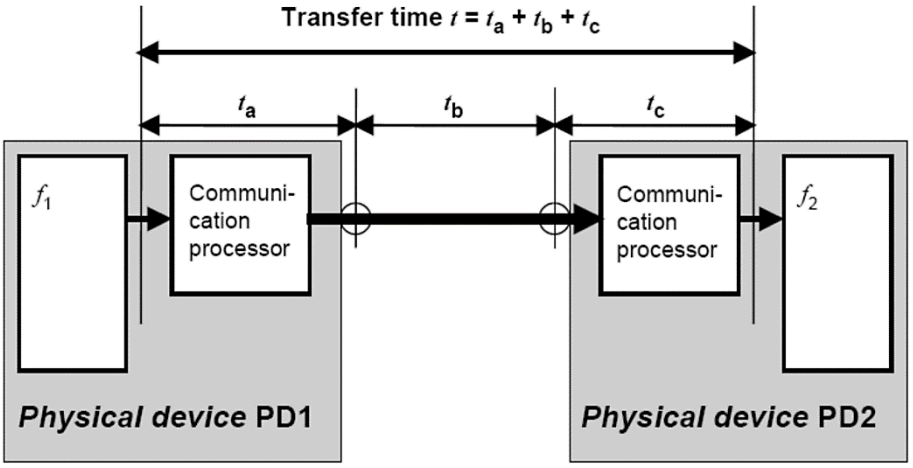


Figure 3.2: Transfer time between PD [24] [97]

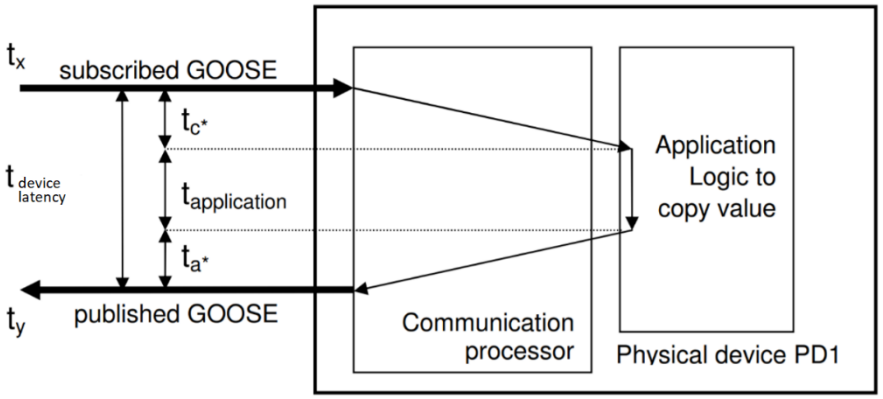


Figure 3.3: Time spent at Communication Processor [46] [97]

3.5 Communication Mechanism

As seen before, GOOSE and SV are communication protocols that work over Ethernet inside a substation. SV is responsible to transfer cyclic data, such as voltage/current values, while GOOSE is responsible to deliver event-based data. Nevertheless, just communicate inside a substation is not enough and it is needed to deliver this data over a wide area.

So, there are two valid options to transmit data to the outside of the substation [31] [5]:

- **Tunnelling**
  Using a VPN connection, GOOSE and SV will be tunneled through an high speed connection, leaving the device in an large, but same LAN.
- **Internet Protocol**
  Encapsulate the GOOSE and SV, giving them an IP and allowing them to be transferred over different networks.

Both options will be approached in the practical part of the document and its behaviour analysed under different transport technologies. GOOSE will be encapsulated into IP based protocol, UDP, already described in chapter 2. UDP can receive a broadcast, multicast or unicast address, allowing it to be sent to different devices. Also, it is important to refer that UDP is the protocol chosen in the standard, because it is more suitable for demanding time requirements.

GOOSE and SV encapsulated into UDP, will be called as Routable-GOOSE and Routable-Sampled-Values, commonly refer as R-GOOSE and R-SV.

3.6 R-GOOSE and R-SV Message Specifications

Remembering that GOOSE service is based on IEC 61850-8-1 and SV service is based on IEC 61850-9-2, it is good to notice that they remain unchanged for routable use. Considering this, in IEC 61850-90-5 is "introduced a new protocol in session layer for sending the GOOSE and SV over OSI connectionless transport" [21]. Henceforth, all described fields are present in the Technical Reports of IEC 61850. More concretly in parts: 90-1/2/5, unless said otherwise.

Remembering the OSI model, it's composed with seven layers and divided in two profiles, as shown in Figure 3.4, the Application and the Transport. As mentioned in TR IEC 61850-90-5, R-GOOSE and R-SV are implemented in Application profile and will be briefly described in next points.

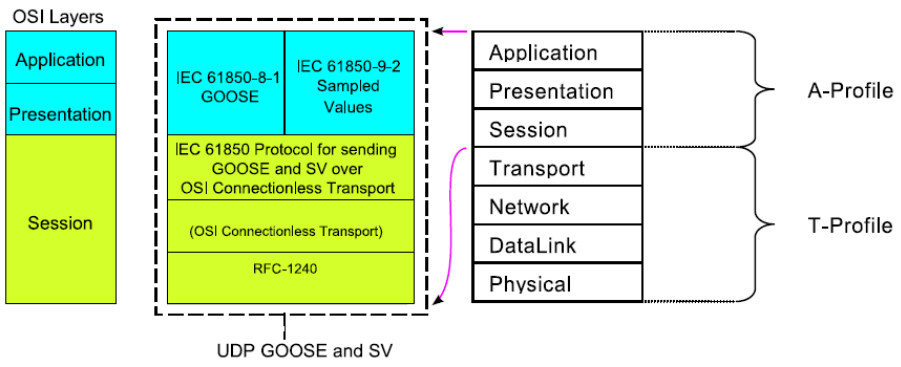


Figure 3.4: OSI Reference model and R-SV/R-GOOSE Application profiles [21]

3.6.1 Application Profile Specification

As a matter of fact, to understand and be aware of the field of each message is indeed important to detect unexpected errors or making "debug" in case of necessity. Therefore, about GOOSE, will be a short identification of each field: session header and user data as well as ASN.1 codification. SV will also be referred, but in an even short way since it will be barely focused in practical part. The following topic will be much

closer with the ones found in other works, since the description consists in a resume of what is found in IEC 61850-90-1, 90-2 and 90-5.

3.6.1.1 Session Header

In Figure 3.5 is shown how the session protocol is constructed as well as the bit and byte order are presented. Session Protocol Data Units (SPDUs) are the packets generated session layer which will be used in the transport layer as Transport Service Data Unit (TSDU).

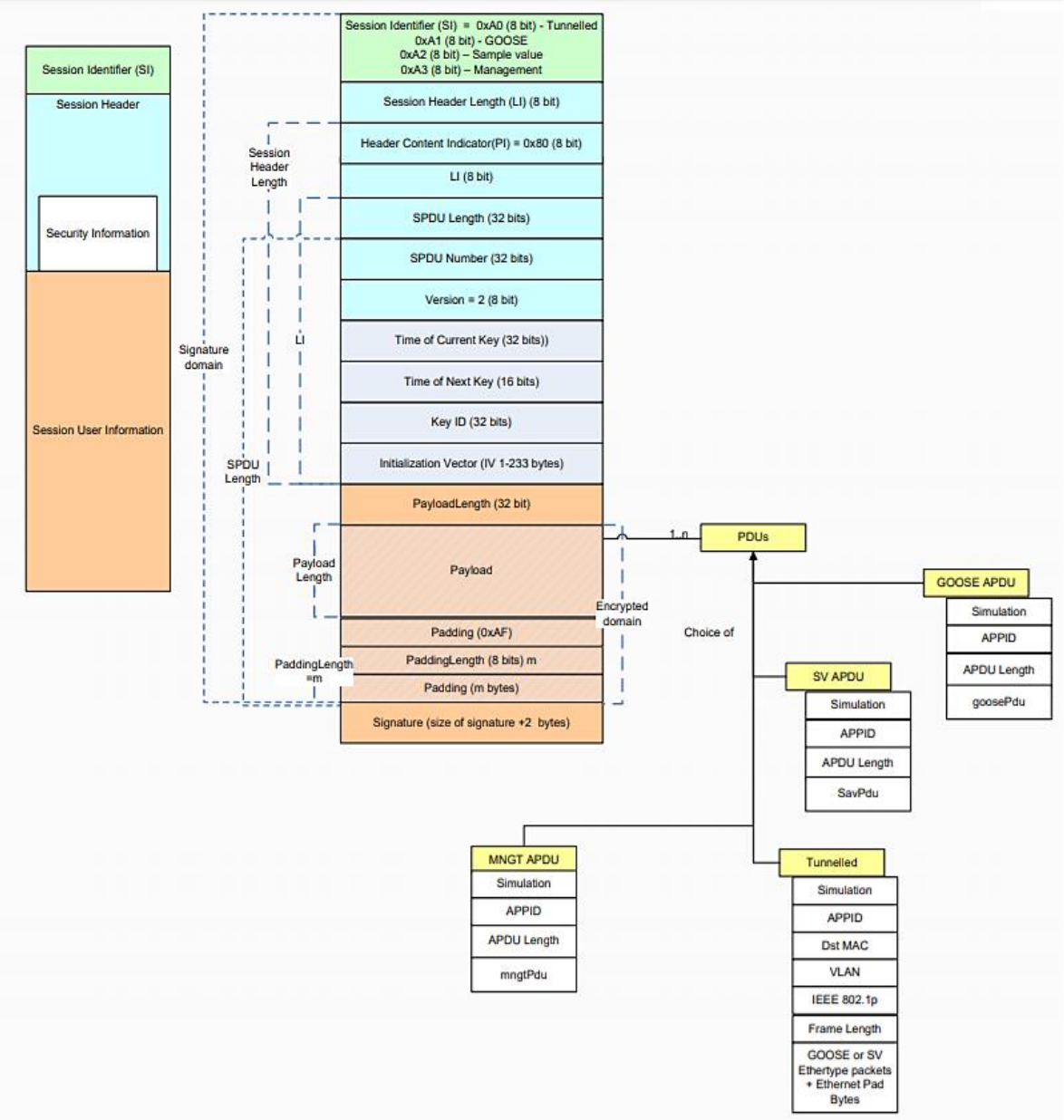


Figure 3.5: IEC 61850 90-5 Session Protocol [89]

The first parameter in SPDU Session Identifier (SI) and the possible four values that it can receive are detailed in Table 3.3. Next field is the Length (LI), composed with a single byte which represents only the length of all parameter fields for the session header, not including the user data session protocol. After SI and LI, the next field is Parameter Identifier (PI) with the hexadecimal 0x80, followed with the Parameter Value (PV) of the session header which should include the sequence of the following values:

Values	*Bytes*
SPDU Length	4
SPDU Number	4
Version	2
TimeofCurrentKey	4
TimetoNextKey	4
SecurityAlgorithms	2
Key ID	4

Table 3.2: Size of PV

SI Description	*SI Hex Value*
Tunneled GOOSE and SV Packets	0xA0
Non-tunneled GOOSE APDUs	0xA1
Non-tunneled APDUs	0xA2
Non-tunneled management APDUs	0xA3

Table 3.3: SI tags

3.6.1.2 User Data

The session user data is composed with two field Length and Payload. Payload is divided in 4 sub-fields starting with Payload Type and ending with APDU Length. A Signature could also be generated, although later in practical work will not be generated, so this field will have the value of 0 byte. The structure for R-GOOSE and R-SV messages is illustrated in Figure 3.6. There, is possible to find, on the left side, the Transport Profile yet at same time, on right side, the Application profile.

Now, depending if the message is GOOSE or SV the rest of APDU will be different. In case of GOOSE, is defined as goosePdu and it is described with more detail in TR of IEC 61850-8-1 while in case of SV, it is defined as savPdu and it is described in TR of IEC-61850-9-2 [5].

Values	*Bytes*
Length	4
Payload type	2
Simulation	1
APPID	2
APDU Length	2
Signature	3

Table 3.4: Sizes of parameters of user data

SI Description	*SI Hex Value*
Non-tunnelled GOOSE APDUs	0x81
Non-tunnelled SV APDUs	0x82
Tunnelled GOOSE and SV Value Packets	0x83
Non-tunnelled management APDUs	0x84

Table 3.5: User data payload tags

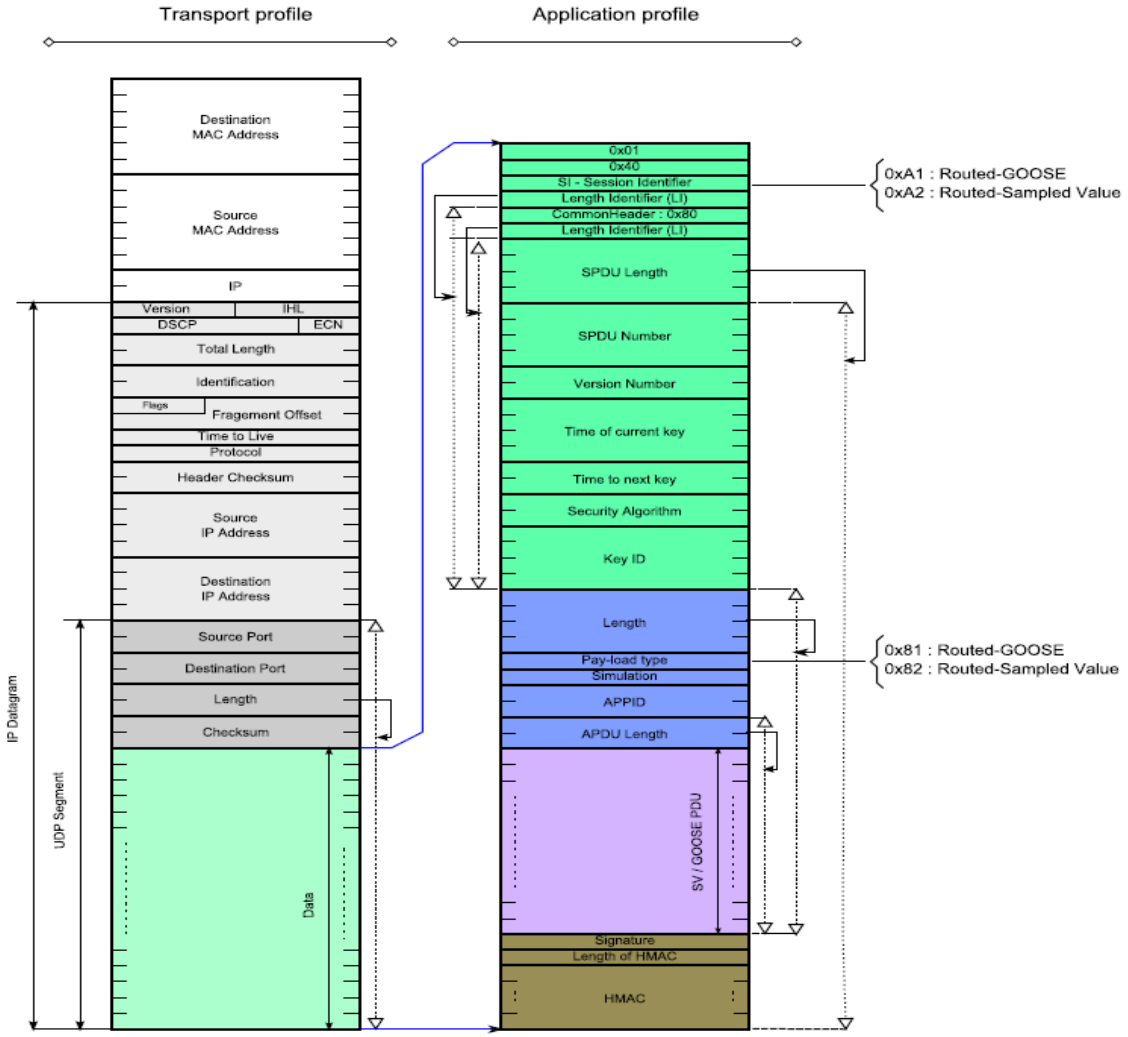


Figure 3.6: R-GOOSE and R-SV message specification [21]

3.6.2 GOOSE Protocol Data Unit Specification

3.6.2.1 ASN.1 Basic Encoding Rule

Abstract Syntax Notation One is a joint of international standards of ISO, IEC and ITU [66] that is used to define protocols by means of encoding rules. Considering this standard, each component of data respects the structure of: Tag, Length and Value, as represented in Figure 3.7.

Values	***Bytes***
Tag	1
Length	1
Value	-

Table 3.6: ASN.1 parameters size

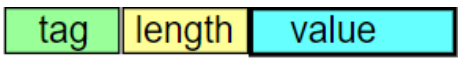


Figure 3.7: ASN.1 Structure

To codify GOOSE messages, it is used an ASN.1/BER adapted to MMS protocol and no the original ASN.1/BER.

3.6.2.2 GOOSE Protocol Data Unit

Jointly with the ASN.1 Basic Encoding Rule, the content inside of each field of GOOSE PDU will be presented. Additionally, in Figure 3.8, a capture of GOOSE message is displayed in order to enumerate the possible fields that could appear in goosePdu. It is also possible to observe that after "Reserved 2" with zeros, comes 61 which corresponds to the goosePdu tag, 0x61. In Appendix A could be found the detailed structure of goosePdu.

Values	*Tag*
GoCBRef	0x80
timeAllowedtoLive	0x81
datSet	0x82
goID	0x83
T	0x84
stNum	0x85

Table 3.7: goosePdu tags-1

Values	*Tag*
sqNum	0x86
Test	0x87
confRev	0x88
ndsCom	0x89
numDatEntries	0x8A
allData	0xAB

Table 3.8: goosePdu tags-2

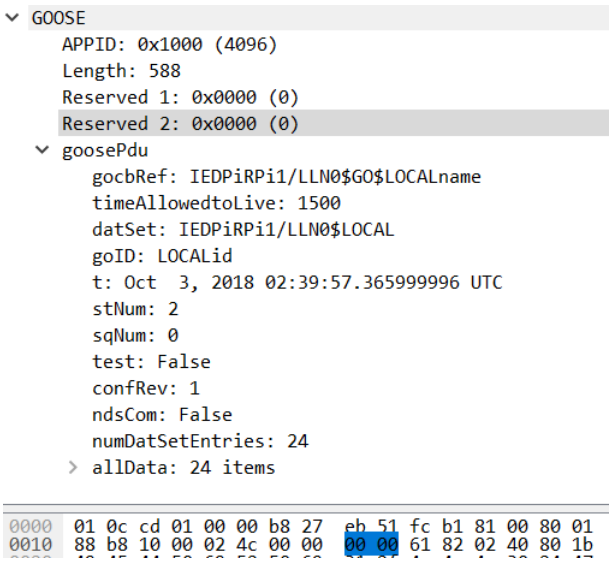


Figure 3.8: GOOSE Capture

3.6.2.3 goosePdu DataSet

GoosePdu DataSet is also codified by ASN.1 BER. The tags for each type of data are:

Type	*Tag*
Array	0x81
Structure	0x82
Boolean	0x83
Bit-String	0x84
Integer	0x85
Unsigned	0x86

Table 3.9: goosePdu DataSet tags-1

Type	*Tag*
Floating-Point	0x87
Octet-String	0x89
Visible String	0x8A
Timeofday	0x8C
BCD	0xAD
BooleanArray	0x8E

Table 3.10: goosePdu DataSet tags-2

3.6.3 Sampled Value Protocol Data Unit Specification

Sample Value Protocol Data Unit is referred as svPdu and is formed with ASN.1 Basic Encoding Rules. Considering that during the tests performed in practical work will not be transferred SV except during the tests with the examples provided by the library that was chosen to perform the practical part, it will be describe a quick overview about each field. Also, it is important to notice that contrary to goosePdu, the dataset of svPdu is not encoded, being transferred in its basic forms. In addition, in Appendix B can be found, like in goosePdu, the detailed structure of svPdu.

Type	*Tag*
noASDU	0x80
Sequence of ADSDUs	0xA2
svID	0x80
dataset	0x81
smpCnt	0x82

Table 3.11: svPdu tags-1

Type	*Tag*
ConfRev	0x83
refrTm	0x84
smpSynch	0x85
smpRate	0x86
sample	0x87

Table 3.12: svPdu tags-2

3.7 Conclusions

During this chapter, there are several things that are important to retain and, if needed, consulted them later. Even if the application is not that important for the future developed work, it is important to be aware how each message is structured, the tags and the content of each field, which could be an added value when it will be the time to analyse the content of the messages in Wireshark.

4. Network Architecture Design and Implementation

4.1 Introduction

In this chapter, it will be made an analysis of important data provided by EFACEC, a review about the hardware that will be used during the practical work, setting up the libIEC61850 and, more important, it will be present all the work performed required to accomplish the capture of the results with both communication mechanisms indicated in previous chapter as well as additional points that are needed to be aware of.

In the first place, it is necessary to be familiarized with the material and software available to develop the work. OMNET and NS3, two network simulators, are tested although they aren't exactly what it is required since they cannot perform tests in real conditions. It's known that environments can be a little tricky to test, due to the interference of other frequencies in the medium, other unexpected processes that are running in parallel in the devices as well as interferences due to the weather conditions, wear of the materials or unexpected variables in area. There are a lot of conditions to have in mind and nothing better than perform a real test. However, these interferences shouldn't have a great impact during the work developed but is something to be considered, since latency in these communications are critical. Even a 1 millisecond more can be too much, insofar as the fast messages need 3 to 5 milliseconds to be delivered which can represent a 20% latency at least.

As pointed before, it is necessary to find a good library that could fulfil the proposed requirements and, after a quick analysis, it is understandable that the best and most widely used solution is libIEC61850 by Michael Zillgith [67]. A few works are already made using this library, however there are still some strands to explore. According to the standard and as pointed before, there are two ways to deliver messages to outside of the substation: across tunnelling, using VPN or Routable-GOOSE/SV, using IP.

Concluding the process to choose the library, it is required to be aware of the hardware that will be used to perform the practical work. Here, it is presented the Raspberry Pis as well as all the routers and switch that will be used. Also, additional preparation is done regarding the libIEC61850 and the Raspberry Pis.

Further up, it comes the most important part of this chapter. In section 4.6, *Design of the Network*, is presented the standard topology to study over the practical part and the initial concepts of the Client/Server. Here, it is required to be aware of that Server is different than Publisher and that Client is different than Subscriber. Additionally, are show other related time characteristics that are common in both entities and it is worked in the application in order to generate failures.

Once the modifications performed in the application are accomplished, it's required to execute additional work to prepare our system to capture the messages correctly. A large part of these modifications aren't used in real cases and they are just performed in order to allow the easy capture of the messages.

Last but not least, it's displayed all the procedures to implement a VPN Server and connect its Clients. Here, since it is used VPN and the objective is to create a large LAN to include all the GOOSE Clients and Server, it is not needed to perform additional modifications in the GOOSE application due to it is originally designed to work in LANs.

4.2 Provided Data by EFACEC

4.2.1 Introduction

EFACEC Power Solutions is the largest Portuguese company that operates in the field of equipment for energy, engineering and mobility. EFACEC was founded in 1948, and it's present in more than 65 countries [68], having also some references in more than 90. Its business card is the infrastructures for fast charging electric vehicles since they are the worldwide leader in this matter.

EFACEC proposed the topic of this MSc thesis at the Institute of Telecommunications of Aveiro in order to understand what technologies are available to implement the standard IEC 61850 and its associated costs. Another suggestion proposed was also to understand how well prepared Portugal and other countries are in terms of mobile networks and if they achieve the latency and throughput necessary to fulfil such demanding and hard time requirements like IEC 61850.

4.2.2 Proposal and Provided Data

During the proposal and first meetings, it was discussed what were the options to implement IEC61850, more concretely in Portugal and what infrastructures were available. As discussed, deploying a copper or optical fibre cable was not an option as result the size of electrical network. According to REN on page 3 of this document [69], in 2017, the length of electric line in Portugal was about 8907 km. So, as result, this would lead to an enormous cost, if the option was to equip the distribution with cables for communications.

Even before the survey of the possible technologies that could be implemented, suggestion was left on the table, try to use the infrastructures that were already built. In this case, see if the mobile operators are suitable to answer the time requirements of the IEC 61850.

EFACEC also provided us some tables in order to evaluate the distances between the reclosers and substations and how many substations and reclosers would be in one working area. In the following three tables, there are:

- SE – Substation
- R – Recloser

Example 1:

- Table 4.1
- 3 outputs of 3 substations
- Area occupied by energy network: ≈ 8.8 km^2
- Density: ≈ 0.45 reclosers/km

	SE 1	R 1	R 2	SE 2	R 3	R 4	SE 3
SE 1		1,1	2,48	6,54	4,31	4,73	7,23
R 1			1,63	5,77	3,41	3,77	6,47
R 2				4,14	1,83	2,28	4,84
SE 2					2,47	2,45	0,72
R 3						0,62	3,17
R 4							3,11
SE 3							

Table 4.1:Distances between the different devices in Example 1 measured in kilometres

Example 2:

- Table 4.2
- 3 outputs of 2 substations
- Area occupied by energy network: ≈ 411 km^2
- Density: ≈ 0.02 reclosers/km^2

	SE 1	R 1	R 2	R 3	R 4	R 5	R 6	SE 2	R 7	R 8	R 9
SE 1		11,16	11,38	22,91	23,8	24,99	26,19	27,95	24,22	20,98	20,53
R 1			0,61	12,24	13,06	13,86	15,11	17,57	13,06	10,65	11,27
R 2				12,3	13,09	13,72	14,83	17,13	12,86	10,18	10,71
R 3					0,93	4,36	8,21	13,72	5,43	10,71	13,68
R 4						3,82	7,77	13,38	5,12	10,86	13,88
R 5							3,97	9,65	1,75	8,42	11,48
R 6								5,71	2,89	6,65	9,36
SE 2									8,28	7	7,98
R 7										6,71	9,75
R 8											3,07
R 9											

Table 4.2: Distances between the different devices in Example 2 measured in kilometres

Example 3:

- Table 4.3
- 2 outputs of 2 substations;
- Area occupied by energy network: ≈ 26 km^2.
- Density: ≈ 0.31 reclosers/ km^2

	SE 1	R 1	R 2	R 3	R 4	R 5	R 6	R 7	R 8	SE 2
SE 1		3,08	4,51	4,27	3,91	3,57	5,73	6,2	5,56	8,38
R 1			1,64	1,3	1,07	3,31	2,67	3,16	2,67	5,5
R 2				0,42	1,76	4,77	1,64	1,83	1,06	3,89
R 3					1,35	4,36	1,61	1,95	1,37	4,2
R 4						3,09	2	2,63	2,53	5,18
R 5							4,93	5,61	5,62	8,22
R 6								0,69	1,33	3,3
R 7									1,09	2,61
R 8										2,84
SE 2										

Table 4.3: Distances between the different devices in Example 3 measured in kilometres

4.2.3 Conclusions

Overall, these tables from EFACEC are representing the extreme cases and conditions in terms of distance and number of devices that should be able to communicate between them. However, this data is provided for a small territory and with low population density, as it is Portugal.

Additionally, there is some relevant information that can be withdrawn from these tables, starting with: the distance and the method of implementation to put on the devices in communication with each other, i.e., through VPN or through IP; if there are devices that may only need to communicate with a specific one; if there are other technologies can be applied here because, looking to Table 4.3, there are devices that are distanced between each other roughly 400 meters (making Wi-Fi an option). Considering this data provided by EFACEC along with the background in chapter two and three, there is an idea of which technologies should be used to perform the tests, which including Wi-Fi, Optical Fibre, ADSL and 4G-LTE.

Another important point to be noticed is the number of devices that will be connected and will send traffic between each others. With this in mind, using VPN or IP approach has its advantages and disadvantages and as tunnelling concerns, having too much devices in the same network could lead to package dropping, since the fluid of traffic could be too high to handle. In this instance, the number of devices isn't considered too much, which could be a favourable point for the tunnelling in performed tests.

4.3 LibIEC61850

4.3.1 Overview

As presented before, "IEC61850 is an international standard for communication of Substation Automation Systems and management of Distributed Energy Resources*"* [67]. So, it is necessary to have a library that could translates the standard into code line and thus, the choice of libIEC61850 by Michael Zillgith. This project provides the possibility to implement a complete server and client application, accomplishes IEC61850 MMS/GOOSE/SV communication protocols and it's an open source library, available under the GPLv3 license. It is implemented in C (according to the C99 standard) delivering this way a maximum portability across platforms. It's developed to run in embedded systems as well as in PC's, either running in Linux, Windows or MacOS.

Additionally, the library includes a group of examples where it is possible to be familiarized and get a quick understanding of how the library works. Later, in this chapter the examples will be covered with more detail.

List of futures that the libIEC61850 covers [70]:

- MMS client/server, GOOSE (IEC 61850-8-1)
- Sampled Values (SV - IEC 61850-9-2)
- Support for buffered and unbuffered reports
- Online report control block configuration
- Data access service (get data, set data)
- online data model discovery and browsing
- all data set services (get values, set values, browse)
- dynamic data set services (create and delete)
- log service; flexible API to connect custom data bases; comes with sqlite implementation
- MMS file services (browse, get file, set file, delete/rename file) ** required to download COMTRADE files
- Setting group handling
- GOOSE and SV control block handling
- TLS support
- C and C#/.NET API

Therefore, this library is quite complete but there is one important thing that wasn't developed yet, R-GOOSE and R-SV, and for further practical analysis it is necessary to implement it.

4.3.2 Related Work using libIEC61850

In the begin, it is made a research exploring the works performed with the libIEC61850. Looking for Msc/PhD Thesis and published papers, it is noticed that most of the work was developed inside of the same LAN, leaving without analysis the possible messages to outside of the substation. Still, it's important to analyse inside the same LAN but here, the work is quite abundant.

In terms of practical work, the one that is better structured is the MSc thesis, "Simulator of Scada Protocols", made by Radim Studený [71]. His work is a better and more complete development of examples, with a complete Client/Server configuration in terms of application. Then, along with the examples of the libIEC61850, it is extremely helpful to generate traffic and analyse it as well as understand the path that messages took during the tests.

As referred before, there are a little bit of scarcity in terms of researches for messages to outside of the substation using R-GOOSE or R-SV. A good and interesting work that was also analysed was: "Interpreting and implementing IEC61850-90-5 Routed-Sampled Values and Routed GOOSE protocols", developed in KTH Royal Institute of Technology, in Stockholm, Sweden and Catalonia Institute for Energy Research, in Barcelona, Spain [21]. As matter of fact, they approached this gap and developed, as they call it, the Khorjin Library. According to them, this library allows to carry R-GOOSE and R-SV messages and to be capable to act as receiver and parser. Unfortunately, it wasn't possible to use their code in order to perform different tests and this is something that isn't implemented in libIEC61850, as pointed before. Therefore, developing a Gateway Sender inside the Server and a Gateway Receiver to receive traffic to outside of the substation is required and will be a topic that will be further approached with more detail.

4.4 First Steps with LibIEC61850

4.4.1 Examples

The first step with the libIEC61850 is to understand how the examples work. Inside the examples folder, there are several examples such as "goose_publisher", "goose_subscriber", "sv_publisher", "sv_subscriber", "server_example_goose", "iec61850_client_example1", among others.

Focus on these examples, it is necessary to compile them, for later, run and analyse a simple publisher/subscriber topology combined with IEC 61850 GOOSE/SV messages. Additional information about setting up the examples can be found in libIEC61850 website as well as the official repository of the library in GitHub [70]. Nevertheless, not every command can be found there, and for that reason, it will be detailed in Appendix O what commands should be executed to perform the analysis of the examples.

As soon as the procedure described in Appendix O is performed, the code is running in a terminal window and there, it is possible to observe some displayed text resulting of sent messages. More important is to execute the capture of the messages that are flowing through the LAN. For that happens, it's possible to use tcpdump command, which will be used later during the work, and save the capture to .pcap file. Using the Wireshark, that uses an interactive GUI, is also an option however, it requires to use processing power, which may slow down the application. Figure 4.1 and Figure 4.2 represent two captures resultant of running examples.

As seen, in Appendix O, the procedure presented regards to GOOSE messages, but for other examples that involves SV, the procedure will be very similar and intuitive after understanding the GOOSE one.

GOOSE

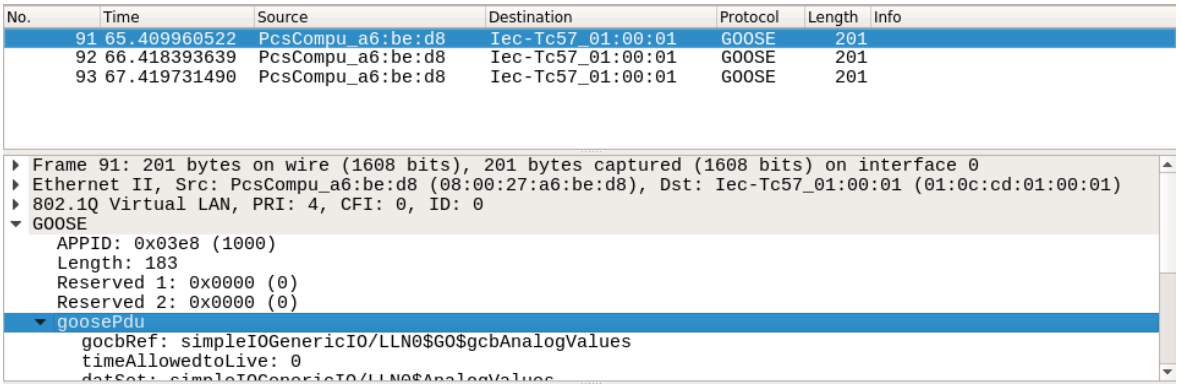


Figure 4.1: GOOSE capture resulted of running examples

Sampled-Values

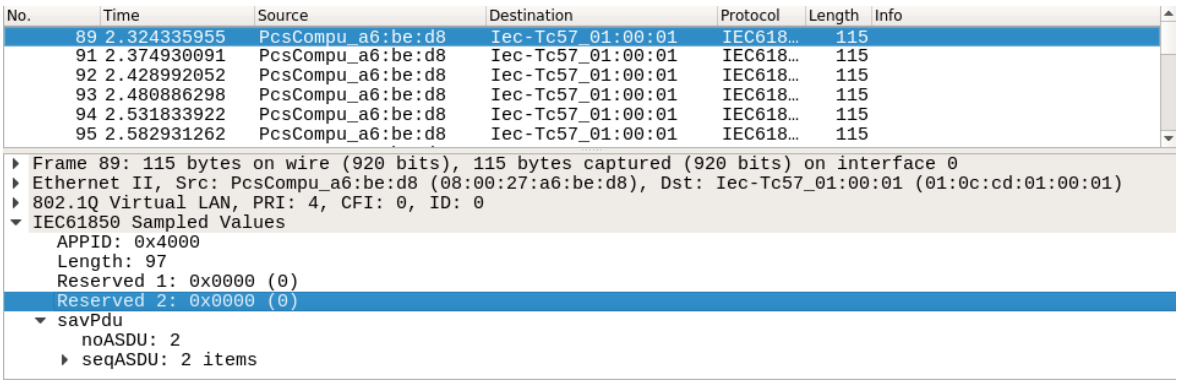


Figure 4.2: SV capture resulted of running examples

4.4.2 Practical Application

Once executed the examples, it is noticed that this library contemplates a big potential to develop a complete work and analyse the transportation part properly almost as it would be used a real IED. However, the examples presented in previous section, are not good enough since each entity just contains a publisher or a subscriber,

contrary to what was needed (a complete Server and Client application). So, a research is made regarding works performed using this library and there is one work that stood out [71]. Even written in Czech and after translating it into English it is possible to understand how the practical part is performed and what particularities it is necessary to be aware of. Additionally, even with a good pre-developed application, it is still necessary to proceed with several adjustments and added some code that will allow to capture messages on both sides in a manner that could be possible to measure the Round Trip Time (RTT). Later, using this RTT, it is calculated the One Way Trip Time (OWTT). In next section 4.6, it will be explained these modifications and insertions that will end up with a symbiose between application and transportation.

4.5 Platforms

4.5.1 Platforms to Implement the Library

During the chapter 2, it was approached the embedded systems and its advantages and limitations regarding to the implementation. Aware of its characteristics, it is looked for embedded systems that are suitable to implement the work in terms of availability, background regarding the chosen library and its capabilities to perform the work. The first one is the BeagleBone where are examples ready to interact with its hardware while in the second one, there is the Raspberry Pi, that is chosen to perform the work insofar, as seen before, a more complete application was already developed.

4.5.1.1 BeagleBoard

BeagleBoard is a low-power open source computer produced by Texas Instruments in association Digi-Key and Newark element14. Later, some BeagleBoard family computers were launched, like BeagleBone and BeagleBone Black, displayed in Figure 4.3 [72].

Looking to BeagleBone is important because, the libIEC61850 have some demos where it is possible to interact with the hardware and the software. The purpose of these demos is to simulate failures by pressing a button and generate GOOSE messages. Additionally, they are taken in consideration to develop code of other researches, being upgraded and modified to deliver a better and complex real simulation of IEDs. Further, a quick look and analysis will be made regarding these demos and examples.

4.5.1.2 Raspberry Pi

Also displayed in Figure 4.3, Raspberry Pi is a small single board computer developed in UK by the Raspberry Pi Foundation [73]. Also, like BeagleBone, it is a low cost computer, usually used in schools and learning process and ideal to use in embedded systems, which by nature does not require a lot of processing power.

Again, considering the work develop in MSc Thesis "Simulator of Scada Protocols" implemented in Raspberry Pis, it is a good starting point to the work that will be developed. Later, it will be discussed the similarities and differences between both works and withdrawn the appropriated conclusions.

Figure 4.3: On the left, BeagleBone Black and on the right, Raspberry Pi

4.5.2 Setting up the Raspberry Pis

4.5.2.1 Operating System

As soon as we are familiarized with all the hardware that will be used during the performed tests, it is also necessary to setting them up before proceeding with them.

Initially, it is downloaded the image *2018-04-18-raspbian-stretch* from official Raspberry website [79]. This image includes a GUI which will be quite helpful to set everything up until the final tests. This was a pondered choice, since later it is possible to boot the Raspberry Pis without the GUI, giving a better flexibility and facility to fight against unexpected errors that may occur.

Once the download finish, it is used a software called Rufus to burn the image into Micro-SD Cards. Since this image has a GUI, it is mandatory to use a Micro-SD card with, at least, 8GB of memory to install all the required software and save all the capture and additional needed files.

4.5.2.2 VNC and SSH

On the first boot it is necessary to connect the Raspberry Pi through HDMI cable to a monitor to get a visual contact with the OS. A mouse and keyboard are also required to have, since it's needed to execute a couple tasks and commands before the Raspberry Pi being controlled remotely.

These procedures and commands are detail in Appendix M and in the next sections, it will be considered as performed.

4.5.2.3 Compiling Libraries

LibIEC61850

Now that the Raspberry Pi is ready to work remotely, it's safe to start to work with the chosen library. First, it's necessary to compile it, so that later it may be possible to work in main program, using its headers. Here, like in the previous section, there is a description in Appendix N, where it describes how to compile the library inside the Raspberry Pi.

Cross-Compiling using NetBeans

On the other hand, programming and building programs could be a tough task given that, the Raspberry Pi isn't a very powerful device. So, a good and comfortable option is programming on IDE in a computer and later, make the compilation to the desirable Raspberry through LAN IP.

Considering that SSH is enabled in a few steps back, it is just needed to make a few additional configurations in NetBeans in order to make a successful cross-compiling. In Appendix J, there is described the procedure as well as three figures to complement it.

4.5.3 Routers

4.5.3.1 Altice FiberGateway GR241AG

The Clients that will contract an Optical Fibre connection from ISP Meo, will receive the Altice FiberGateway Gr241AG, displayed in Figure 4.5, the new router developed by Altice Labs. It's an Optical Network Terminal (ONT) and router at the same time for the termination of Passive Optical Network (PON). Basically, it's an all in one compared with the older Meo devices where it was necessary to have ONT and a router [74]. It includes 4 RJ45 ethernet ports that supports at least 100Mbps, which is excellent to connect all the Raspberry PIs and my computer. In all the tests performed using optical fibre, FiberGateway was the router used and the bandwidth hired from ISP was 100Mbps.

4.5.3.2 LinkSys WRT1200AC

Linksys WRT1200AC is the router used to perform the tests that are executed in same LAN and used to create a VPN Server. It is a dual-band Wi-Fi Router however, the 5Ghz band will not be used since the Raspberry Pis

network cards don't support it. More important, includes 4 RJ45 Gigabit ethernet ports which is great to connect all of them through cable without needing an extra switch, reducing this way delaying times. It's displayed in Figure 4.4 and considering that is a router, it has a powerful 1.3Ghz dual-core ARM to manage different and demanding situations [75].

4.5.3.3 Huawei E5172

Displayed in Figure 4.5, Huawei E5172 is the router used to perform the 4G-LTE tests. It is an LTE CAT4 wireless router that can support download speeds up to 150Mbps. Also supports LTE FDD/TDD/UMTS/GSM. For FDD, the maximum download speed is 150Mbps and upload is 50Mbps while for TDD the maximum download speed is 112Mbps and upload 10Mbps.

There are some editions of this router and the one that was used the Huawei E5172As-22 and his characteristics are [76]:

- 4G-LTE: FDD LTE 800/900/1800/2100/2600 MHz, TDD 2600 MHz
- 3G HSDPA/HSUPA/UMTS/WCDMA 900/2100Mhz
- 2G GSM/GPRS/EDGE 850/900/1800/1900MHz

In terms of relevant interfaces, there is one slot to insert a full-size SIM card and one RJ-45 Fast ethernet port.

4.5.3.4 ADSL Home Routers

There are two different routers used to perform the tests. Netgear GC3100 [77] and ADB P.DG A1000G [78], both displayed in Figure 4.5 and Figure 4.4 respectively. The Netgear is just used to perform one ADSL to ADSL test while the ADB is used also to perform additional tests involving ADSL to 4G. In both cases, the routers are used with an ethernet cable, using simple configurations like port forwarding. Again, in both cases the speed hired from the ISP is 12Mbps.

4.5.3.5 Switch TP-Link TL-SG108E

TP-Link TL-SG108E, in Figure 4.4 , is a small and compact switch with 8 RJ45 Gigabit ethernet ports, allowing this way to connect all the devices through ethernet when the router used lacks in terms of ethernet ports, like the Huawei E5172 or the ADB P.DG A1000G routers.

Figure 4.4: From left to right: Linksys WRT1200AC, Switch TP-Link TL-SG108E, P.DG A1000AG

Figure 4.5: From left to right: Altice FibreGateway GR241AG, Netgear GC3100, Huawei E5172

4.6 Design of the Network

4.6.1 Introduction

Considering that a large number of works from other thesis and papers were performed running the GOOSE or SV examples, it is essential to look to initial questions and what could be developed in order to answer them. The configuration in Figure 4.6 is the one run in the examples, allowing GOOSE messages to travel between the publisher (acting as a Server) and the subscriber (acting as a Client).

In the first place, it will be explained what code is developed, as well as additions and modifications that are made to the Server and the Client compared with previous works. Here, the most important part is to explain what is made in order to generate more traffic, for later be analysed.

Secondly, and as mentioned before, there are IEDs that will be outside of substation and it is necessary to send the GOOSE message to them, existing two communication mechanisms to accomplish this objective: it could be done through tunnelling, using VPN or through IP, using R-GOOSE. Using VPN means that a big LAN network will exist while using the R-GOOSE, means that the GOOSE need to be packed and sent over WAN using a layer 3 protocol, which will UDP, in this case. Either way, these devices outside of substation need to receive the messages and will be a considerable distance between them. According to EFACEC data, in page 43, they will be distanced by hundreds of meters to dozens of kilometres, which means that, according to the research that was made in chapter two, Optical Fibre, ADSL, 4G-LTE and Wi-Fi could be valid options.

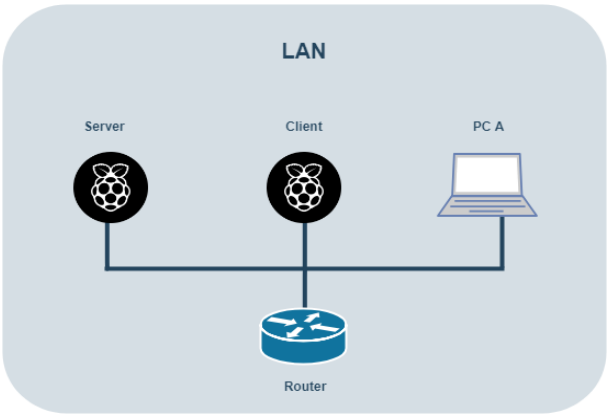


Figure 4.6: Configuration Client/Server

4.6.2 GOOSE Server and Client

4.6.2.1 Introduction

Starting with Server and Client, they are two entities where GOOSE messages in LAN will be generated and the traffic will flow between them. It is important to understand what is the role of each one, to understand how and why the messages are created and which ones belong to each other. Thus, Server is the entity that contain the information about every client while the Client just have the information about itself. In this work, one Server and Client are developed. However, if more clients would be added, it wouldn't change anything else instead of the generating more traffic and lead to bigger messages produced by the Server, since it is the one that contains the information about all other clients.

Another thing important to remark and already implicit in the previous sentences is that both Server and Client have to send its data over the LAN as well as to be listening. Hence, this implies that both have publishing and listening functions despite what happens with the examples, where the Server is just sending data with publisher function and the Client is just listening with listener function.

4.6.2.2 Common characteristics

Simultaneously, there are some characteristics that matches both Server and Client and it is important to be aware of them since they will appear more than once and is not desirable to be repetitive. Thereby, in terms of communication, it is truly important to understand how, when and why the GOOSE messages are generated. Different from the example of *goose_publisher.c* where it just generates a GOOSE message at every each second for three seconds, here in GOOSE Server and GOOSE Client, are used the configurations in libIEC61850 that are present in *stack_config.h*. Looking for the Figure 4.7, there are three *"define"* that helps to understand how the GOOSE flows in further captures:

- In line 104, *CONFIG_GOOSE_STABLE_STATE_TRANSMISSION_INTERVAL* is equal to 5000, which represents 5 seconds. This means, as described in code as commentary, if there are no changes made in database of the entity, Server or Client, for 5 seconds, it will send a GOOSE message with its unchanged data, also known as "heartbeat". These retransmissions are used mainly to keep the connection alive, as well security redundancy.
- On the other hand, in line 107, *CONFIG_GOOSE_EVENT_RETRANSMISSION_INTERVAL* is equal to 500 which means that every 500 milliseconds, a new GOOSE message will be sent if a new event occurs. This value is lower than the one presented in *goose_publisher.c* example. However, as shown in chapter 3, (Figure 3.1) the messages have to be retransmitted in order to prevent delivering failures which means that the faster they will be retransmitted, the faster they will be received.
- Last but not least, in line 111, there is the number of the transmissions left. Number 2 means that, beside the first message and the first retransmission, there are two more to be accomplished. In total, four GOOSE messages will be sent, one derived from change of the state, another from the default transmission and two more that are defined here. In Figure 4.8, there is the part of the same capture that will appear later however, it's here to elucidate the retransmission process, having all the GOOSE messages separated by 500 ms. Another thing important to notice, is that the last message of retransmission have a bigger "timeToLive" of 15000 ms, while the previous ones have just 1500 ms. The messages 481 and 482 do not belong to the retransmission of the previous change of state but to the next one and that's why they are sent a few seconds later.

```
104  /* The GOOSE retransmission interval in ms for the stable condition - i.e. no monitored value changed */
105  #define CONFIG_GOOSE_STABLE_STATE_TRANSMISSION_INTERVAL 5000
106
107  /* The GOOSE retransmission interval in ms in the case an event happens. */
108  #define CONFIG_GOOSE_EVENT_RETRANSMISSION_INTERVAL 500
109
110  /* The number of GOOSE retransmissions after an event */
111  #define CONFIG_GOOSE_EVENT_RETRANSMISSION_COUNT 2
```

Figure 4.7: Defined values that re-transmission concerns

4.6.2.3 GOOSE Server

Here, in GOOSE Server, the common characteristics displayed in previous section, are also present in this entity, although there are a few particular points that have to be noticed.

Particular communication characteristics and behaviour of GOOSE Server:

- GOOSE Server is implemented in an independent Raspberry Pi.
- It's an entity that contains all the information about itself and its clients.
- Previous transmission and retransmission methods are applied here.

Simplifying, as soon as a new message with different data than the one that is store in it database arrives, it will trigger the GOOSE transmission and retransmission process as described above through broadcast. Otherwise, it just transmits the "heartbeat" at every 5 seconds, due to the value present in *CONFIG_GOOSE_STABLE_STATE_TRANSMISSION_INTERVAL*.

Additionally, the Server is an entity that contains the information about its Clients, so it's natural that its GOOSE messages will be larger than the Clients one. As shown in Figure 4.8, GOOSE Server messages has 606 bytes while the Client ones have 194 bytes.

Besides the communication part, here could be implemented a panoply of new functions that could be used to alert and interact with the control centre, to take actions to protect and control other IEDs as well as shut down the system. Of course, all these configurations will depend on the engineer that will program the device. For example, during this work, several functions to print data in command line are placed in order to track some problems that may occur during the execution.

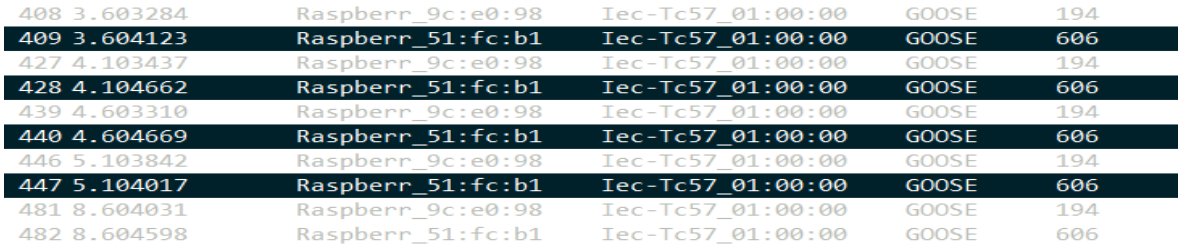


Figure 4.8: Example of a GOOSE retransmission

4.6.2.4 GOOSE Client

The GOOSE Client is very similar to GOOSE Server in terms of communication. Its differences rest that the client just stores information about itself and, as a consequence, its messages will be smaller since they carry less data.

Communication characteristics and behaviour of GOOSE Client:

- GOOSE Client is implemented in an independent Raspberry Pi.

- It's an entity that contains all the information about itself.
- Besides the Client, the Server is the one that store the same data as the Client, turning the Server the one that the Client only will interact with. Considering this, to trigger a new GOOSE message in Client, there are two possibilities: a GOOSE message received by the Server or a new data collected by the sensors. In both cases, the data have to be different than the one that is already stored. Either way, in both cases, it is applied the retransmission mechanism.
- Naturally, if no other changes occur, at every 5 seconds, the "heartbeat" is sent as described above.

Such as the Server, the Client have different functions to work with. Clients usually relies in functions that are associated to its sensors.

4.6.2.5 Preparing Application and Provoking Failures

As described in 4.6.2.2, an effective way to provoke new GOOSE messages, is to receive a value different that the one that is already store.

In Radim's work [71] it is given more attention to the application part, where the failures are provoked using real buttons configured in I/Os of the Raspberry Pi. This isn't practical to generate GOOSE messages since *"I don't press buttons at the same speed all the time"* or it's hard to generate more messages in milliseconds if it's necessary. So, the conclusion is to edit his work and ignore all the code that would need hardware to interact, e.g., pins configurations. Then, a simple way to explain how to provoke more messages is, in the Client, where it is desirable to provoke the *"Overvoltage failure"*, every time that a cycle of *"for cycle"* happens, update the value in *IedServer_updateBooleanAttributeValue*, for example, with the Boolean value different than the one that was stored in previous cycle. This way, every new cycle, the stored value will be different leading to a new GOOSE message being broadcast in the LAN. This explanation is related to the code that is presented in Appendix H.

4.6.2.6 Modifications in Server and Client to Capture GOOSE over Wi-Fi

Before going to routed messages and since the topic Client/Server was opened just right before, something that is also interesting to explore is to modify the connection between IEDs to an wireless connection, in the same LAN. An approach like this has a good motivation to be explored, because even if the distance between the devices inside the same substation is small, it may entail an extra cost. So, in terms of engineering perspective, saving costs is always one of the highest priorities when the work has to be deployed, making this strand an interesting one to be explored.

The modifications are performed in the Client since the Server is usually close the router. However, these modifications can be performed in any entity, being them Server or Client. Looking again to the code in

Appendix P, it is necessary to modify the interface from eth0, that belong to the wired connection, to the wlan0, that matches to the wireless one.

Additionally, it's modified in both entities, Server and Client, the traditional destination MAC Address in order to receive the GOOSE over Wi-Fi. These modifications take place in *static_model.c* file and it is necessary to change the destination MAC Address from *"01-0C-CD-01-00-00"* to the broadcast one, *"FF-FF-FF-FF-FF-FF"*. Notice that, the standard IEC 61850 have destination MAC Address reserved, from 01-0C-CD-01-00-00 to 01-0C-CD-01-01-FF for GOOSE [80], so making this change not totally correct in terms of application. Either way, this modification is effective, since the GOOSE packets are also sent to the wireless network cards, allowing this way to perform the tests, using the Wi-Fi.

4.6.3 R-GOOSE via UDP/IP

Here, a different kind of work will be performed. To allow new messages to be sent over WAN, it is required to add a module to the Server that will act as Gateway Sender and create an entity that will act as Gateway Receiver. This both entity and module, will allow that UDP packets will be sent over the WAN. Additionally, these UDP packets, as illustrated in Figure 3.6, had to contain the GOOSE messages encapsulated in its payload. Hence, these UDP packets will be used to measure RTT and calculate OWTT. In this case, OWTT is more important than RTT, because GOOSE messages are one directional messages and, as consequence, it only interests the time that one message takes from the sender to the receiver.

4.6.3.1 GOOSE Publisher/Receiver

GOOSE Publisher and GOOSE Receiver are functions that are used to send and receive GOOSE messages. As illustrated in Figure 4.9, they belong to Hardware/Platform Abstraction Layer. Both functions are also located in libiec61850-1.2/src/hal. In brief, GOOSE Receiver, is used to receive GOOSE packets and parse them. Additionally, in this case, they will not suffer any modifications because the new UDP messages will have other destiny than GOOSE Receiver. Further, it will be explained.

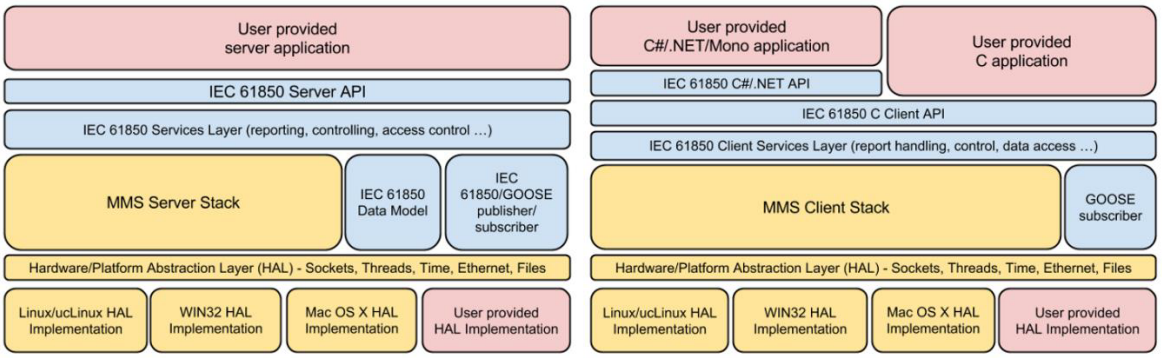


Figure 4.9: Server and Client API

4.6.3.2 Gateway Sender

Gateway Sender will be the name given to the modifications performed in *goose_publisher.c*, because it will take a different behaviour when compared with the library. It will send the UDP messages with encapsulated GOOSE from the Gateway Sender to the Gateway Receiver. With the objective of these modifications to be successful, it is necessary to analyse GOOSE Publisher and discover where the GOOSE messages will be send, for further proceed with the modifications.

Function *"GoosePublisher_publish()"* is where the message will be sent through *"Ethernet_sendPacket()"*, being later forwarded to Linux, Win32 or Mac OS Hal. Here, the approach is quite simple. Basically, each time a GOOSE message would be sent in LAN, that message will also be sent through UDP from Gateway Sender to the Gateway Receiver. In order to accomplish that, a new function RGOOSE was added to goose_publisher.c as well as some variables and includes.

```
392     Ethernet_sendPacket(self->ethernetSocket, self->buffer, self->payloadStart + payloadLength);
393     RGoose(self->buffer, self->payloadStart + payloadLength);
394     return 0;
```

Figure 4.10: Addition of RGoose function in line 393

From line 38 to 40 of Figure 4.11, there are three variables important to notice:

- **SERVER:** where is stored the IP of the Gateway Receiver.
- **BUFLEN:** usually 512 bytes, but changed to 1024 bytes, since goose_publisher is sending more than 512 bytes. Now, there is some margin to work with.
- **PORT:** the port where the messages will be sent. Very important to remember this port because it will be needed to configure in router Port Forwarding.

```
32  //send udp
33  #include<stdio.h> //printf
34  #include<string.h> //memset
35  #include<stdlib.h> //exit(0);
36  #include<arpa/inet.h>
37  #include<sys/socket.h>
38  #define SERVER "192.168.1.15" //Ip Destination
39  #define BUFLEN 1024  //Max length of buffer
40  #define PORT 8888   //The port where data will be sent
```

Figure 4.11: Code added in begin of goose_publisher.c

At the end of goose_publisher.c, two more functions are added. The code of both functions can be found in Appendix K.

- void die(char *s);
- void RGoose(uint8_t* buffer, int packetSize);

void die(char *s);

- A function that will allow to exit the program in case of any error.

void RGoose(uint8_t* buffer, int packetSize);

- This is the function where the UDP packets will be sent over the network. Here, two arguments are passed to RGoose:
    - uint8_t* buffer: this is a pointer to the memory where GOOSE message data is allocated.
    - int packetSize: the total size of payload of UDP, which includes total size of payload of GOOSE in addition with the size of GOOSE headers.
- Then, a socket is created, the connection established and the message is sent through *"sendto()"* function.
- Also, in the code, after the message is sent, it will be listening for a reply that will come from Gateway Receiver with the same content to the original sent message. As soon as receive that data, the function ends and closes the connection successfully.

However, there are some interesting questions that have to be made in this case, "why the Gateway Sender will wait for a reply from the Gateway Receiver?" Of course, UDP doesn't need acknowledgment neither GOOSE protocol waits for a reply when a message is sent. However, it is extremely important to have this reply for later time calculations, removing the necessity to use additional synchronization methods. Also, with this reply, it becomes possible to capture the messages on both Gateway Sender and Receiver side and use them to calculate the OWTT, as desired. Even if the application is not a priority to the developed work in this document, it's important to remark that this additional code doesn't belong to the normal application, being used in cases to calculate times like this one.

4.6.3.3 Gateway Receiver

On the other side of Gateway Sender, there is a Gateway Receiver that is listening for the UDP messages and will parse them to print on command line. Here, parsing the message is not that important, because it belongs to the application. However, in Gateway Receiver, the messages will be split in order to evaluate if everything is sent how it was supposed to be.

Again, like in Gateway Sender, in Gateway Receiver are added the same BUFLEN and PORT.

```
31  //RGoose
32  #include<stdio.h> //printf
33  #include<string.h> //memset
34  #include<stdlib.h> //exit(0);
35  #include<arpa/inet.h>
36  #include<sys/socket.h>
37  #include <unistd.h>
38  #define BUFLEN 1024  //Max length of buffer
39  #define PORT 8888   //The port on which to send data
```

Figure 4.12: Gateway Receiver includes and defines

In Appendix L can be found part of the code that was developed. Describing the main function of the program:

- A socket is created and is waiting in an infinite loop for an UDP message through the port 8888.
- As soon as the message is received, it is printed the source IP and the port as well.
- Without spending any additional time, the same message is sent again to Gateway Sender.
- As soon as the process of receiving and sending is completed, the message is split and printed on command line to be evaluated. This is the best way to not interfere with processor and not increasing the delay between receive and sent process.
- Once the splitting process is done, the program will return to the begin of infinite loop and will wait to arrive a new message.

4.6.4 VPN Tunnelling

4.6.4.1 Introduction

Concluding the objective to send R-GOOSE via UDP, it is necessary to make the proper modifications to implement the second method to send data over WAN. Therefore, after making a research about the options available to implement a VPN Server, OpenVPN is the one that seems better for this case, due to its flexibility. OpenVPN is a free and open-source software application that implements VPNs. Additionally, OpenVPN can be incorporated in several embedded systems as well as being incorporated in routers firmware such the DD-WRT, that contains the OpenVPN Server function.

Considering the router provided by the Institute of Telecommunications of Aveiro, the Linksys WRT1200AC, it is suitable to receive the DD-WRT firmware which allows to create a native VPN Server from the laboratory. Later, for the VPN clients, exist two options: incorporate the client function inside the router or incorporate it each individually inside the Raspberry Pi. First option allows all the devices connected to the router to be in the same VPN LAN. On the other side, with the second option, only the devices that are able to connect to VPN Server inside its OS, will be in VPN LAN. Both options have advantages and disadvantages and some of them are displayed in Table 4.4.

VPN function inside the Router	VPN function inside the Raspberry Pi
• Processing occurs in router. Good if a low number of devices are connected, bad if there are too much since it could lead a slowness to manage all the traffic • Unnecessary traffic flow through VPN • Less secure since once a device is plugged to Router, it is automatically VPN LAN • Easy to configure	• Each device processing its own application. The opposite when compared with VPN function in router • Could interact with devices in router's LAN that aren't in VPN's LAN • More secure, each device has its own authentication • Has to be configure in each separated device

Table 4.4: Comparison between VPN function implemented in Router or Raspberry Pi

Both options could be applied in the same WAN, depending the case study. In this particular work, VPN function for the Clients is implemented individually on each Raspberry Pi. The reason behind this decision relates with the 4G routers as well as Optical Fibre ones used to perform the tests don't support natively VPN Client function. As consequence, even if it was added an extra router with this capability, it would introduce more latency because of its additional processing. So, the choice lies on the second option.

4.6.4.2 Server Configuration

In the first place, before proceeding with the Server configuration it is necessary to flash a new DD-WRT firmware to the Linksys WRT1200AC. Original firmware also allows its users to create an OpenVPN Server however, it is very limited in terms of flexibility and configurations, not giving all the tools that are required to configure the Server as wished.

It is pretty intuitive to flash the new firmware in the router. It is necessary to download the correct firmware from DD-WRT website [81], searching for WRT1200AC Version 2. After the download is completed, open a new browser webpage, insert the LAN IP in the tab and enter the login credentials. Once the login is done successfully, it is necessary to enter in the Administration tab and press the Firmware Upgrade button. More details could be found here [82].

As soon as the firmware is successfully upgraded, it is necessary to create the certificates, which will be the authentication method used. First, it is necessary to download OpenVPN GUI [83] and during the installation it is needed to also sign "Easy RSA" option in order to be possible to generate the certificates. They are produced in windows command line and the procedure is exactly the same found here, in topic "Creating Certificates Using Easy RSA in Windows" [84].

Next, it is necessary to enter router's GUI webpage to proceed with modifications:

- Check memory available:
    - The key size that will be used will be 2048, which will require 6000 bytes available in router's NVRAM to implement the VPN Server, otherwise the router can be bricked.
    - Press Administration tab, and Command sub-tab.
    - In Commands type: $nvram show | grep size
    - If there is enough memory available, it is safe to proceed with the implementation otherwise, an option is to make factory reset that will free NVRAM memory however, other configurations will also be lost.

- Basic Setup:

Important configurations	
WAN Connection Type	Automatic Configuration DHCP
Router IP	192.168.2.1
Network Mask	255.255.255.0
DHCP Server	Enable
Start IP Address	192.168.2.100

Table 4.5: Basic setup in VPN Server router

In basic setup, it is important that the router and its network don't have the standard "192.168.1.1" IP in order to not enter in conflict with ISP router that usually uses the 192.168.1.0 network. Start IP address shouldn't occupy all the range, because, as it will be shown below, it is necessary some IPs to give to VPN Server clients.

- VPN Server configuration:
    - Press Services tab and VPN sub-tab

Important configurations	
WAN Up	Server start when the router receives an WAN IP
Config as Server	To be configure with router webpage
Server Mode	TAP
Pool Start IP	192.168.2.10
Pool End IP	192.168.2.20
Gateway	192.168.2.1
Network Mask	255.255.255.0
Port	1010
Tunnel Protocol	UDP
TLS Cipher	None
Allow Client to Client	Enable

Table 4.6: Configurations to create VPN Server

The first two configurations are explained in table. The third one, Server Mode is very important since there are two options: TUN or TAP. Here [85] can be found more information about the advantages of each other. TUN is better in terms of performance since it only allows the transportation of layer 3 IP packets, leaving the layer 2 and its broadcasts behind. However, the protocol in study, GOOSE, is a layer 2 protocol, which requires

that Ethernet frames can be passed over VPN tunnel. So, TAP configuration is the chosen one, admitting this way the GOOSE packets be transferred in the VPN tunnel.

Next, Pool Start and End IP, have to be in the same network as the router and the pool range cannot match with the range given in basic configuration. Since no more than three or four devices will be used during the tests, this range is more than enough. Gateway points to the router and the Server network mask is the same as the router.

The port number is 1010 because, some standard ports may be blocked by ISP and this way it may be avoided unreached traffic to its destination. Protocol is the UDP due to it being the faster and, with retransmission mechanism, it isn't necessary the acknowledgment of sent data.

TLS Cipher is left in blank. During the connection process, the TLS Cipher will be negotiated as well as other fields not shown above. After the negotiation, TLS Cipher is: TLS-RSA-WITH-AES-256-GCM-SHA384.

Last but not least, it's necessary to allow the communication between clients. There are other configurations not shown here that can be found in Appendix C.

- Certificates:

Public Server Cert	server.crt
CA Cert	ca.crt
Private Server Key	server.key
DH PEM	dh2048.pem

Table 4.7: Certificates to create VPN Server

In section 4.6.4.2 are described how these certificates are generated. Open the files with any text editor and copy the certificates, starting with "-----BEGIN CERTIFICATE-----" and ending with "-----END CERTIFICATE-----". Additionally, no more configurations are need in this VPN sub-tab leaving other fields in blank.

- Startup and Firewall:
    - First, press again Administration tab and Commands sub-tab.
    - For startup:
        - Paste the following command lines
        - openvpn --mktun --dev tap0
          brctl addif br0 tap0
          ifconfig tap0 0.0.0.0 promisc up

        - Press Save Startup
    - For firewall:
        - Paste the following command lines
        - iptables -A INPUT -i tap0 -j ACCEPT
          iptables -I INPUT -p udp --dport 1010 -j ACCEPT
        - Press Save Firewall

- Reboot the router and OpenVPN Server is active.

Finally, some debug procedure should also be done in order to eliminate some errors that may occur. A larger part of the description can be found here[84]. However, each case is a case and it is necessary to perform the different modifications.

Most of the time the router with VPN Server will be connected to the ISP router, meaning that it is required to forward the UDP traffic from ISP router to VPN Server router. Usually in security or NAT tab, it is possible to find a configuration called Port Forwarding and add the desired rule. In the example illustrated in Figure 4.13, it should be added a rule with port 1010, UDP protocol to 192.168.1.5, which is the IP of VPN Server router in ISP Router LAN.

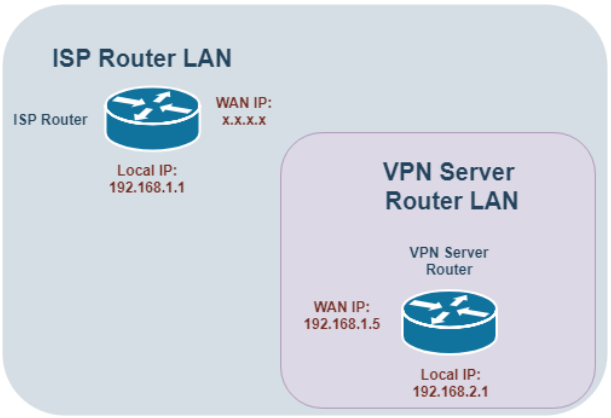


Figure 4.13: ISP and VPN Server LAN

4.6.4.3 Client Configuration

The first step after configuring the VPN Server is to generate a Client configuration file according to the Server configuration. So, with the help of tutorial, the configuration file is used to connect to VPN Server during this work, is the Appendix D. Note that, the certificates are inside the file. Every time that is necessary to connect to a different WAN IP, it is also necessary to modify the first line of Client configuration file. So, replace

WAN_IP in “remote WAN_IP 1010” for the WAN IP that is wanted to connect and save it with .ovpn extension.

With a Client configuration file created, place the configuration file in Home directory, install the software in Raspberry Pi and connect it to the Server.

- Install OpenVPN software:
    - $sudo apt-get install openvpn
- Connect to the server:
    - $sudo openvpn ClienConfigurationFile.ovpn

After these steps, the connection should be achieved successfully, getting this way a new network interface called tap0 and a new IP in the pool range designed in Server, which are 192.168.2.10-20.

In the end, using this implementation, all the Raspberry Pis should be connected to the VPN Server allowing the exchange of GOOSE messages, by one of the communication mechanisms provided by the IEC61850 standard. Additionally, it will be performed an extra test topology to study, which consists in utilize the R-GOOSE implementation via UDP along with VPN tunnel. In other words, send the R-GOOSE, in the VPN tunnel instead of traditional GOOSE.

5. Results

5.1 Introduction

After all the changes and additions referred in the previous chapter, it is necessary to perform the tests and collect data as well as to analysed it.

Initially, in section 5.2, it will be explained the procedure made to capture UDP messages, including all the steps of the one changed state as well as the travel of each message. Later, in section 5.3, it will be explained the capture procedure for the R-GOOSE messages.

After the explanation of capturing messages using R-GOOSE, it's time to collect the data and analyse it. First tests are performed in the same LAN in order to introduce the method of processing and analysis the data and understand the impact of the hardware in the next captures. Here, once the messages will only travel in the same LAN and it is used ethernet cables, the connection is very steady, being good to analyse the impact of having GUI and non-GUI OS in Raspberry Pi.

In section 5.4, it will be explained the capture procedure for the GOOSE messages. It is very similar procedure with the one used to capture the UDP messages. Here, it is also performed the tests using Wi-Fi instead of ethernet cable inside the same LAN and see if the results are good enough to serve the requirements in Table 3.1.

After raising the awareness about additional delays intrinsic of LAN environment, it will be performed, in section 5.5, the R-GOOSE tests with different types of technologies such as ADSL, Optical Fibre and 4G-LTE and making a final sum up of the latency impact of these technologies in the transportation of the messages. The same technologies and similar tests are performed in section 5.6, however in this section besides measuring the transmission time of the GOOSE messages, it will also be measured the transmission time of UDP messages inside the VPN LAN. Testing this transmission time of UDP messages isn't contemplated in IEC61850 standard although, this possibility is also interesting to be analysed. In the end, again, a recap comparing both approaches inside the VPN and if the R-GOOSE inside a VPN tunnel is a viable option or not.

Last but not least, a final discussion about the pros and cons about each used technology and which communication mechanism should be used to deliver messages to outside of substation, between VPN with GOOSE and R-GOOSE.

5.2 Procedure to capture UDP messages

5.2.1 Introduction

This explanation presents the methodology to analysis the R-GOOSE transmission time, where these messages are captured as UDP traffic. So, all the three Raspberry Pis are in the same LAN connected to the router, along with a computer. VNC and SSH are the protocols used to interact with the Raspberry Pis and later will be analysed the impact of using each other, since VNC is used to interact with GUI while SSH is used to interact only with command line.

Considering what was described from 4.6.2.2 to 4.6.2.4 for the Server and the Client, the GOOSE traffic flow will not be detailed again, being only approached the flow of UDP messages in the LAN. The explanations will consist in two parts: the first about one sequence, meaning the flow of one UDP message; the second about one state, which include the retransmission process. Note that the modifications described in 4.6.3 for UDP are applied here.

5.2.2 One Sequence Transmission

The topology in study is represented in Figure 5.1. Below, it is explained how the messages flow between the three entities, Server/Gateway Sender, Client and Gateway Receiver.

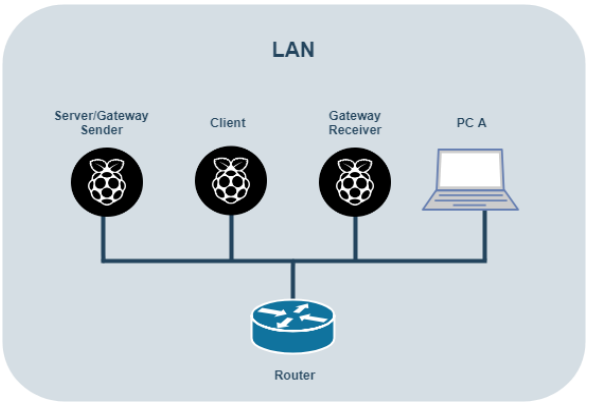


Figure 5.1: Different devices in the same LAN

1. In Figure 5.1, there is:
   1.1. **PC A** is the computer that interacts with Raspberry Pis so those can execute commands.
   1.2. **Server/Gateway Sender** is one Raspberry where the biggest part of processing and analysis will be done. It has two functions: Server for running GOOSE protocol and the modification module, called as Gateway Sender to send R-GOOSE.

1.3. **Client** is another Raspberry that acts as a client. Here, it will be provoked intentionally Over Voltage failures to generate new events in order to force the changes of state.

1.4. **Gateway Receiver** is the last Raspberry, where it will receive the R-GOOSE. He will return the same message to the Gateway Sender (this only happens for our tests and measures some delivering times).

2. In Figure 5.2, there is tcpdump capture open in Wireshark, with some points that will be used following messages. It's represented what happens when the Client changes its state.

3. Considering again the Figure 5.1, Figure 5.2 and Figure 5.3:

3.1. First, in message 324, represented with point a), the Server broadcasts an "heartbeat" for everyone, sending its data.

3.2. Since the Gateway Sender is added, it will send also the R-GOOSE message, displayed in message 327 and represented on point b), to Gateway Receiver.

3.3. At Gateway Receiver, it is received the R-GOOSE message, displayed in message 52, on point f). After a quick short time (processing time) it will be immediately sent the same message by the Gateway Receiver, shown in message 53 and on point g), to Gateway Sender.

3.4. In message 328, on point h), it is possible to observe that the message is received by the Server/Gateway Sender.

3.5. Something changes on Client, for example and as referred before, occurs an over voltage problem. On point c), it is shown that information is broadcasted to everyone by the Client and received also by the Server, displayed in message 408, on point d).

3.6. Server updates its database and broadcast a GOOSE message to everyone, displayed in message 309, on second point a), sending also the R-GOOSE messages due to its repeating the process, displayed in message 410, on point b).

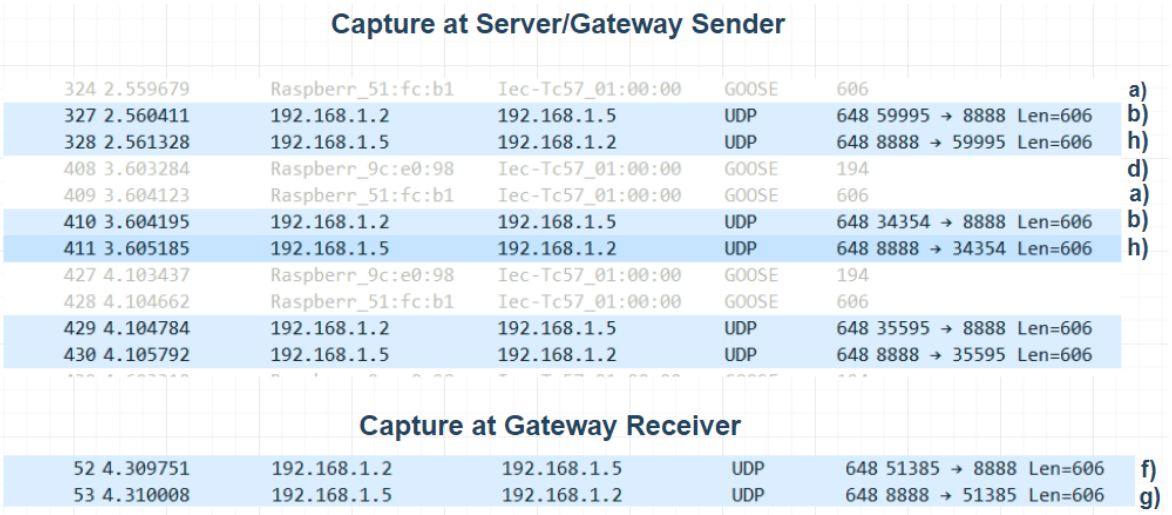


Figure 5.2: GOOSE and R-GOOSE capture

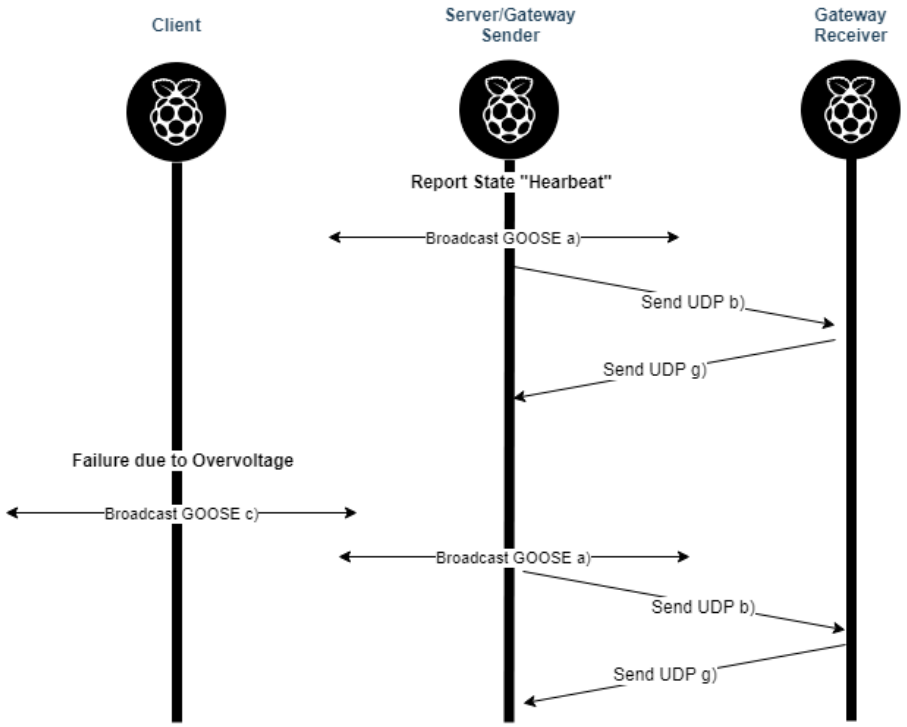


Figure 5.3: Messages diagram of GOOSE and R-GOOSE

5.2.3 One State Transmission

Similar to the flow of the messages present in Sequence Transmission, this capture in Figure 5.4 is also showing the transmission and the retransmissions:

1. Continuing the capture in the previous topic since the messages represented are the same.

1.1. In message 408, Client sends a broadcast GOOSE message.

1.2. In message 409, Server sends a broadcast GOOSE message with its update values.

1.3. In messages 410 and 410, sends and receive UDP message as explained before.

1.4. Now, looking for timestamp, both Client and Server send its messages 500 ms after the previous one, in message 427 and 428.

1.5. Again, 500 ms later, new messages are sent by both, displayed in messages 439 and 440.

1.6. The last two messages provoked by this change are sent in 446 and 447.

1.7. Notice that, all the messages sent before are respectively about one change of state and, as pointed before, the protocol sent four messages spaced by 500 ms between each other. During this time, UDP messages from Server/Gateway Sender are also sent when its GOOSE messages are sent.

1.8. Since, it is programmed to deliver a new failure with a delay(5000), which translates a 5 seconds delay, and the four messages took about 1.5 seconds to be sent, it remains 3.5 seconds without any change of state.

1.9. When this time ends, another message is sent to simulate a failure as it shown in message 481 and, thus, the process repeats all over again.

408	3.603284	Raspberr_9c:e0:98	Iec-Tc57_01:00:00	GOOSE	194
409	3.604123	Raspberr_51:fc:b1	Iec-Tc57_01:00:00	GOOSE	606
410	3.604195	192.168.1.2	192.168.1.5	UDP	648 34354 → 8888 Len=606
411	3.605185	192.168.1.5	192.168.1.2	UDP	648 8888 → 34354 Len=606
427	4.103437	Raspberr_9c:e0:98	Iec-Tc57_01:00:00	GOOSE	194
428	4.104662	Raspberr_51:fc:b1	Iec-Tc57_01:00:00	GOOSE	606
429	4.104784	192.168.1.2	192.168.1.5	UDP	648 35595 → 8888 Len=606
430	4.105792	192.168.1.5	192.168.1.2	UDP	648 8888 → 35595 Len=606
439	4.603310	Raspberr_9c:e0:98	Iec-Tc57_01:00:00	GOOSE	194
440	4.604669	Raspberr_51:fc:b1	Iec-Tc57_01:00:00	GOOSE	606
441	4.604788	192.168.1.2	192.168.1.5	UDP	648 33750 → 8888 Len=606
442	4.605758	192.168.1.5	192.168.1.2	UDP	648 8888 → 33750 Len=606
446	5.103842	Raspberr_9c:e0:98	Iec-Tc57_01:00:00	GOOSE	194
447	5.104017	Raspberr_51:fc:b1	Iec-Tc57_01:00:00	GOOSE	606
448	5.104148	192.168.1.2	192.168.1.5	UDP	648 51385 → 8888 Len=606
449	5.105167	192.168.1.5	192.168.1.2	UDP	648 8888 → 51385 Len=606
481	8.604031	Raspberr_9c:e0:98	Iec-Tc57_01:00:00	GOOSE	194
482	8.604598	Raspberr_51:fc:b1	Iec-Tc57_01:00:00	GOOSE	606
483	8.604711	192.168.1.2	192.168.1.5	UDP	648 34794 → 8888 Len=606
484	8.605692	192.168.1.5	192.168.1.2	UDP	648 8888 → 34794 Len=606

Figure 5.4: Change of state in Client and then, in Server

An extra thing to point is that both sequence and state transmission are represented by a variable. For the sequence, the variable sqNum "is increased for every consecutive GOOSE message without state" while for the state, the variable stNum "is increased if any of the values in the GOOSE data set changed due to a valid trigger condition". Additionally, "when a state change occurs (stNum is increased) the sequence number (sqNum) will be reset" [86].

5.2.4 Conclusions

To sum up, all the performed tests will be similar to this one and usually, the changes are made in Gateway Receiver in order to evaluate the transmission times through ADSL, Optical Fibre or 4G-LTE. All the changes will be highlighted in the beginning of each test in order to achieve better understanding and withdraw the maximum information about each one. As side note, when an UDP test will be performed, the GOOSE messages will not recall attention, since only UDP ones will be analysed afterwards.

5.3 Tests in same LAN

5.3.1 Test performed with Ethernet Cable

5.3.1.1 Procedure

To start with the captures, all the Raspberry Pis need to be turn on and connected by ethernet cable to the router, as illustrated in topology of Figure 5.1.

Denote that no other program is running, which allows not to spend any processing capacity in unwanted processes. Time is crucial and it's required to optimize everything that has impact. A folder in each Raspberry Pi is created using the command line in order to save the captures. Additionally, there are some commands that are used to collect them.

Linux command used to capture the packets:

1. Gateway Sender: $ sudo tcpdump – w Server_test_01.pcap -i eth0
2. Gateway Receiver: $ sudo tcpdump – w Gateway_test_01.pcap -i eth0

After starting the captures, it is necessary to run the code in the three Raspberry Pis by this order:

1. Gateway Receiver: $ sudo ./
2. Server/Gateway Sender: $ sudo ./
3. Client: $ sudo ./

Onwards, all the tests are performed in the same conditions, 40 changes of State in the Server/Gateway Sender and Client, capturing in this way the same number of messages, 314, corresponding to 157 UDP time measures. Generally, this is a large number of measures to analyse, since capturing 10 measures usually gives a good estimate. In this case, 10 measures would be fine since this test is performed with ethernet, but later, other mediums will be used and they aren't as stable as ethernet. Needless to say that the stability of wireless communications depends from a lot of variables and, accounting with that, it necessary to have a larger capture size turning the result much closer to the most common and probable scenario.

Afterwards, both captures are opened in Wireshark:

- On Server/Gateway Sender applying the filter:
  (ip.addr == 192.168.1.2 and ip.addr == 192.168.1.5)
- On Gateway Receiver applying the filter:
  (ip.addr == 192.168.1.2 and ip.addr == 192.168.1.5)

Using the respective filter, the relevant messages are exported to CSV file so that later, the time of the messages can be uploaded to Matlab. Hence, it is obtained graphics and important values about OWTT such as, mean, standard deviation, minimum, maximum and 95/99 percent confidence interval.

5.3.1.2 Time Calculation

This is the method used to measure the transmission time of R-GOOSE and represented in Figure 5.5.

$$RTT = GS\ Time - GR\ Time \quad 5.1$$

GS Time is captured with tcpdump command at Gateway Sender, where:

1. T1 is the time when UDP message leaves the GS to GR.
2. T4 is the time when UDP message arrives at GS from GR.

GR Time is captured with tcpdump command at Gateway Receiver, where:

1. T2 is the time when UDP message arrives at GR from GS.
2. T3 is the time when UDP message leaves from GR to GS.

As said before, it is used a Matlab Script to obtain the RTT. Either GOOSE or UDP, we are not interested in total RTT but in OWTT. The following Figure 5.5 shows that a good approximation of OWTT, is divide the RTT by 2. However, this is not true in connections where are used different technologies in both terminations, since it's hard to synchronize both devices in terms of time and it would require extra network protocols running in background. The Matlab script can be found in Appendix E.

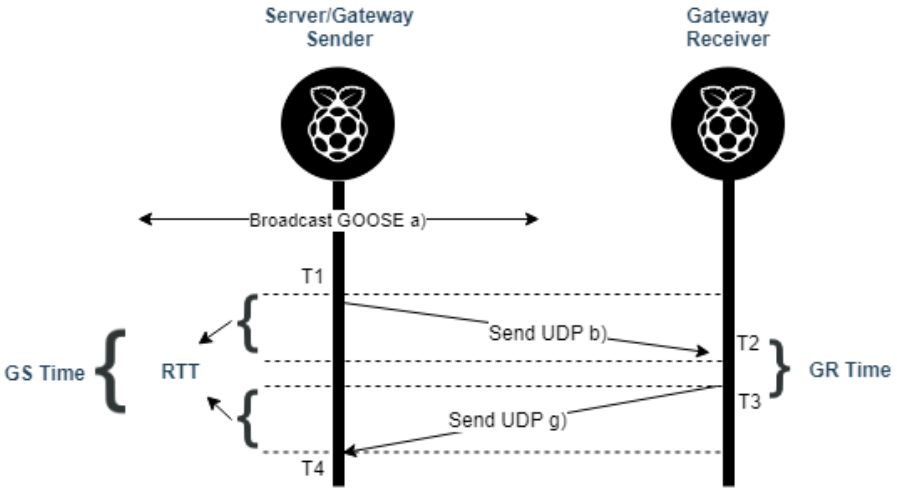


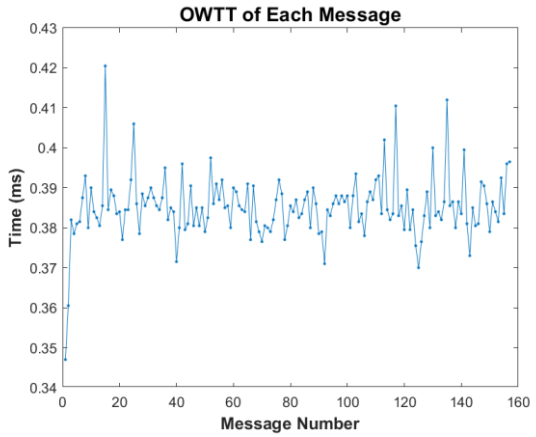
Figure 5.5: Message diagram of one R-GOOSE message

5.3.1.3 Results Analysis

After the procedure described before, it is possible to calculate also some values related with OWTT: mean, standard deviation, minimum, maximum and confidence intervals of 95% and 99%, as stated before.

During the rest of the document, other tables like the Table 5.1 will appear every time at the end of each different test performed. A good method to detect eventual errors is to compare the values in the tables with

the ones in the graphics, making this way a first "eye debug". In the table, Std. Dev. should have a lower value, as well as 95/99-Perc when compared with the mean. Additionally, in an ideal case, also the minimum and maximum should have its values the closest possible to the mean. Looking to the graphic, it is important to notice if there are samples that escape from the mean which naturally, also escape from the most of other points. A bad sample is if it escapes more or less from the 50% of the mean value. In this case with a stable medium, all the samples seem to be good to analyse.



Graphic 5.1: Messages of test performed with ethernet cable

Data	*Time (ms)*
Mean	0.3851
Std. Dev.	0.0079
Min	0.3470
Max	0.4205
95-Perc	0.0012
99-Perc	0.0016

Table 5.1: Results of performed test with ethernet cable

5.3.1.4 Conclusion

This test provides expected results in terms of transmission time. However, it is important to keep these values in mind to be critical when other ones are collected because, since they should be higher than these ones.

Additionally, this test is also important to consolidate and assimilate the execution of the capture procedure since it is less susceptible to errors that could occurs during the capture. In the end, it is also the easiest procedure to explain. Last but not least, this is also a good test to perform with or without graphical user interface in a way to accomplish the impact of GUI on the system, which will be approached in the next topic.

5.3.2 Impact of GUI

5.3.2.1 Introduction

Insofar the ethernet cable test was performed in previous section 5.3.1, it is noticed that it could be a good opportunity to execute the same test without GUI.

In Figure 5.6, with GUI, it is used VNC Viewer to interact with Raspberry Pis and running the commands previously referred in its command line.

In Figure 5.7, without GUI, it is used SSH to make the interaction and execute the same commands, in PuTTy's windows.

Important to say, that from GUI to non-GUI test, it is also disable the GUI at boot. So, using the command $sudo raspi-config, it is possible to disable it and turn it again if necessary in the future.

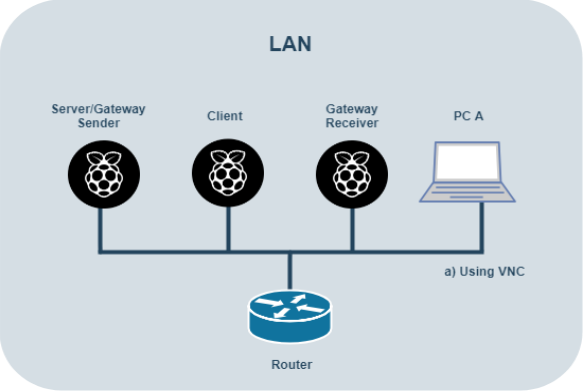


Figure 5.6: Connection done through VNC

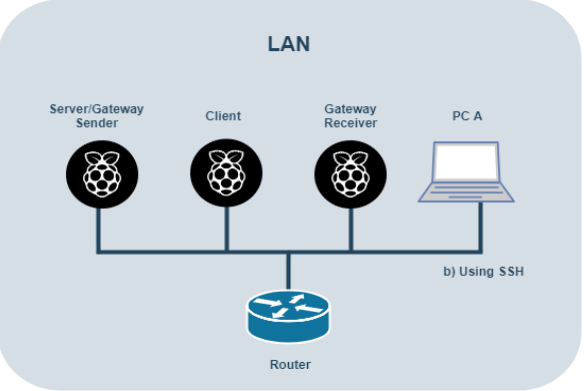


Figure 5.7: Connection done through SSH

5.3.2.2 Using GUI

OWTT

Data	*Time (ms)*
Mean	0.3851
Std. Dev.	0.0079
Min	0.3470
Max	0.4205
95-Perc	0.0012
99-Perc	0.0016

Table 5.2: OWTT using GUI

Time spend at Gateway Receiver

Data	*Time (ms)*
Mean	0.2351
Std. Dev.	0.0143
Min	0.1540
Max	0.2810
95-Perc	0.0022
99-Perc	0.0029

Table 5.3: Time spend at GS using GUI

5.3.2.3 Without GUI

OWTT

Data	*Time (ms)*
Mean	0.3828
Std. Dev.	0.0111
Min	0.3615
Max	0.4225
95-Perc	0.0017
99-Perc	0.0023

Table 5.4: OWTT using non-GUI

Time spend at Gateway Receiver

Data	*Time (ms)*
Mean	0.2030
Std. Dev.	0.1070
Min	0.1160
Max	0.8450
95-Perc	0.0167
99-Perc	0.0220

Table 5.5: Time spend at GS using non-GUI

5.3.2.4 Comparison GUI vs Non-GUI

Looking for both OWTTs it is possible to observe that both are very close to each other. This is totally expected since this time only includes the transmission time, as shown in Equation 5.1 and illustrated in Figure 5.5, and not the time spent in Gateway Receiver. The difference between both transmission times is almost zero and it is displayed in Table 5.6. For this reason, it can be considered a good collection of data and thus, take as valid the values at Gateway Receiver.

On the other hand, and a little bit surprising, is the low value obtained as result of the difference between both Gateway Receiver times in GUI and non-GUI. Using GUI, the time that the UDP message arrives and a new one with the same size leaves, will take, on average, 0,2351 ms whereas using a non-GUI results in 0.2030 ms. The difference translates into the mean value of 32.0955 µs displayed in Table 5.7, which reflects the high-power that a small computer board can achieves nowadays, letting these little concerns aside.

Data	*Time (µs)*
Mean	2.3

Table 5.6: The difference of OWTT between using GUI and non-GUI

Data	*Time (µs)*
Mean	32.0955

Table 5.7: The difference of time spent at Gateway Receiver between using GUI and non-GUI

5.3.3 Test performed with Wi-Fi

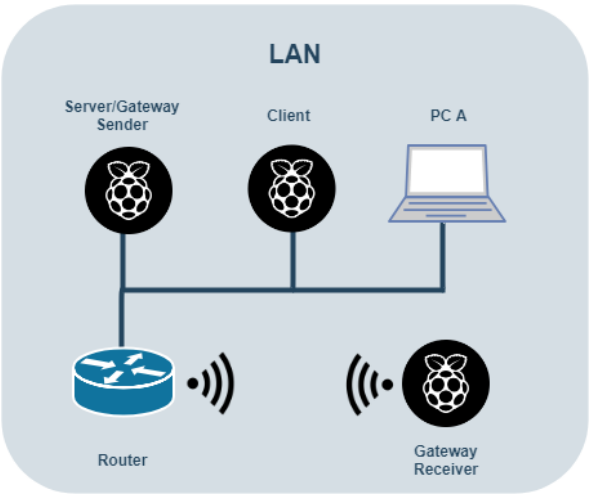


Figure 5.8: Topology used with Wi-Fi at Gateway Receiver

5.3.3.1 Introduction

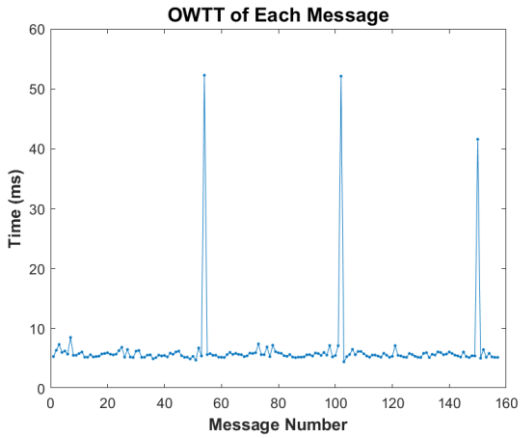
Likewise the previous test, this one is performed in same LAN, but with the particularity of the connection between the Gateway Receiver and the router being made by Wi-Fi. Introducing a wireless communication will add some latency and a wider range of collected values.

Important to remark that the connection between Gateway Sender and Router is still done through ethernet cable, as illustrated in Figure 5.8.

5.3.3.2 Results Analysis

The values reflect in a good OWTT for the UDP transmission time, since the time spent on the travel with ethernet cable is despicable when compared with Wi-Fi. Important to notice that at around every 50 messages, a very high peak time was taking to deliver the message. Looking to the capture, it was found additional layer 2 packets, but nothing that would generate such a big time.

Later, bad samples like these three will be treated carefully, but in this case they weren't since this isn't a test that merit special attention. Although, it is important to have in mind that unexpected running program or flow of other packets can influence the data collection.



Graphic 5.2: Messages of test performed with Wi-Fi

Data	*Time (ms)*
Mean	6.5251
Std. Dev.	5.9668
Min	4.4685
Max	52.2805
95-Perc	0.9334
99-Perc	1.2267

Table 5.8: Results of test performed with Wi-Fi

5.4 Procedure to capture GOOSE messages

5.4.1 GOOSE test performed with Ethernet Interface

5.4.1.1 Introduction

GOOSE was designed to operate with wired connection, although, as stated before, it could be also relevant to study if the GOOSE packets could be sent in the same LAN over a wireless technology, in this case, the Wi-Fi, due to the proximity of the devices. Therefore, considering the modifications performed in section 4.6.2.6, it is possible to capture the packets for the topology that is illustrated in Figure 5.10. For the first configuration, that was already represented in Figure 5.1 no addition modifications are perform. In both cases, the captures are made in the Server and in the Client.

Moving onto the time calculation method, this is a little bit tricky since there are two options on the table when this issue was discussed. Considering the Client/Server configuration and analysed related works, it is perceptible that using the Network Time Protocol could be too challenging due to the time remaining to perform the work. So, the other option is to use the method that was already thought for UDP messages, the method "send and receive", that was already used in previous works [61]. However, and unlike to the method that is described in section 5.2 for UDP, the sent and received message used for the time calculations don't have the same size, leading to a different transmission time between in the different messages. This is a minor issue that is taken in consideration.

5.4.1.2 Procedure

Here, all the Raspberry Pis are connected through ethernet cables, the process is very similar as the one performed before in 5.2, although, now, the capture is performed in the Raspberry Pi with the Client instead the one that has the Gateway Receiver. Also, it's good to be aware that this time will be applied different filters at each side, since now there two different sized GOOSE messages at each collection and they have to be separated.

Linux command used to capture the packets:

3. Server: $ sudo tcpdump – w Server_test_01.pcap -i eth0
4. Client: $ sudo tcpdump – w Client_test_01.pcap -i eth0 - (or wlan0 for wireless interface)

After starting the captures, it is necessary run the code on the three Raspberry Pis by this order:

1. Gateway Receiver: $ sudo ./
2. Server/Gateway Sender: $ sudo ./
3. Client: $ sudo ./

Again, all the tests are performed in same conditions, 40 changes of State in the Server/Gateway Sender and Client, which translates into 157 GOOSE messages from the Server/Gateway Sender and 157 GOOSE messages from the Client, capturing this way the same number of messages on both entities.

Afterwards, both captures are open in Wireshark:

- On Server/Gateway Sender applying the filter:
  - For variable **valsServer606**: (goose) && (eth.src == b8:27:eb:51:fc:b1)
  - For variable **valsServer194**: (goose) && (eth.src == b8:27:eb:9c:e0:98)
- On Gateway Receiver applying the filter:
  - For variable **valsClient606**: (goose) && (eth.src == b8:27:eb:51:fc:b1)
  - For variable **valsClient194**: (goose) && (eth.src == b8:27:eb:9c:e0:98)

Over again using the respective filter, the relevant messages are exported to CSV file so that later, the time of the messages can be processed and obtain the same values as before.

5.4.1.3 Time Calculation

This is the method to measure the transmission time with GOOSE messages and it's represented in Figure 5.9.

$$RTT = Server\ Time - Client\ Time$$

Server Time is captured with tcpdump command at Server side, where:

1. T1 is the time when GOOSE message with 606 bytes leaves the Server to the Client.
2. T4 is the time when GOOSE message with 194 bytes arrives at the Server from the Client.

Client Time is captured with tcpdump command at Client side, where:

1. T2 is the time when GOOSE message with 606 bytes arrives at the Client from the Server.
2. T3 is the time when GOOSE message with 194 bytes leaves the Client to the Server.

Here, it is used a similar Matlab to the one used in 5.2. The difference between both scripts can be found in Appendix F.

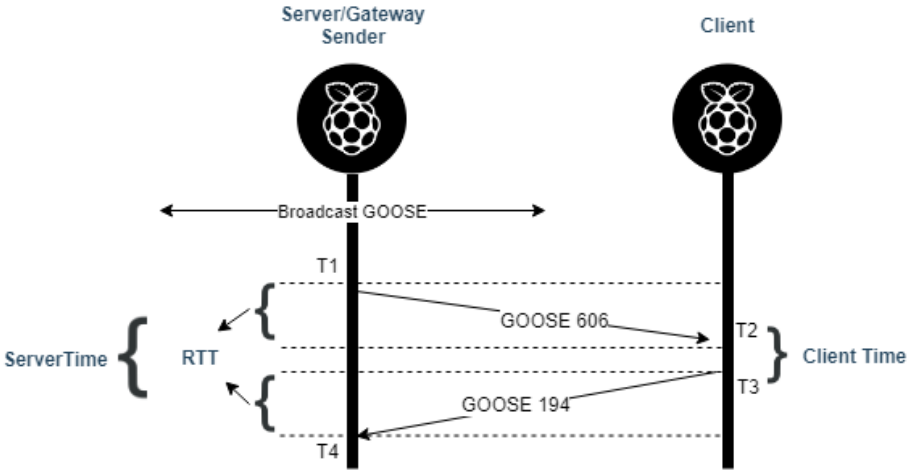
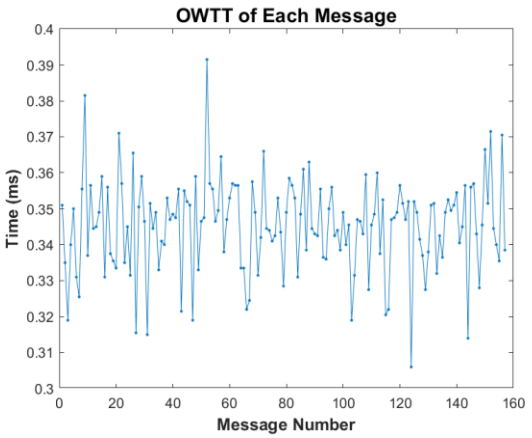


Figure 5.9: Message diagram of one GOOSE message

5.4.1.4 Results Analysis and Conclusion

This test is merely demonstrative to describe the method of capturing GOOSE messages in the same LAN. Looking to both Table 5.9 and Graphic 5.3, the collection is very similar with the one using UDP messages, found in section 5.3.1.3. It is noticeable that the mean value is around 40 µs less. This difference could be explained because the GOOSE messages have smaller size when compared with UDP ones and GOOSE is a layer 2 protocol while UDP is a layer 3 one.



Graphic 5.3: Messages of test performed with ethernet cable

Data	*Time (ms)*
Mean	0.3453
Std. Dev.	0.0130
Min	0.3060
Max	0.3915
95-Perc	0.0020
99-Perc	0.0027

Table 5.9: Messages of test performed with ethernet cable

5.4.2 GOOSE test performed with Wireless Interface

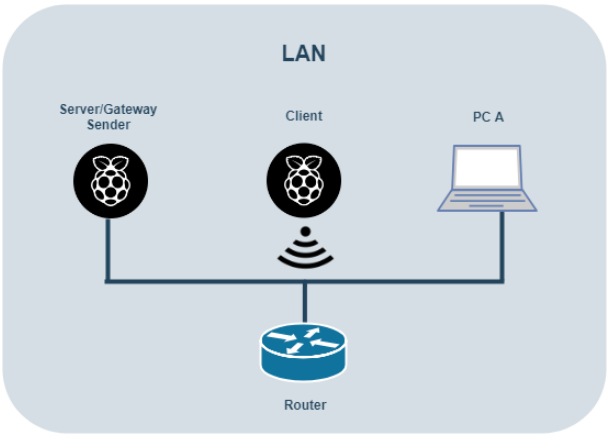
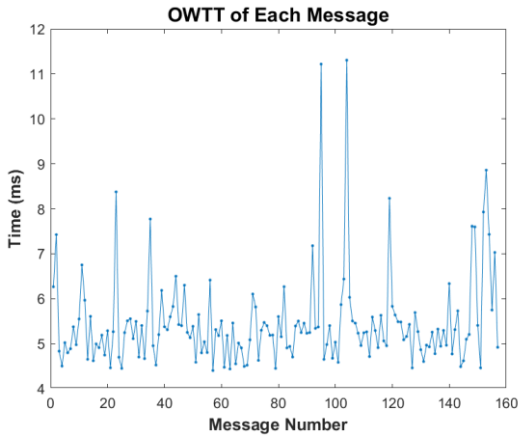


Figure 5.10: Topology used with Wi-Fi at Client

With the topology represented in Figure 5.10 and applying the modifications in section 4.6.2.6, it is possible to capture GOOSE messages using the same procedure, with minor changes, as explained in previous section 5.4.1.

Looking for both Table 5.10 and Graphic 5.4, same conclusions can be withdrawn: similar results when compared with the UDP Wi-Fi test in same LAN. Also, the mean value is close to the other one, being this one about 1.1 ms faster than the other, due to the same reasons. The wider variation of range values is a consequence of the medium used to carry the messages and not to the protocol used itself.



Graphic 5.4: Messages of test performed with Wi-Fi

Data	*Time (ms)*
Mean	5.4663
Std. Dev.	1.0521
Min	4.4020
Max	11.3060
95-Perc	0.1646
99-Perc	0.2163

Table 5.10: Results of test performed with Wi-Fi

5.5 Tests performed in different LANs over WAN using IP

5.5.1 Introduction

Up until now, the previous tests were made in the same LAN, which means that the packets didn't travel through the internet.

From now and further, the tests will be performed with the packets going through the internet until they reach its destiny, which means that they need to leave its LANs, be routed to the destiny public IP, and forwarded by the router using NAT to the device that is listening and waiting for the message. The port number used is 8888, as shown in section 4.6.3, as well as other modifications there described.

Equally important are the nomenclature used in the next topics: the first pair of words of the title refers to the ISP and which technology will be used by the Server/Gateway Sender and by the Client; the second pair of words refers to the ISP and which technology that will be used by the Gateway Receiver.

In the end, all mean values obtained for each test will be presented in Table 5.9 and a comparison will be made between them.

5.5.2 Case I: Meo ADSL – NOWO ADSL

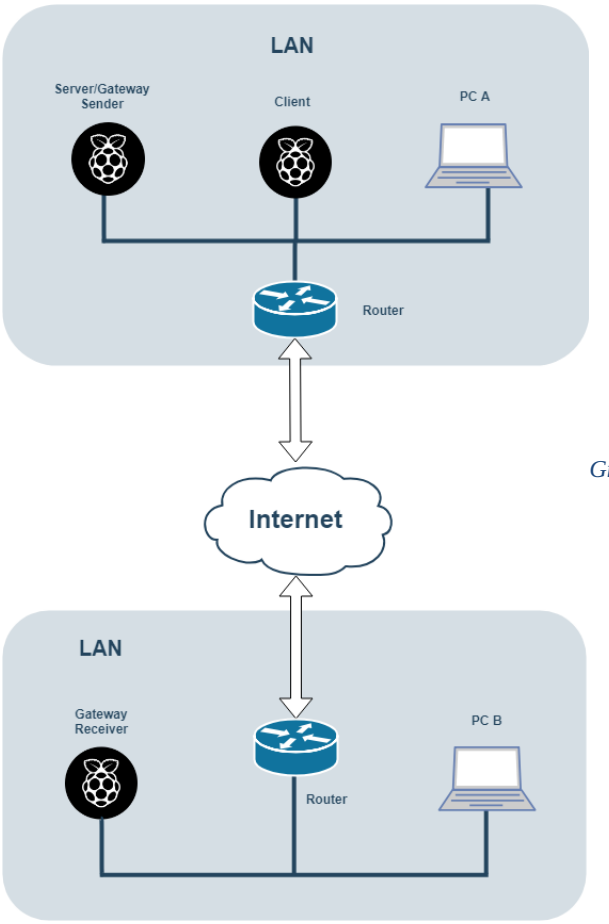
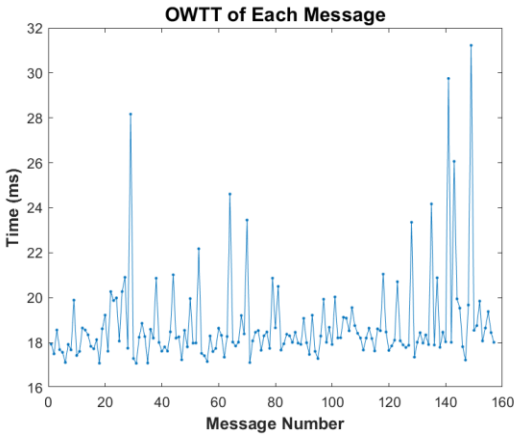


Figure 5.11: Topology of Case I



Graphic 5.5: Messages of Case I

Data	*Time (ms)*
Mean	18.8177
Std. Dev.	2.0876
Min	17.0876
Max	31.2335
95-Perc	0.3266
99-Perc	0.4292

Table 5.11: Results of Case I

Considering the Figure 5.11, it is used:

- **Server/Gateway Sender and Client:** Meo ADSL 12Mbps
- **Gateway Receiver:** NOWO ADSL 12Mbps

For convenience and because it was easier to perform this test at the time, it is considered in the work. Not only because of the previous reason, but also because it will exist a comparation to do with the Optical Fibre and then, the differences will be remarked. This configuration will not be performed with VPN and the explanation will be in section 5.7, when the comparation is made.

5.5.3 Case II: Meo Optical Fibre – Meo Optical Fibre

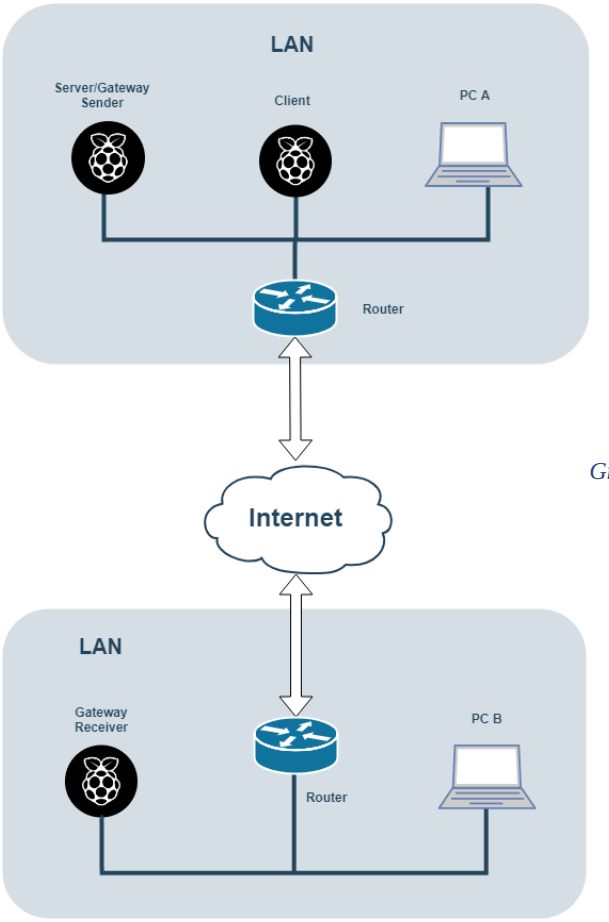
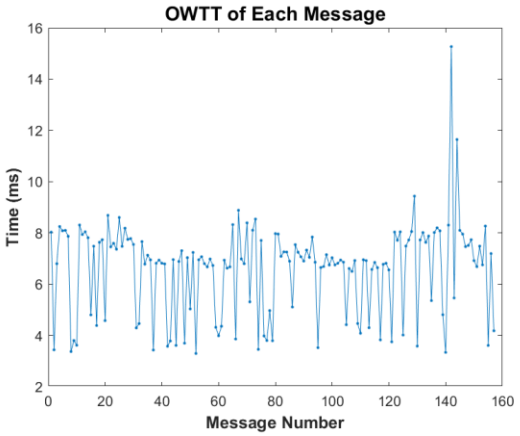


Figure 5.12: Topology of Case II



Graphic 5.6: Messages of Case II

Data	*Time (ms)*
Mean	6.6063
Std. Dev.	1.7749
Min	3.2975
Max	15.2690
95-Perc	0.2776
99-Perc	0.3649

Table 5.12: Results of Case II

Considering the Figure 5.12, it is used:

- **Server/Gateway Sender and Client:** Meo Optical Fibre 100Mbps
- **Gateway Receiver:** Meo Optical Fibre 100Mbps

In the first time that this test was performed, the two accesses to Optical Fibre were distanced by 500 meters. As result, the mean value is less than 3 milliseconds which is too low and suggesting that both accesses are in same PON, resulting in the packets only flow in a short path. A way found to solve this, was to use two different Optical Fibre accesses more distanced between each other. Thus, in the presented results, both accesses are distanced around 35 km.

5.5.4 Case III: Meo ADSL – Vodafone 4G

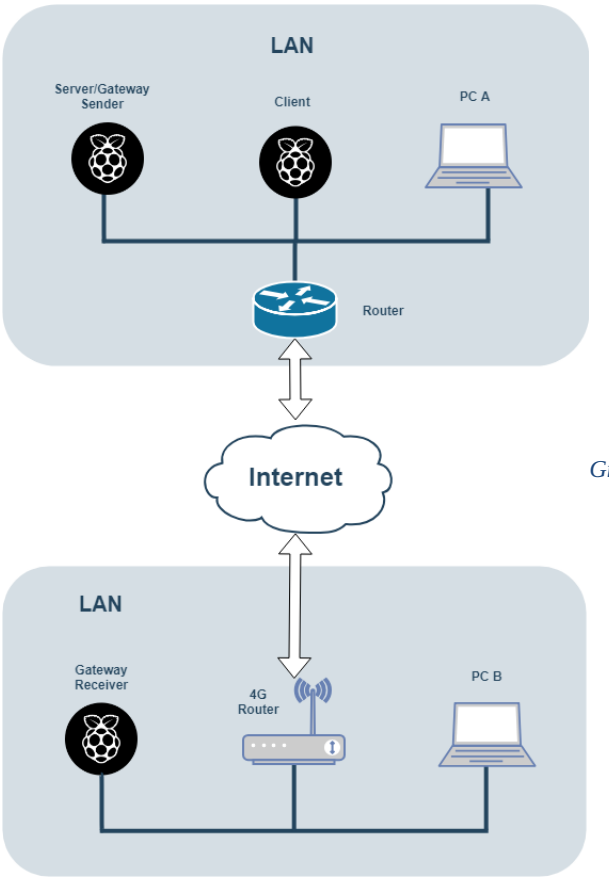
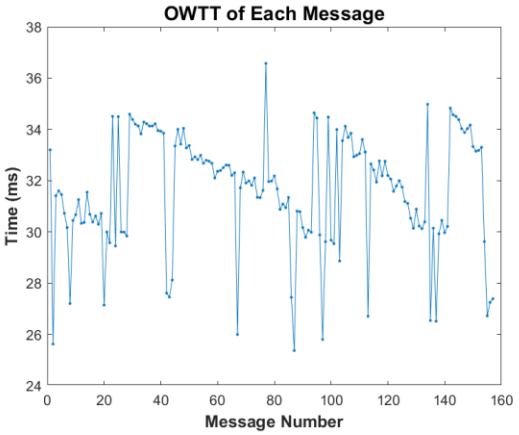


Figure 5.13: Topology of Case III



Graphic 5.7: Messages of Case III

Data	*Time (ms)*
Mean	31.6269
Std. Dev.	2.2527
Min	25.3705
Max	36.5740
95-Perc	0.3524
99-Perc	0.4631

Table 5.13: Results of Case III

Considering the Figure 5.13, it is used:

- **Server/Gateway Sender and Client:** Meo ADSL 12Mbps
- **Gateway Receiver:** Vodafone 4G-LTE up to 150Mbps

5.5.5 Case IV: Meo Optical Fibre – Vodafone 4G

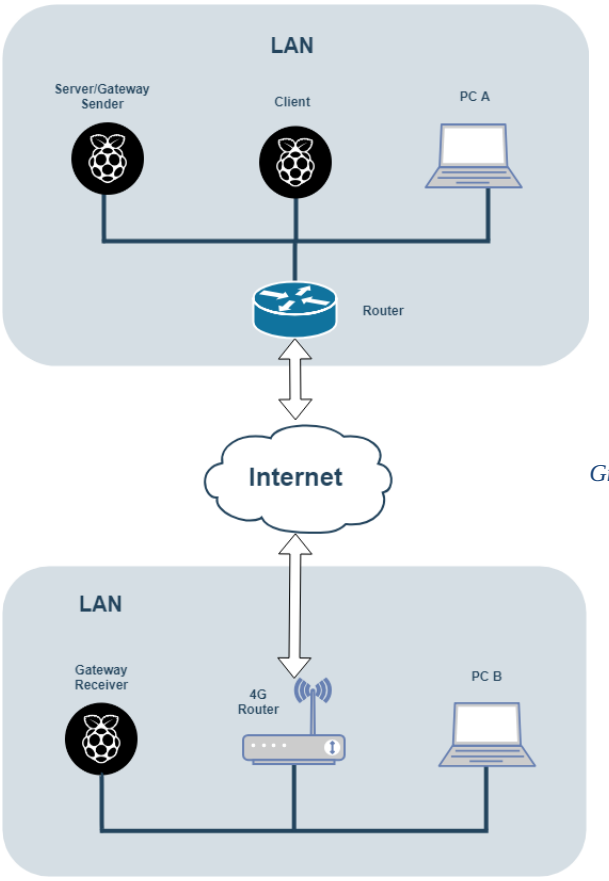
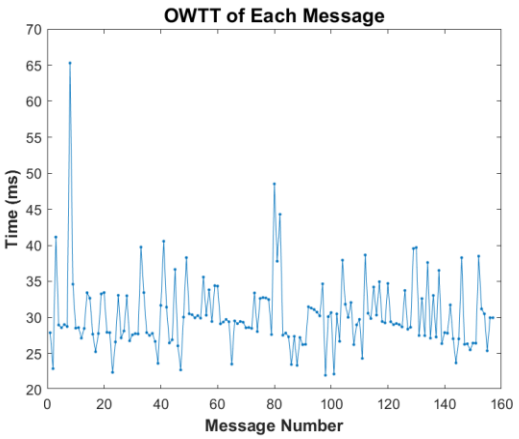


Figure 5.14: Topology of Case IV



Graphic 5.8: Messages of Case IV

Data	*Time (ms)*
Mean	30.2082
Std. Dev.	4.7054
Min	21.8975
Max	44.8570
95-Perc	0.7360
99-Perc	0.9674

Table 5.14: Results of Case IV

Considering the Figure 5.14, it is used:

- **Server/Gateway Sender and Client:** Meo Optical Fibre 100Mbps
- **Gateway Receiver:** Vodafone 4G-LTE up to 150Mbps

There is a problem in the next topic, 5.5.6, due to the establishment of the first connection. The problem is described below and, in this case, a first attempt was already done, eliminating the initials establishment times, resulting in a steady steady state.

5.5.6 Case V: Meo 4G – Vodafone 4G

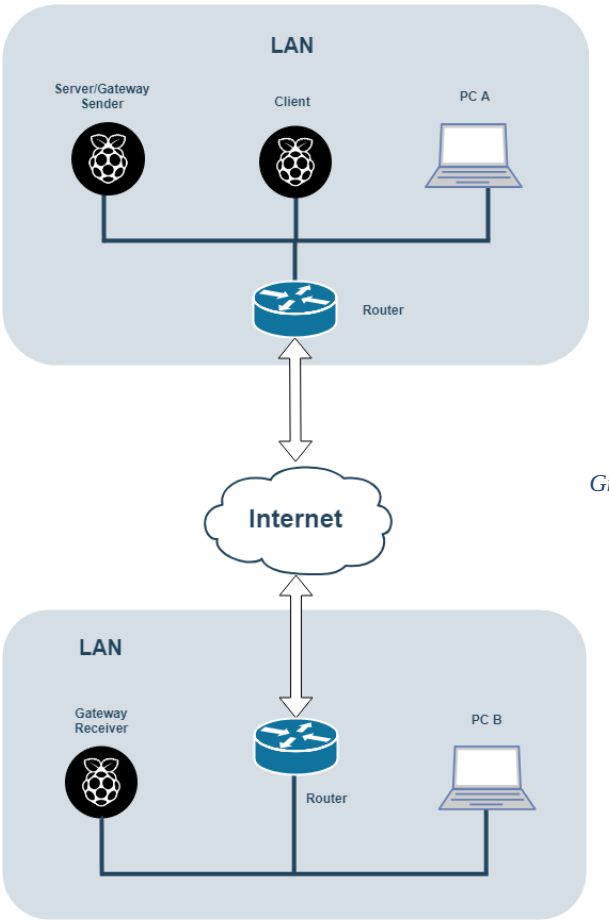
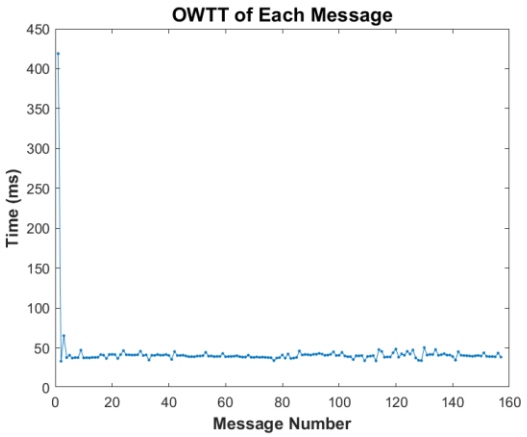


Figure 5.15: Topology of Case V



Graphic 5.9: Messages of Case V

Data	*Time (ms)*
Mean	42.7377
Std. Dev.	30.4300
Min	33.2655
Max	419.0005
95-Perc	4.7600
99-Perc	6.2560

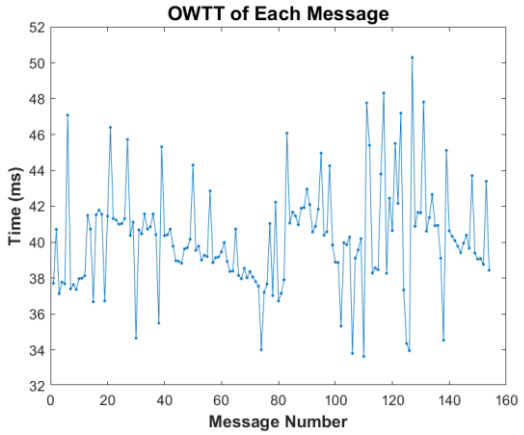
Table 5.15: Results of Case V

Considering the Figure 5.15, it is used:

- **Server/Gateway Sender and Client:** Meo 4G-LTE up to 150Mbps
- **Gateway Receiver:** Vodafone 4G-LTE up to 150Mbps

Looking to the Graphic 5.9, it stands out the first value since it is very high when compared with the other ones. This abnormal value results due to the first time that the connection is established. Addressing this suspicion, the first three values were removed, generating a new graphic and filling a new Table 5.16 with the new calculated values.

5.5.6.1 Steady State

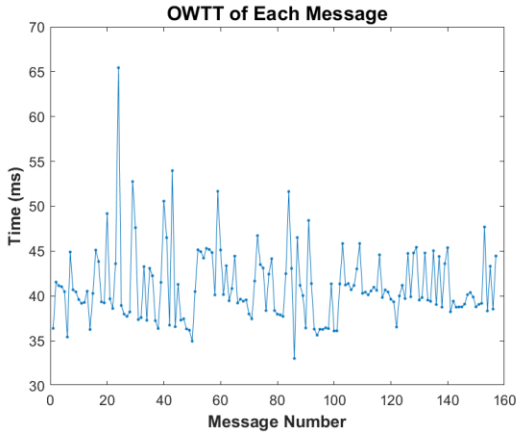


Graphic 5.10: Messages of Case V in Steady State

Data	*Time (ms)*
Mean	40.2103
Std. Dev.	2.9198
Min	33.6290
Max	50.2935
95-Perc	0.4612
99-Perc	0.6061

Table 5.16: Results of Case V in Steady State

5.5.6.2 Extra test performed



Graphic 5.11: Messages of Case V of an extra test performed

Data	*Time (ms)*
Mean	41.1264
Std. Dev.	4.1931
Min	33.0365
Max	65.4460
95-Perc	0.6559
99-Perc	0.8620

Table 5.17: Results of Case V of an extra test performed

A second test is performed to see if the high initial values that appear in Graphic 5.9 are caused due to it be the first connection. As can be seen in Table 5.17, the values are very similar to the ones in Table 5.16 where it were removed the values of the establishment state, confirming that in steady state they should be almost equal.

As done in the previous section 5.5.5, from now and further, it will always be performed a first connection to eliminate establishment times and obtain values that correspond to steady states.

5.5.7 Case VI: Vodafone 4G – Vodafone 4G

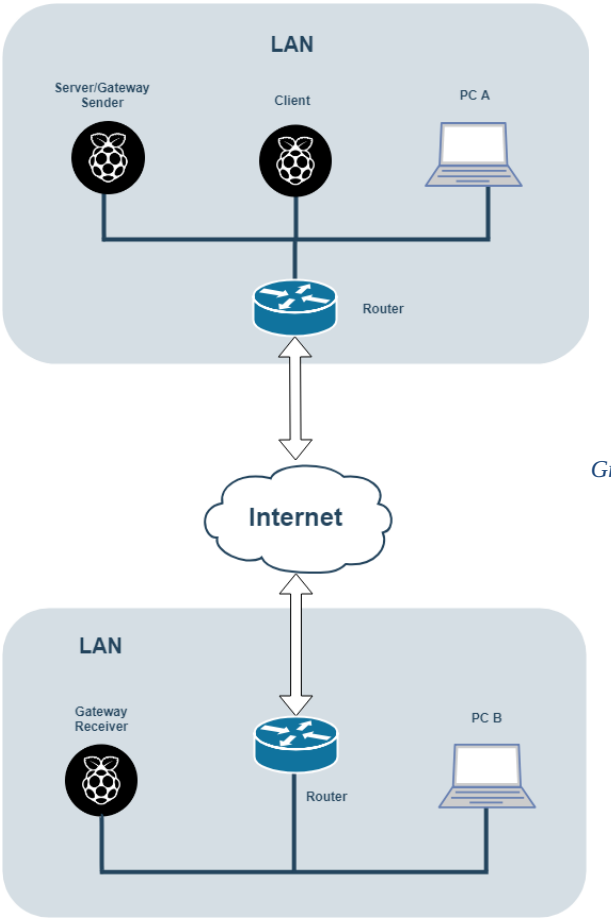
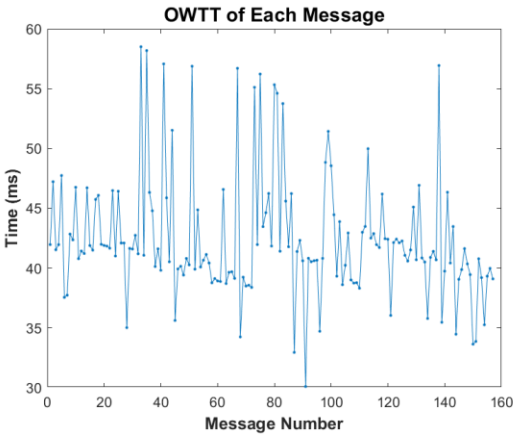


Figure 5.16: Topology of Case VI



Graphic 5.12: Messages of Case VI

Data	*Time (ms)*
Mean	42.3828
Std. Dev.	5.1275
Min	30.0965
Max	58.5035
95-Perc	0.8021
99-Perc	1.0542

Table 5.18: Results of Case VI

Considering the Figure 5.16, it is used:

- **Server/Gateway Sender and Client:** Vodafone 4G-LTE up to 150Mbps
- **Gateway Receiver:** Vodafone 4G-LTE up to 150Mbps

5.5.8 Conclusions

All the results of the tests performed were displayed in small tables along with other important values to validate its reliability. Below, in Table 5.19, are all the mean values that were being shown in previous sections.

Technologies	Time using UDP (ms)	Std. Dev.
Meo ADSL – NOWO ADSL	18.8177	2.0876
Meo Optical Fibre – Meo Optical Fibre	6.6063	1.7749
Meo ADSL – Vodafone 4G	31.6269	2.2527
Meo Optical Fibre – Vodafone 4G	30.2082	4.7054
Meo 4G – Vodafone 4G	40.2103	2.9198
Vodafone 4G – Vodafone 4G	42.3828	5.1275

Table 5.19: Comparison between the different used technologies using the IP approach

Considering the Table 3.1, where are displayed the time requirements for the messages within and to outside of the substation, some conclusions are deduced:

- With the wired technologies:
  - In the 1st and 2nd lines of the table, there are the tests using ADSL or Optical Fibre. Between using ADSL or using Optical Fibre, the difference is huge. ADSL takes almost the triple of the time to deliver one message when compared with Fibre. Despite ADSL being slower when compared with Optical Fibre, both technologies are suitable to implement and use in routable messages.
- Comparing the wired with the wireless technologies:
  - In the 3rd and 4th lines of the table, in Server/Gateway Sender are used ADSL or Optical Fibre while in Gateway Receiver is used the 4G of Vodafone. It's interesting to notice that both values are pretty similar. A possible explanation is that the time that ADSL test takes more than Optical Fibre one, is caused by the small path taken from the ISP Router to the Digital Subscriber Line Access Multiplexer (DSLAM). Thus, in the case of ADSL test, the short path is done through copper telephone lines and then, through an high speed connection to reach its core network. In the Fibre Optic one, the path from ISP router to the core network is done through Optical Fibre, which is already an high speed connection.. After that, the rest

of the path to the Gateway Receiver from the core network is the same for both cases. Again, both technologies are suitable to implement and use in routable messages.

- Comparing the wireless technologies:
    - In the 5th and 6th lines of the table, it is used the 4G-LTE in both terminations. Comparing the values, they are similar. However, it's strange that the connection between different mobile operators is a little bit faster than the one that includes the same mobile operators. The difference is although very small and these similar values can be due to:
        - First possibility is: Meo's 4G could be faster than Vodafone's 4G during the half of the handover path. The time that the Meo takes is less to reach the core network when compared with the Vodafone one and that time can be spent inside core network to execute the commute between different mobile operators.
        - Contrary, another possibility is: considering that mobile operators are equally equipped with high quality devices and with high-speed wired connections inside and between its core network, the time spent inside the core network will be the same even if the commute is done inside or between both companies. Thus, during the handover path, the variable that could affect significantly the time of delivering a message is the quality of signal provided by the base stations.
        - From the practical point of view, both values being similar, allows the electronic company during the implementation subscribes different mobile operators without compromising the quality and time delivery of the service, choosing for each place the convenient mobile operator, as proposed before. Equally to the previous analysed cases, even with the higher delivery times, using 4G to 4G still accomplishes the time requirements as wished.

In the end, all the configurations in tests performed received a green light. However, it is desirable to perform additional tests in different places and with different access points in order to turn results robust and error proof as well as to discover potential problems that may not appear during this exercise.

5.6 Tests performed in different LANs over WAN using VPN

5.6.1 Introduction

Alternatively to R-GOOSE, IEC 61850 also allows the option of using a VPN connection to create a big LAN, giving this way the possibility the packets could be send through the layer 2 using GOOSE.

In order to perform all the captures and to process the data, in this section 5.6, it is necessary to have a VPN Server and its clients connected to them. VPN Server and Server/Gateway Sender are placed in the same LAN, as shown in Figure 5.17, in order to avoid the double of transmission time that could happen, as it is shown in Figure 5.18. This allows to make the connection to the VPN Server from Server/Gateway Sender locally, giving a local IP to the OpenVPN configuration file, forcing this way not having two paths through internet. For the VPN to be a valid option and being able to compete with the IP approach, it has to be done this way, since with IP the packets only flow one time through internet. If with the VPN they would pass more than one time, the transmission time through VPN could be around the double, leaving this option way worse than the IP one.

In the end, after being presented the results for each performed tests, it will be shown the Table 5.19 and compared each technology like it was done in 5.5.8. It will also be done a comparison between using GOOSE or UDP inside the VPN tunnel.

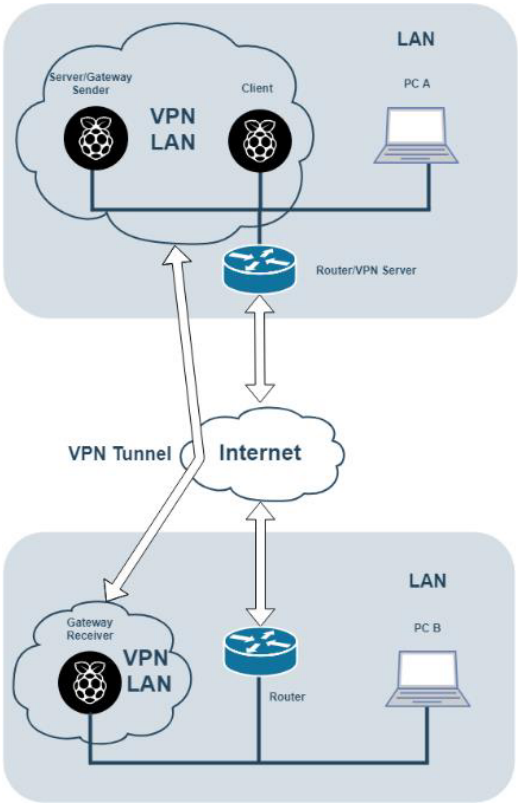


Figure 5.17: VPN Server at Server/Gateway Sender's side

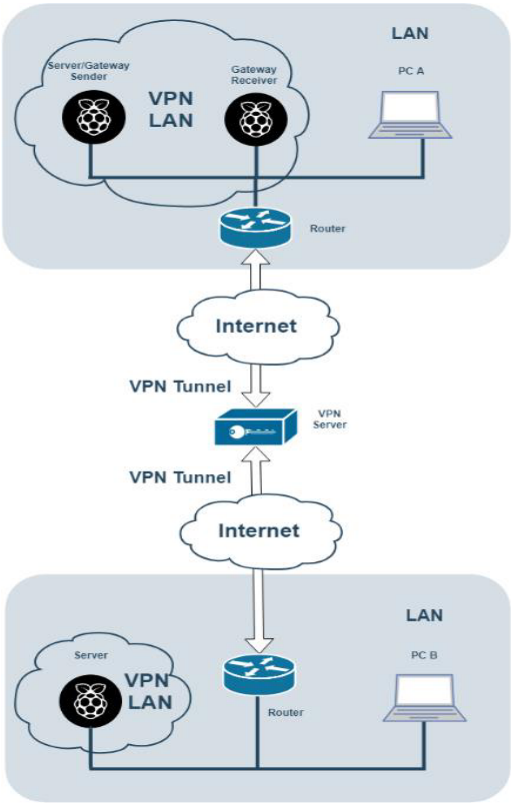


Figure 5.18: VPN Server in another Internet Access

5.6.2 Case I: Meo Optical Fibre – Meo Optical Fibre

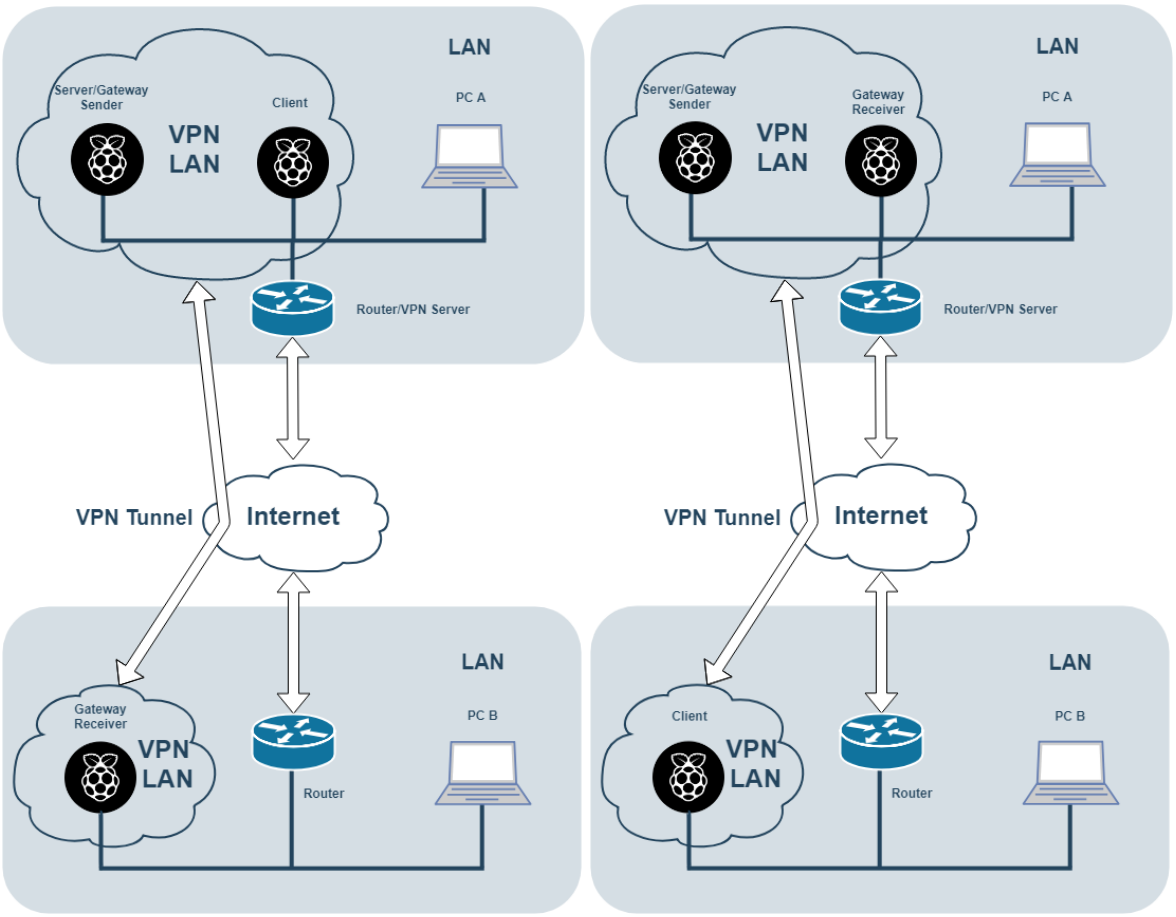
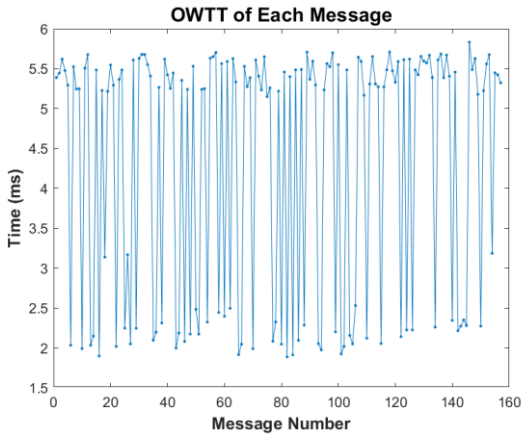


Figure 5.19: Topology of of Case I

5.6.2.1 GOOSE – 1

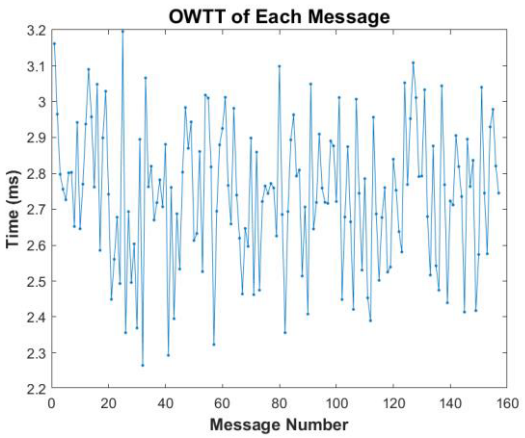


Graphic 5.13: Messages of Case I-1

Data	*Time (ms)*
Mean	4.3279
Std. Dev.	1.5718
Min	1.8865
Max	5.8325
95-Perc	0.2459
99-Perc	0.3231

Table 5.20: Results of Case I-1

5.6.2.2 UDP/IP – 2



Graphic 5.14: Messages of Case I-2

Data	*Time (ms)*
Mean	2.7423
Std. Dev.	0.1992
Min	2.2650
Max	3.1955
95-Perc	0.0312
99-Perc	0.0410

Table 5.21: Results of Case I-2

5.6.2.3 Considerations

Considering the Figure 5.19, it is used in each configuration:

- Configuration GOOSE:
  - **Server/Gateway Sender and Gateway Receiver:** Meo Optical Fibre 100Mbps
  - **Client:** Meo Optical Fibre 100Mbps
- Configuration UDP:
  - **Server/Gateway Sender and Client:** Meo Optical Fibre 100Mbps
  - **Gateway Receiver:** Meo Optical Fibre 100Mbps

5.6.3 Case II: Meo ADSL – Vodafone 4G

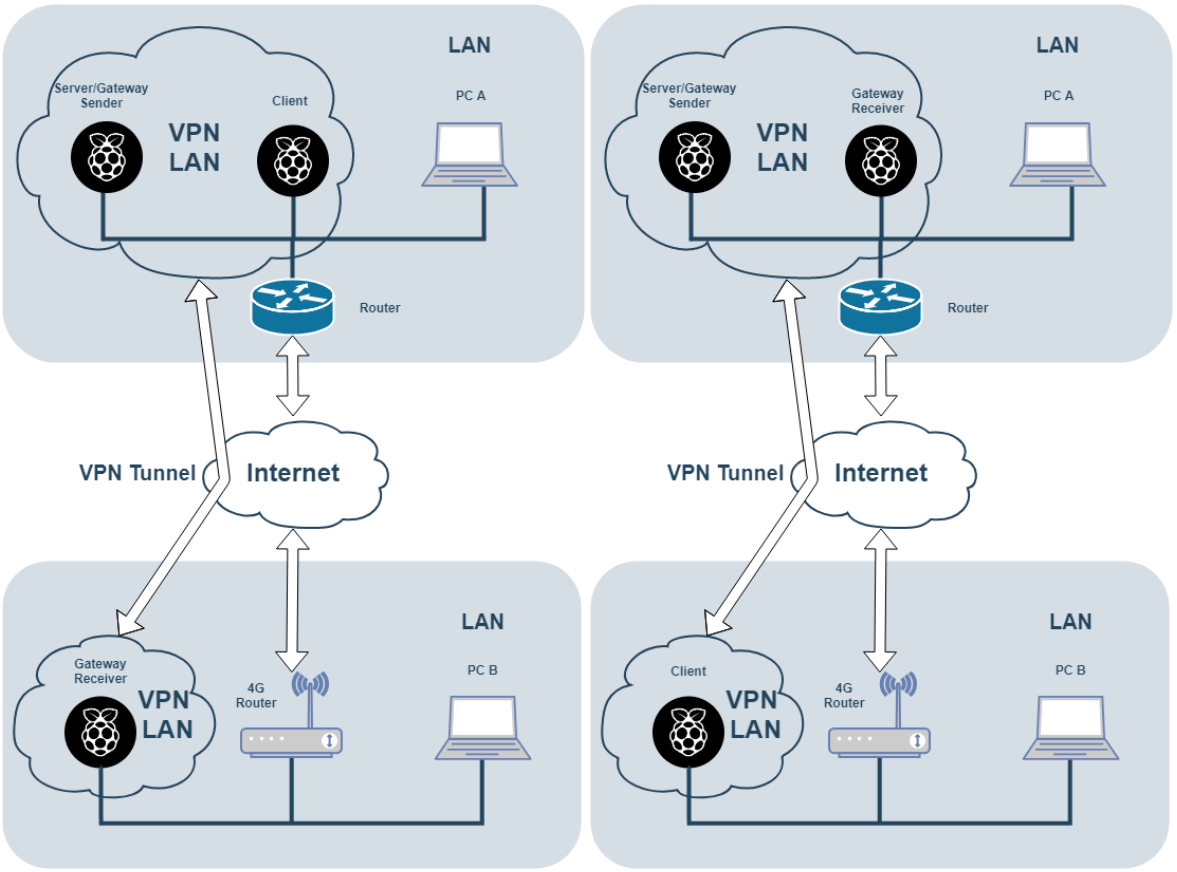
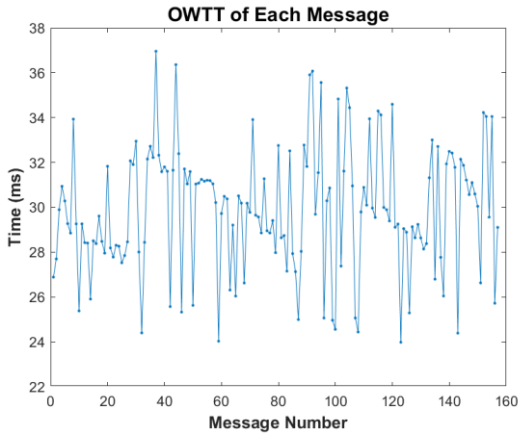


Figure 5.20: Topology of Case II, III and IV

5.6.3.1 GOOSE – 1

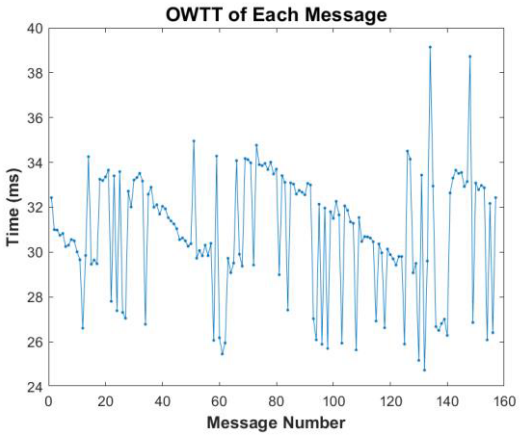


Graphic 5.15: Messages of of Case II-1

Data	*Time (ms)*
Mean	29.8900
Std. Dev.	2.7947
Min	23.9830
Max	36.9570
95-Perc	0.4372
99-Perc	0.5746

Table 5.22: Results of Case II-1

5.6.3.2 UDP/IP – 2



Graphic 5.16: Messages of Case II-2

Data	*Time (ms)*
Mean	30.8764
Std. Dev.	2.7460
Min	24.7380
Max	39.1415
95-Perc	0.4296
99-Perc	0.5646

Table 5.23: Results of Case II-2

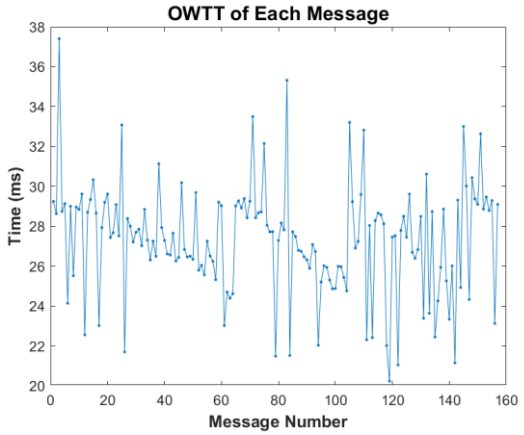
5.6.3.3 Considerations

Considering the Figure 5.20, it is used in each configuration:

- Configuration GOOSE:
  - **Server/Gateway Sender and Gateway Receiver:** Meo ADSL 12Mbps
  - **Client:** Vodafone 4G-LTE up to 150Mbps
- Configuration UDP:
  - **Server/Gateway Sender and Client:** Meo ADSL 12Mbps
  - **Gateway Receiver:** Vodafone 4G-LTE up to 150Mbps

5.6.4 Case III: Meo Optical Fibre – Vodafone 4G

5.6.4.1 GOOSE – 1

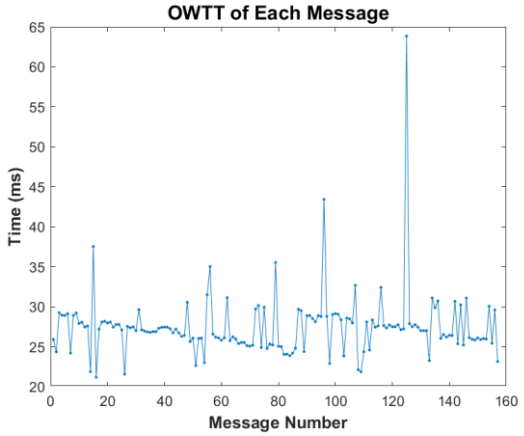


Graphic 5.17: Messages of Case III-1

Data	*Time (ms)*
Mean	27.3249
Std. Dev.	2.7715
Min	20.2295
Max	37.4085
95-Perc	0.4335
99-Perc	0.5698

Table 5.24: Results of Case III-1

5.6.4.2 UDP/IP – 2



Graphic 5.18: Messages of Case III-2

Data	*Time (ms)*
Mean	27.4851
Std. Dev.	4.0475
Min	21.1650
Max	63.8555
95-Perc	0.6331
99-Perc	0.8321

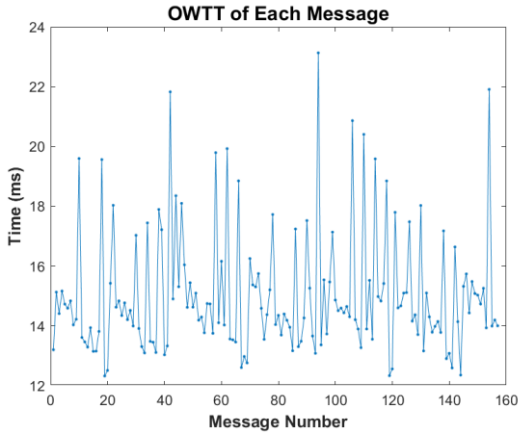
Table 5.25: Results of Case III-2

Considering again the Figure 5.20, it is used in each configuration:

- Configuration GOOSE:
  - **Server/Gateway Sender and Gateway Receiver:** Meo Optical Fibre 100Mbps
  - **Client:** Vodafone 4G-LTE up to 150Mbps
- Configuration UDP:
  - **Server/Gateway Sender and Client:** Meo Optical Fibre 100Mbps
  - **Gateway Receiver:** Vodafone 4G-LTE up to 150Mbps

5.6.5 Case IV: Meo Optical Fibre – Meo 4G

5.6.5.1 GOOSE – 1

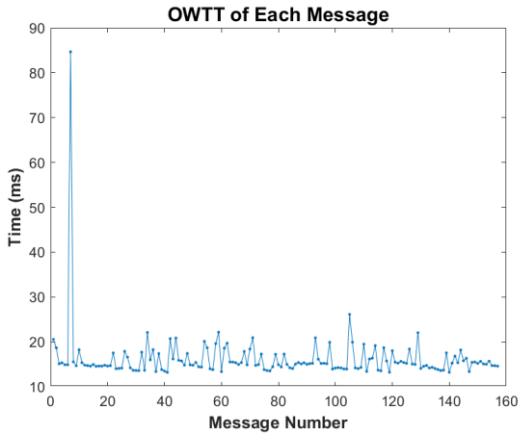


Graphic 5.19: Messages of Case IV-1

Data	*Time (ms)*
Mean	15.0086
Std. Dev.	2.0466
Min	12.3245
Max	23.1305
95-Perc	0.3201
99-Perc	0.4208

Table 5.26: Results of Case IV-1

5.6.5.2 UDP/IP – 2



Graphic 5.20: Messages of of Case IV-2

Data	*Time (ms)*
Mean	16.2015
Std. Dev.	5.9332
Min	13.1900
Max	84.6555
95-Perc	0.9281
99-Perc	1.2198

Table 5.27: Results of Case IV-2

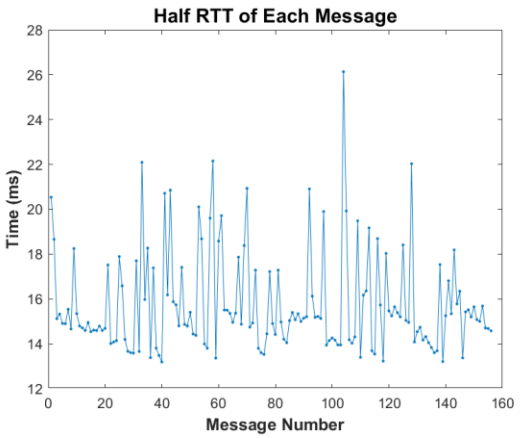
5.6.5.3 Considerations

Considering again the Figure 5.20, it is used in each configuration:

- Configuration GOOSE:
  - **Server/Gateway Sender and Gateway Receiver:** Meo Optical Fibre 100Mbps
  - **Client:** Meo 4G-LTE up to 150Mbps
- Configuration UDP:
  - **Server/Gateway Sender and Client:** Meo Optical Fibre 100Mbps
  - **Gateway Receiver:** Meo 4G-LTE up to 150Mbps

This configuration was not performed with IP and the explanation will be in 5.7 when the comparation is made.

5.6.5.4 UDP without abnormal value



Graphic 5.21: Messages of Case IV-2 corrected

Data	*Time (ms)*
Mean	15.7627
Std. Dev.	2.2371
Min	13.1900
Max	26.1325
95-Perc	0.3511
99-Perc	0.4614

Table 5.28: Results of Case IV-2 corrected

Likewise it was done in section 5.5.6, the value that fled a lot from the average was removed. Again, it is calculated the new values shown in Table 5.28 as well as it is generated the Graphic 5.21, allowing a close look to the values of each message.

After analysing the capture, it wasn't found any justification for this value, however it is important to notice and be aware of delays like this can happen in a future implementation.

5.6.6 Case V: Vodafone 4G – Meo 4G

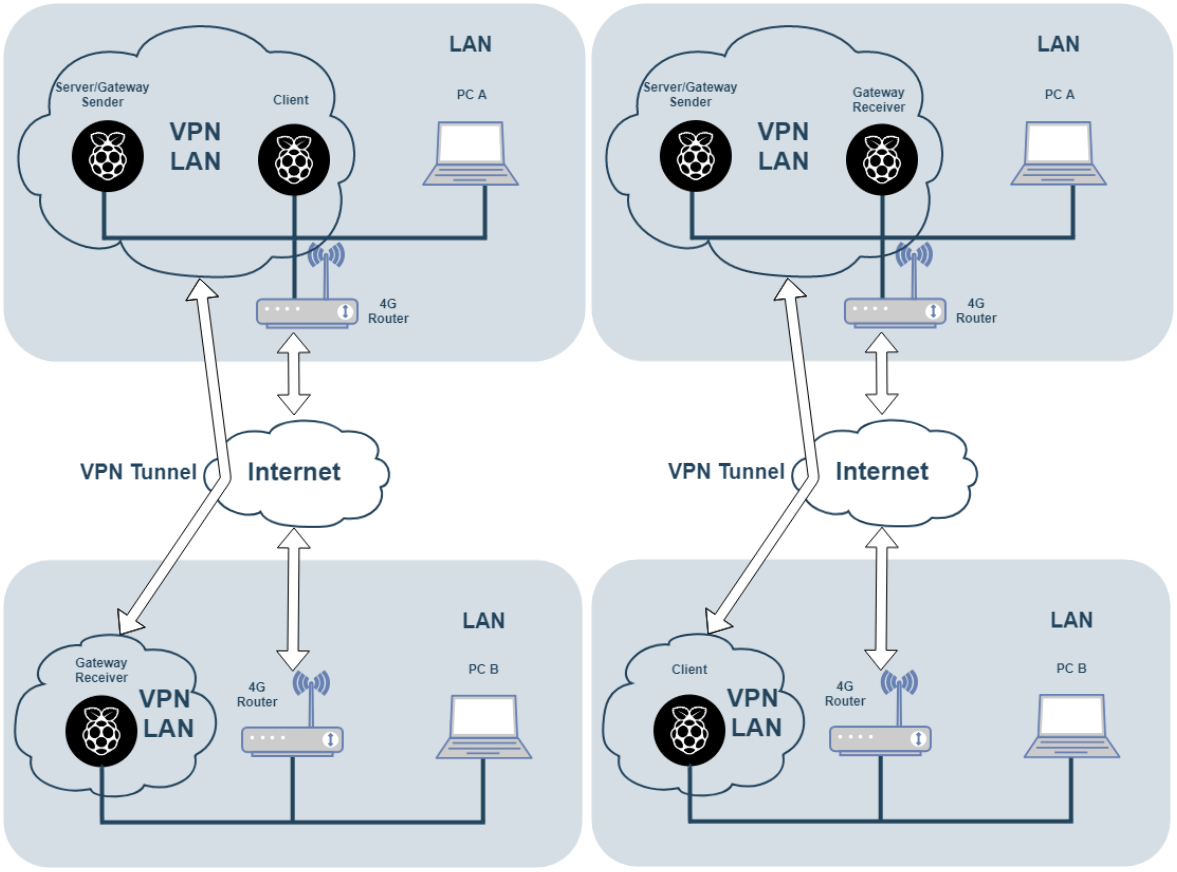
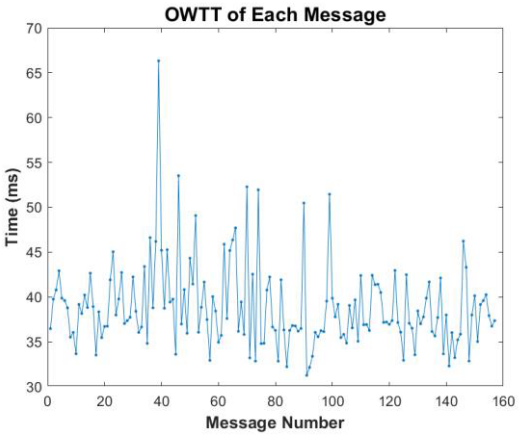


Figure 5.21: Topology of Case V and VI

5.6.6.1 GOOSE – 1

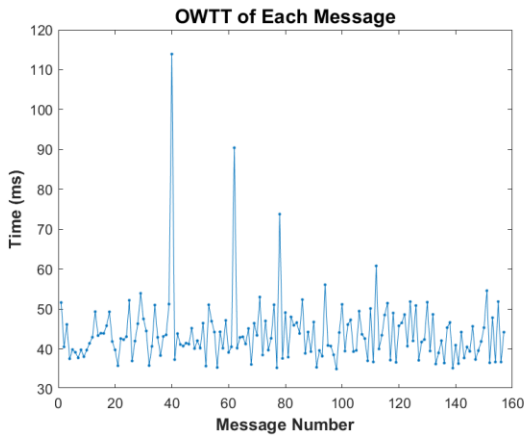


Graphic 5.22: Messages of Case V-1

Data	*Time (ms)*
Mean	38.9085
Std. Dev.	4.7840
Min	31.2635
Max	66.3185
95-Perc	0.7483
99-Perc	0.9835

Table 5.29: Results of Case V-1

5.6.6.2 UDP/IP – 2



Graphic 5.23: Messages of Case V-2

Data	*Time (ms)*
Mean	43.9785
Std. Dev.	8.8204
Min	34.9565
Max	113.9270
95-Perc	1.3797
99-Perc	1.8134

Table 5.30: Results of Case V-2

5.6.6.3 Considerations

Considering the Figure 5.21, it is used in each configuration:

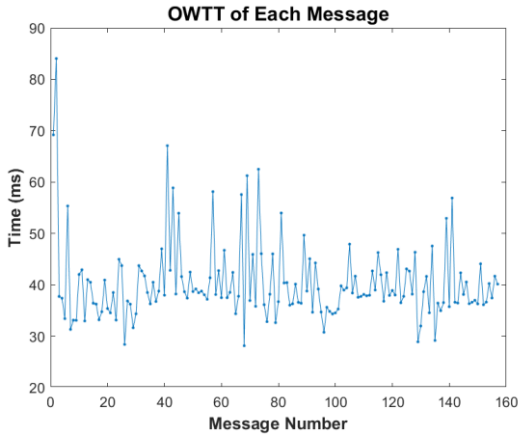
- Configuration GOOSE:
  - **Server/Gateway Sender and Gateway Receiver:** Meo ADSL 12Mbps
  - **Client:** Vodafone 4G-LTE up to 150Mbps
- Configuration UDP:
  - **Server/Gateway Sender and Client:** Meo ADSL 12Mbps
  - **Gateway Receiver:** Vodafone 4G-LTE up to 150Mbps

There is a particular nuance performed in this test when compared with the one performed in 5.5.7. Here in Server/Gateway Sender, it is used Vodafone 4G instead of Meo 4G and in Client or Gateway Receiver is used Meo 4G instead of Vodafone 4G.

Basically, both mobile operators switched places between them to perform both tests because, in 5.5.7, the port forwarding is performed in Gateway Receiver to receive UDP messages, while here in 5.6.6, it is performed to VPN Server to receive VPN data. During the tests, it wasn't possible to send data over WAN IP and forward it through the chosen port when using Meo, most probably due to the ports being blocked, even when used nonstandard ones. So, using Vodafone that doesn't block ports, the solution found is to switch mobile operators in both terminations since the path is expected to be the same, not compromising the shape of path itself.

5.6.7 Case VI: Vodafone 4G – Vodafone 4G

5.6.7.1 GOOSE – 1

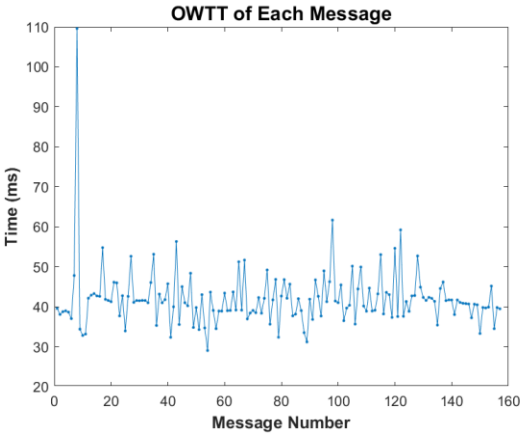


Graphic 5.24: Messages of Case VI-1

Data	*Time (ms)*
Mean	40.2960
Std. Dev.	7.7483
Min	28.1560
Max	84.0485
95-Perc	1.2120
99-Perc	1.5930

Table 5.31:Results of Case VI-1

5.6.7.2 UDP/IP – 2



Graphic 5.25: Messages of Case VI-2

Data	*Time (ms)*
Mean	41.9535
Std. Dev.	7.5785
Min	29.0245
Max	109.5905
95-Perc	1.1855
99-Perc	1.5580

Table 5.32: Results of Case VI-2

5.6.7.3 Considerations

Considering again the Figure 5.21, it is used in each configuration:

- Configuration GOOSE:
  - **Server/Gateway Sender and Gateway Receiver:** Meo ADSL 12Mbps
  - **Client:** Vodafone 4G-LTE up to 150Mbps
- Configuration UDP:
  - **Server/Gateway Sender and Client:** Meo ADSL 12Mbps
  - **Gateway Receiver:** Vodafone 4G-LTE up to 150Mbps

5.6.8 Conclusions

Here, like it was done in section 5.5.8, Table 5.33 shows a comparison between the technologies used. Additionally, since it is used UDP/IP over VPN tunnel, it is also done a comparison between it and GOOSE.

Bellow, there are the Table 5.33 that contains all the mean values that were being shown:

Technologies	**Time using GOOSE (ms)**	**Time using IP (ms)**
Meo Optical Fibre – Meo Optical Fibre	4.3279	2.7423
Meo ADSL – Vodafone 4G	29.8900	30.8764
Meo Optical Fibre –Vodafone 4G	27.3249	27.4851
Meo Optical Fibre – Meo 4G	15.0086	15.7627
Vodafone 4G – Meo 4G	38.9085	43.9785
Vodafone 4G – Vodafone 4G	40.2960	41.9535

Table 5.33: Comparison between the different used technologies using the VPN approach

Considering again the Table 3.1, where are displayed the time requirements for the messages within and to outside of the substation, some conclusions are deduced:

- Starting with both methods, GOOSE and UDP inside the VLAN connection:
    - Not considering the first test where Optical Fibre is used in both terminations, using UDP method is slightly slower than using GOOSE, in general. The first test may be an exception when compared with the other ones and additional tests should be performed to find out if the results using Optical Fibre are trustable or not. It's strange that in Optical Fibre using UDP, inside a VLAN connection, is faster than using GOOSE. Independently from the technology used, it is expected that using GOOSE would be faster than UDP, since GOOSE is a layer 2 protocol while UDP is a layer 3 one. Additionally, UDP messages are a little bit bigger than the GOOSE ones, which could explain the small difference between both methods. On average, it is spent around 1/2 ms more using UDP and in the end, both methods are suitable to send messages to outside of the substation.
- Henceforth, the focus will be on GOOSE method and looking to the Optical Fibre:
    - Here, contrary to what was done before, ADSL test were not performed. In section 5.5, using UDP is concluded that using Optical Fibre would be much better than using ADSL. In this case, using Optical Fibre is the best and fastest result so far.

- Next, it was performed three tests between a wired and wireless connection:
    - From these three tests there is a lot of information that can be withdrawn. Starting with the 2nd and 3rd lines of the table, it is used the same mobile operator, Vodafone 4G, in the Client's termination, leaving the Meo ADSL or Meo Optical Fibre for the Server/Gateway Sender. Comparing both, Optical Fibre has a slightly better value than using ADSL. Again, like in 5.5.8, the difference is minimal and can be explained due to the same reasons, which is: the difference between both is just the path that takes from ISP Router to the DSLAM.
    - Now, comparing the 3rd and 4th lines of the table, in both cases is used Meo Optical Fibre in Server/Gateway Sender and at Client's side, it is used 4G of Meo and 4G of Vodafone respectively. Here, there is a major difference between both values. Using Meo's 4G is almost two times faster than using Vodafone's 4G. This can be explained due to the mobile operator and optical fibre belong to the same company, Meo, while in the other one that takes more time, the test is performed between different companies. So, inside the core network, the time that takes to exchange between a landline and mobile connection of different network companies is bigger than the exchange that it would be if they were the same.
    - Making a final recap, all of three options are suitable however, it is desirable that the connection between a landline and a mobile network can be done using the same network operator. It is also concluded that using ADSL in conjunction with a mobile operator doesn't compromise the good functioning of the system since the ADSL is just used in a small part of the path that messages take.
- Finishing with the comparison of wireless technologies:
    - In 5th and 6th lines of the table, it is used the 4G-LTE in both terminations. At Sever/Gateway Sender, it is used in both cases Vodafone's 4G and at the Client's side it is used Meo's and Vodafone's 4G, respectively. Again, strangely, using different mobile operators at terminations is slightly faster than using the same one. This small difference between both can be explained with same reason as described in 5.5.8. Again, since the time that takes to deliver the messages is almost the same in both cases, it allows the electric company with both options when it comes the real implementation. Over again, even if they are the worst case in terms of latency times, they still accomplish the time requirements for a real implementation.

Then again, all the configurations in performed tests are suitable to implement and, as said before, additional tests in different conditions are desirable to perform to rule out possible errors and turn the results error proof.

5.7 Comparing IP vs VPN

Finally, and more important, it is made an overview about latency times and a comparison about technologies used in both communication mechanisms proposed by the standard and explained in 3.5.

Bellow, there is the Table 5.34 that contains all the mean values of the Table 5.19 and the Table 5.33:

Technologies	**Time using IP (ms)**	**Time using VPN (ms)**
Meo ADSL – NOWO ADSL	18.8177	-
Meo Optical Fibre – Meo Optical Fibre	6.6063	4.3279
Meo ADSL – Vodafone 4G	31.6269	29.8900
Meo Optical Fibre – Vodafone 4G	30.2082	27.3249
Meo Optical Fibre – Meo 4G	-	15.0086
Meo 4G – Vodafone 4G / Vodafone 4G – Meo 4G	40.2103	38.9085
Vodafone 4G – Vodafone 4G	42.3828	40.2960

Table 5.34: Comparison between the IP and VPN approach

Passing all the lines of the Table 5.34 and comparing the time values of both methods:

- In the 1st line, there isn't a comparation term to make between using IP or VPN, since it's not expectable that ADSL will be used in both terminations. Nevertheless, it is a good value that already was used to compare with Optical Fibre.
- In the 2nd line, using VPN approach, it is about 2 ms faster than using the IP one. It's important keep in mind this value since it will be similar over the table when compared both methods.
- In the 3rd line, same time difference as before, around 2 ms.
- In the 4th line, the difference increased to around 3 ms, with the VPN approach being faster than IP one.
- In the 5th line, again, there isn't a comparation term to make between using IP or VPN, because, similar to what happened in 5.6.6. Meo's 4G didn't allow to send data over IP through the chosen port, 1010, even not being a standardized one. One possibility would be to exchange the ISPs in both terminations, although the Server/Gateway Sender usually stays with landline connection, since it produces more

traffic when compared with its clients. This exchange is something that could be done, however it's not a priority, since it was possible to perform the test using the Vodafone's 4G.

- In the 6th line, it is performed the exchange of ISPs when compared both tests. Here, it is more important to make the exchange between ISPs and to perform both tests since in both connections it is used 4G. So, the question about flowing more traffic in the side where the wireless connection is, does not apply. In the end, comparing both values, even with this exchange, the time difference between both methods still verifies, resulting in a little more than 2 ms, which also allows us to conclude that exchanging the ISPs in terminations is a good procedure to work around this problem.
- In the 7th line, there is the same time difference as observed before.

It stands out the constant difference between both communication mechanisms, where using VPN is, on average, 2 ms faster than using IP. So, to summarize everything:

- Between IP or VPN:
    - Using VPN is faster, although both accomplish the time requirements and the difference is minimal.
    - Using IP is better when the number of devices connected to the same working area is higher, since it will avoid the flood of the unwanted packets.
    - Using IP requires more work in terms of software engineering and using VPN would require additional and more powerful network devices as well as additional VPNs configurations.

Depending on the application case, either cases are good choices.

6. Conclusions and Future Work

6.1 Conclusions

Initially, in Chapter 1, it was introduced the questions that are necessary to be answered. In chapter 2, it was described all the network and other protocols necessary to perform the proper modifications in the practical part. In chapter 3, it was analysed the standard IEC 61850 itself, allowing to work deeply with the chosen library. In chapter 4, it was analysed relevant data as well as performed the system improvements used in the chapter 5. In chapter 5, it was finally captured the wished results and comparing them. So, looking back, here in chapter 6 almost everything was reached.

From the economical point of view, in a tough market such as the ICT one, there is a lot of information in this master dissertation that can be used to develop a complete product that can generate revenue in the order of millions since it's a large product that can equip an entire country. Additionally, when the development work is concluded, it can be sold internationally, since the adjustments to the other countries will be minimal for those that are similarly developed.

Now, we are in conditions to answer the principal question: Is 4G-LTE suitable to use with R-GOOSE and R-SV? Yes, it is and, more important, with countries with a similar or better mobile infrastructure it may be also suitable, allowing, as said before, a lot of business opportunities to achieve. Additionally, it is also concludable that other important technologies such as ADSL or Optical Fibre can be used to send messages over a WAN, providing flexibility in terms of implementation, having this way additional solutions.

The wide range of results shows what was expected. Wired connections are more steady, fast and reliable while wireless ones are more susceptible to errors. In the end, all of them achieve the time requirements but the mobile connections are the ones that have a large margin to improve in terms of stability, latency and bandwidth. The 5G is at around the corner and will bring solutions for these problems.

Looking back, it was a long road to get here and achieve these results. Hopefully, they can be used to fight the scarcity of energy in some countries as well as turn the electricity grid more complete in the others, making this way our world a little bit better.

6.2 Future Work

In this last topic, it will be presented what should be done next and I'll take this opportunity to leave a few notes that I would like to include.

Here, there are several lines that could be taken in order to continue this work:

- First, in terms of transportation, validate and re-validate in all different kind of scenarios, investigate what is the influence of having different kinds of electrical lines close to the Optical Fibre or 4G-LTE transmitter, the influence of having a large number of connected devices to the same base station and the influence of the weather conditions. At least, all of these things have to be analysed.
- Second, in terms of negotiation, if it is possible with ISPs to provide an high priority connection when compared to the normal users to avoid the situations like described above. If it is possible to achieve better bandwidth and latency times as well as, if it is possible to install additional infrastructures to serve our proposes.
- Third, in terms of application, it is necessary to understand the needs of the electrical company and design the application in accordance with its needs. So, here, there are a long path to go through.

Leaving some notes and considering what was done, the transportation part is the one that makes more sense to continue, however it would be important that the further tests would be performed with an robust and complete application, which in this case, doesn't exist. Also, considering the way that this dissertation took, R-SV were left a little bit aside, but measuring the messages transmission time was the main goal, leaving the content of themselves in second plan.

Also, in my opinion, a good long term investment would be using IP. I would make this choice because, looking to the values of Optical Fibre and looking again to the Table 3.1, the difference between them starts to be short. In the results, using Optical Fibre, we got 6 ms in tests performed in a WAN and in the Table 3.1 shows that it is needed 3 ms in a LAN.

As far as I understand, the research shows that with the emergence of 5G and with latency times around 1 ms, it makes almost no sense to separate between LAN and WAN traffic. Of course, there are situations that don't justify having all the time traffic flowing through a WAN but, probably, in the future, the quality of the connections will be even more secure, will support even lower latency times and will be able to handle the overflood of the unnecessary packets easily.

7. Bibliography

[1] E. H. E. Bayoumi, "Power electronics in smart grid consumption systems: a review," in *International Journal of Industrial Electronics and Drives*, Abu Dhabi, UAE, 2017.

[2] A. Ashour, "Modelling of Smart Auto-Recloser with Over Current Protection," *Int. Journal of Engineering Research and Application,* vol. 8, no. 7, pp. 1-5, 2018.

[3] M. M. Morteza Shabanzadeh, "What is the Smart Grid? Definitions, Perspectives, and Ultimate Goals," in *International Power System Conference*, Tehran, Iran, 2013.

[4] T. C. H. L. R. Z. Linwei Chen, "Virtual site acceptance test platform for IEC 61850 based substations with multi-vendor bay solutions," in *The 14th International Conference on Developments in Power System Protection,* 2018.

[5] IEC, "IEC 61850-90-5 TR Ed.1: Communication networks and systems for power utility automation – Part 90-5: Use of IEC 61850 to transmit synchrophasor information according to IEEE C37.118," Germany, 2011.

[6] S. AR, "Substation Automation system," *International Journal of Scientific & Engineering Research*, vol. 7, no. 5, 2016.

[7] E. Padilla, Substation Automation Systems: Design and Implementation, 2015.

[8] A. P. David Hewings, "Structured approach to maintenance and modification of a railway traction IEC 61850 substation automation system," in *The 14th International Conference on Developments in Power System*, 2018.

[9] S. K. a. D. S. O. D. R. Gurusinghe, "Testing of IEC 61850 sampled values based digital substation automation systems," in *The 14th International Conference on Developments in Power System Protection*, 2018.

[10] P. C. Mohamed Hosny Tawfeek Essa, “GOOSE performance assessment on an IEC 61850 redundant network,” in *The 14th International Conference on Developments in Power System Protection*, 2018.

[11] B. X. Z. Z. Y. C. C. B. Tony Yip, “Application of IEC 61850 for distribution network automation with distributed control,” in *The 14th International Conference on Developments in Power System Protection* , 2018.

[12] K. Schwarz, "NIST recommends IEC 61850 and other IEC TC57 Standards for Regulation," 2010. [Online]. Available: https://www.nettedautomation.com/download/Newsletter/IEC61850-Blog_until_2012-01-27.pdf. [Accessed 09 2018].

[13] J. E. L. P. E. C. S. a. R. M. N. Giovanni Manassero, "IEC61850–Based Systems—Functional Testing and Interoperability Issues," *IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, VOL. 9, NO. 3, AUGUST 2013*, 2013.

[14] D. H. E. (. A. N. G. Henry Dawidczak, "Integration of DER systems into the Electrical Power System with a generic IEC 61850 interface," in *Internationaler ETG-Kongress 2013*, 2013.

[15] R. E. Mackiewicz, "Overview of IEC 61850 and Benefits," IEEE, 2006.

[16] IEC, "IEC 61850 Part 7-2: Basic communication structure for substation and feeder equipment - Abstract communication service interface (ASCI)," Chapter 3, https://www.sis.se/api/document/preview/563012/.

[17] V. V. N.-K. N. a. K. S. Neil Higgins, “Concept for intelligent distributed power system automation with IEC 61850 and IEC 61499,” in *IEEE International Conference on Systems, Man and Cybernetics*, Singapore, Singapore, 2008.

[18] S. Mohagheghi, "A fuzzy cognitive map for data integrity assessment in a IEC 61850 based substation," in *IEEE PES General Meeting*, Providence, RI, USA, 2010.

[19] L. L. Bello, "MMS - Manufacturing Message Specifications," [Online]. Available: http://www.diit.unict.it/users/llobello/reti/lezione5.pdf. [Accessed 09 2018].

[20] A. Giasis, "EVALUATION OF THE IEC 61850 COMMUNICATION SOLUTIONS," Vaasa, Finland, 2016.

[21] S. R. Firouzi, "Design, Implementation and Validation of an IEC 61850-90-5 Gateway for IEEE C37.118.2 Synchrophasor Data Transfer," Barcelona, Spain, 2015.

[22] K. Sachintha, “Implementation of an IEC 61850 Sampled Values Based Line Protection IED with a New Transients-Based Hybrid Protection Algorithm,” Winnipeg, MB, Canada, 2016.

[23] K. R. Jim Kurose, Computer Networking 7th Ed, 2016.

[24] D. W. Andrew Tanenbaum, Computer Networks 5th Ed, 2012.

[25] B. Mitchell, "The Layers of the OSI Model Illustrated," 07 2018. [Online]. Available: https://www.lifewire.com/layers-of-the-osi-model-illustrated-818017.

[26] V. Beal, "The 7 Layers of the OSI Model," [Online]. Available: https://www.webopedia.com/quick_ref/OSI_Layers.asp.

[27] U. I. Ikechukwu, "A Survey on Bandwidth Management Techniques Via the OSI Model Network Application Layers," *Global Journal of Computer Science and Technology: ENetwork,Web & Security,* vol. 17, no. 4.

[28] Wireshark, "Virtual Bridged LAN (VLAN, IEEE 802.1Q)," [Online]. Available: https://wiki.wireshark.org/VLAN. [Accessed 10 2018].

[29] F. G. L. F. V. A. P. S. C. Vincenzo Eramo, "Simulation and Experimental Evaluation of a FlexibleTime Triggered Ethernet Architecture Applied inSatellite Nano/Micro Launchers," *MDPI,* 2018.

[30] E. G. P. T. N. F. D. F. Glenn Parsons, "Media Access Control Bridges and Virtual Bridged Local Area Networks," [Online]. Available: http://www6.ietf.org/meeting/86/tutorials/86-IEEE-8021-Thaler.pdf.

[31] IEC, "Communication networks and systems for power utility automation - Part 90-1: Use of IEC 61850 for the communication between substations," Germany, 2009.

[32] N. Unuth, "What IP Means and How It Works," [Online]. Available: https://www.lifewire.com/internet-protocol-explained-3426713.

[33] Z. S. a. A. B. Saad Saleh, “Capacity Analysis of Combined IPTV and VoIP Over IEEE 802.11n,” Islamabad, Pakistan, 2013.

[34] Y. S. O. Y. J. W. a. J. L. Kai Shi, "Improving TCP Performance for EAST Experimental Data in the Wireless LANs," in *IEEE TRANSACTIONS ON NUCLEAR SCIENCE, VOL. 58, NO. 4*, 2011.

[35] S. P. D. H. Abdullah Alshalan, "A Survey of Mobile VPN Technologies," in *IEEE Communications Surveys & Tutorials, Vol. 18, No. 2,* 2016.

[36] B. Mitchell, "What a VPN Can Do for You," [Online]. Available: https://www.lifewire.com/what-a-vpn-can-do-for-you-818176. [Accessed 10 2018].

[37] OpenVPN. [Online]. Available: https://openvpn.net/. [Accessed 2018 10].

[38] X. C. Y. C. a. D. W. Yuke Li, "Smart Choice for the Smart Grid: Narrowband Internet of Things (NB-IoT)," *IEEE INTERNET OF THINGS JOURNAL*, vol. 5, 2018.

[39] I. Bartolic, "Disadvantages of WiMAX Technology. Is the WiMAX the right choice?," [Online]. Available: http://thebestwirelessinternet.com/disadvantages-of-wimax.html. [Accessed 08 2018].

[40] M. J. A. S. a. B. N. Matthew Weiner, "Design of a low-latency, high-reliability wireless communication system for control applications," in *IEEE ICC 2014*, Sydney, Australia, 2014.

[41] "The wireless era," [Online]. Available: http://www.wirelesscommunication.nl/reference/chaptr07/history.htm. [Accessed 06 2018].

[42] M. L. J. Vora, "Evolution of Mobile Generation Technology: 1G to 4G and Review of Upcoming Wireless Technology 5G," [Online]. Available: https://pdfs.semanticscholar.org/4dfd/40cc3a386573ee861c5329ab4c6711210819.pdf.

[43] D. P. R. B. Anil Kumar, "Overview of Advances in Communication Technologies," in *13th International Conference on Electromagnetic Interference and Compatibility*, Visakhapatnam, India, 2015.

[44] M. L. Roberts, M. A. Temple, R. F. Mills and R. A. Raines, "Evolution of the air interface of cellular communications systems toward 4G realization," *IEEE Communications Surveys & Tutorials*, vol. 1, pp. 2-23.

[45] "The State of LTE (February 2018)," [Online]. Available: https://opensignal.com/reports/2018/02/state-of-lte. [Accessed 08 2018].

[46] "Mobile Technology Statistics - Global," [Online]. Available: http://www.5gamericas.org/en/resources/statistics/statistics-global/. [Accessed 05 2018].

[47] T. G. L. I. D. L. A. P. R. I. D. A. N. H. a. P. Stanley Chia, "3G evolution," *EEE Microwave Magazine*, vol. 4, pp. 52-63, 2008.

[48] T. E. B. a. L. B. Le, "Massive MIMO and mmWave for 5G Wireless HetNet: Potential Benefits and Challenges," *IEEE Vehicular Technology Magazine*, vol. 11, pp. 64-75, 2016.

[49] "LTE to 5G: Cellular and Broadband Innovation," [Online]. Available: http://www.5gamericas.org/files/1915/0282/6623/LTE_to_5G_Cellular_and_Broadband_Innovation_-_Rysavy_for_upload.pdf. [Accessed 06 2018].

[50] Vodafone, "Our contribution, country by country," [Online]. Available: https://www.vodafone.com/content/sustainabilityreport/2015/index/operating-responsibly/tax/our-contribution-country-by-country.html. [Accessed 10 2018].

[51] V. Portugal, "Principais Marcos da História da Vodafone Portugal," [Online]. Available: https://www.vodafone.pt/NR/rdonlyres/ABB9A0A9-6E3D-4E2A-A8BB-8B9E7324E58B/0/080128MilestonesdaVodafonePortugal.pdf. [Accessed 08 2018].

[52] A. Labs, "Uma empresa histórica," [Online]. Available: https://www.telecom.pt/pt-pt/a-pt/Paginas/historia.aspx. [Accessed 10 2018].

[53] NOS, "História," [Online]. Available: http://www.nos.pt/institucional/PT/sobre-a-nos/quem-somos/Paginas/a-nossa-historia.aspx. [Accessed 10 2018].

[54] A. Brito, "A nova Cabovisão quer ser low cost e à la carte," *https://www.publico.pt/2016/09/13/economia/noticia/a-nova-cabovisao-quer-ser-low-cost-e-a-la-carte-1743988*.

[55] ANACOM, "Frequências ANACOM," [Online]. Available: https://www.anacom.pt/render.jsp?categoryId=382989. [Accessed 08 2018].

[56] nperf, "Mapa de cobertura 2G / 3G / 4G, Portugal," [Online]. Available: https://www.nperf.com/pt/map/PT/-/-/signal/. [Accessed 10 2018].

[57] "Mapa de Portugal com a localização das antenas de telemóveis.," [Online]. Available: http://www.gsmhouse.pt/index.php?showtopic=35197. [Accessed 10 2018].

[58] U. G. V. T. Pallab Dutta, "A Comprehensive Review of Embedded System Design Aspects for Rural Application Platform," *International Journal of Computer Applications*, vol. 106, 2014.

[59] V. K. A. L. Suseela, "Embedded Systems in Real Time Appplication, Design and Architecture," [Online]. Available: https://ubiquity.acm.org/article.cfm?id=1086450. [Accessed 08 2018].

[60] J. M. Garrido, "Improving Software Development for Embedded Systems," in *SouthEast Conference*, 2017.

[61] A. Apostolov, "R-GOOSE: what it is and its application in distribution automation," in *24th International Conference & Exhibition on Electricity Distribution*, 2017.

[62] IEC, "IEC 61850-5: Communication requirements for functions and device models," 2007.

[63] S.-C. H. C.-K. C. M.-S. L. Tzu-Han Yeh, "Conformance Test for IEDs Based on IEC 61850 Communication Protocol," in *Journal of Power and Energy Engineering*, Taipei, Taiwan, 2015.

[64] C. M. M. S. a. F. V. H. León, "Simulation Models for IEC 61850 Communication in ElectricalSubstations Using GOOSE and SMV Time-critical Messages," in *EEE World Conference on Factory Communication Systems (WFCS)*, Aveiro, 2016.

[65] S. B. a. O. K. Charles M. Adrah, "Fusion networking technology for IEC 61850 inter substation communication," in *2017 IEEE International Conference on Smart Grid and Smart Cities*, Singapore, 2017.

[66] T. T. D. L. a. M. P. G. Mamais, "An ASN.1 compiler for embedded/space systems," in *ERTS*, 2012.

[67] M. Zillgith. [Online]. Available: http://libiec61850.com/libiec61850/.

[68] EFACEC. [Online]. Available: https://www.efacec.pt/quem-somos/. [Accessed 06 2018].

[69] REN, "Caracterização da Rede Nacional de Transporte Para Efeitos de Acesso à Rede," [Online]. Available: http://www.mercado.ren.pt/PT/Electr/ActServ/AcessoRedes/CaractRNT//BibRelAno/CaracterizacaoRNT2017.pdf. [Accessed 09 2018].

[70] M. Zillgith. [Online]. Available: https://github.com/mz-automation/libiec61850. [Accessed 05 2018].

[71] R. Studený, "SIMULÁTOR KOMUNIKACE PROTOKOLŮ SCADA," Brno, Czech Republic, 2018.

[72] BeagleBone, "About BeagleBoard.org and the BeagleBoard.org Foundation," [Online]. Available: https://beagleboard.org/about. [Accessed 06 2018].

[73] R. Pi, "What is a Raspberry Pi?," [Online]. Available: https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/. [Accessed 06 2018].

[74] MEO, "Manual de Utilizador da FiberGateway," [Online]. Available: https://conteudos.meo.pt/meo/Documentos/Manuais/Routers/FiberGateway-Manual-Utilizador-V4.0-3.PDF. [Accessed 09 2018].

[75] Linksys, "WRT1200AC," [Online]. Available: http://downloads.linksys.com/downloads/datasheet/WRT1200AC.pdf. [Accessed 09 2018].

[76] 3GRouterStore.co.ul, "Huawei E5172 Quick Start Guide," [Online]. Available: https://exprtz.files.wordpress.com/2015/10/huawei_e5172_quick_start_guide_uk.pdf. [Accessed 09 2018].

[77] Netgear, "Wireless Cable Voice Gateway CG3000/CG3100 User Manual," [Online]. Available: https://www.comhem.se/asset/rqq4gefrk8ol/10sdhzs6R0Yi20OcqomCkk/3658d7d4d5a28043aec8d832dc419931/manual-netgear-cg3100-data.pdf. [Accessed 09 2018].

[78] Meo, "P.DG A1000G," [Online]. Available: https://conteudos.meo.pt/meo/Documentos/Manuais/Routers/Manual-Router-P-DG-A1000G.pdf. [Accessed 09 2018].

[79] "Raspbian," [Online]. Available: https://www.raspberrypi.org/downloads/raspbian/. [Accessed 04 2018].

[80] "MAC-address — what is this?," [Online]. Available: http://digitalsubstation.com/en/2013/12/01/what-is-mac-address/. [Accessed 09 2018].

[81] dd-wrt. [Online]. Available: https://dd-wrt.com/support/router-database/. [Accessed 10 2018].

[82] dd-wrt, "Installation," [Online]. Available: https://wiki.dd-wrt.com/wiki/index.php/Installation. [Accessed 10 2018].

[83] OpenVPN. [Online]. Available: https://openvpn.net/community-downloads/. [Accessed 10 2018].

[84] dd-wrt, "VPN (The Easy Way) V24+," [Online]. Available: https://wiki.dd-wrt.com/wiki/index.php/VPN_(the_easy_way)_v24+. [Accessed 10 2018].

[85] OpenVPN. [Online]. Available: https://community.openvpn.net/openvpn/wiki/BridgingAndRouting. [Accessed 10 2018].

[86] M. ZIllgith, "IEC 61850 GOOSE subscriber API," [Online]. Available: https://support.mz-automation.de/doc/libiec61850/c/latest/group__goose__api__group.html#ga3ed317b00bf2fb0a819f7941f06c0521. [Accessed 10 2018].

[87] [Online]. Available: https://www.worldtimezone.com/gsm.html. [Accessed 08 2018].

[88] F. K. M. K. S. S. B. R. Abderrahmane BenMimoune, "Inter-Cell Handoff Performance Improvement in LTE-A Multi-hop Relay Networks," in *12th ACM International Symposium on Mobility Management and Wireless Access*, Montreal, Canada, 2014.

[89] A. Apostolov, "Inter-Substation Communication leveraging IEC 61850 for wide area substation communications," [Online]. Available: https://www.smartgrid-forums.com/wp-content/uploads/2018/06/29.-IEC-61850-Europe-2016-Pacworld-Alex-Apostolov.pdf. [Accessed 08 2018].

[90] S. B. a. O. K. Charles M. Adrah, "Fusion networking technology for IEC 61850 inter substation communication," in *2017 IEEE International Conference on Smart Grid and Smart Cities*, Singapore, 2017.

[91] Cisco, "VPN vs Tunneling," [Online]. Available: https://learningnetwork.cisco.com/thread/112183. [Accessed 08 2018].

[92] L. G. G. K. M. Z. S. S. T. K. J. P. C. Frank Meylan, "An experimental study for transmitting MPEG-2 streamsover ATM networks," in *Presented at the 1st IEEE International Conference an ATM - ICATM'98*, Colmar, France, 1998.

[93] R. Hunt, "IEC 61850 Overview and Application," [Online]. Available: http://sites.ieee.org/houston/files/2016/01/Asset-Management-IEC61850_Overview_and_Application-Day-2.pdf. [Accessed 08 2018].

[94] IEC, "IEC 61850 Part 7-1: Basic communication structure for substation and feeder equipment - Principles and models," 2002.

[95] IEC, "IEC 61850-1 TR Ed.2: Communication networks and systems for power utility automation – Part 1: Introduction and Overview," 2012.

[96] A. R. I. G. A. I. S. a. H. D. Imtiaz Parvez, "A Survey on Low Latency Towards 5G: RAN, CoreNetwork and Caching Solutions," in *IEEE Communications Surveys & Tutorials PP(99)*, 2017.

[97] R. S. a. T. Xu, "Test procedures for GOOSE performance according to IEC 61850-5 and IEC 61850-10," [Online]. Available: http://testing.ucaiug.org/Testing/UCAIug%20Testing%20Quality%20Assurance%20Program/Current%20IEC%2061850%20Testing%20Procedures/Archives(ObsoletedVersions)/Test%20procedures%20for%20GOOSE%20Performance%20rev1p0_14Apr2010Final.pdf. [Accessed 08 2018].

[98] Qualcomm. [Online]. Available: https://www.qualcomm.com/media/documents/files/download-the-evolution-of-mobile-technologies-1g-to-2g-to-3g-to-4g-lte-qualcomm.pdf. [Accessed 08 2018].

[99] KTH, "Lecture 5: Substation Automation Systems," [Online]. Available: https://docplayer.net/40767859-Lecture-5-substation-automation-systems-course-map.html. [Accessed 11 2018].

Appendix A

The structure of GOOSE PDU

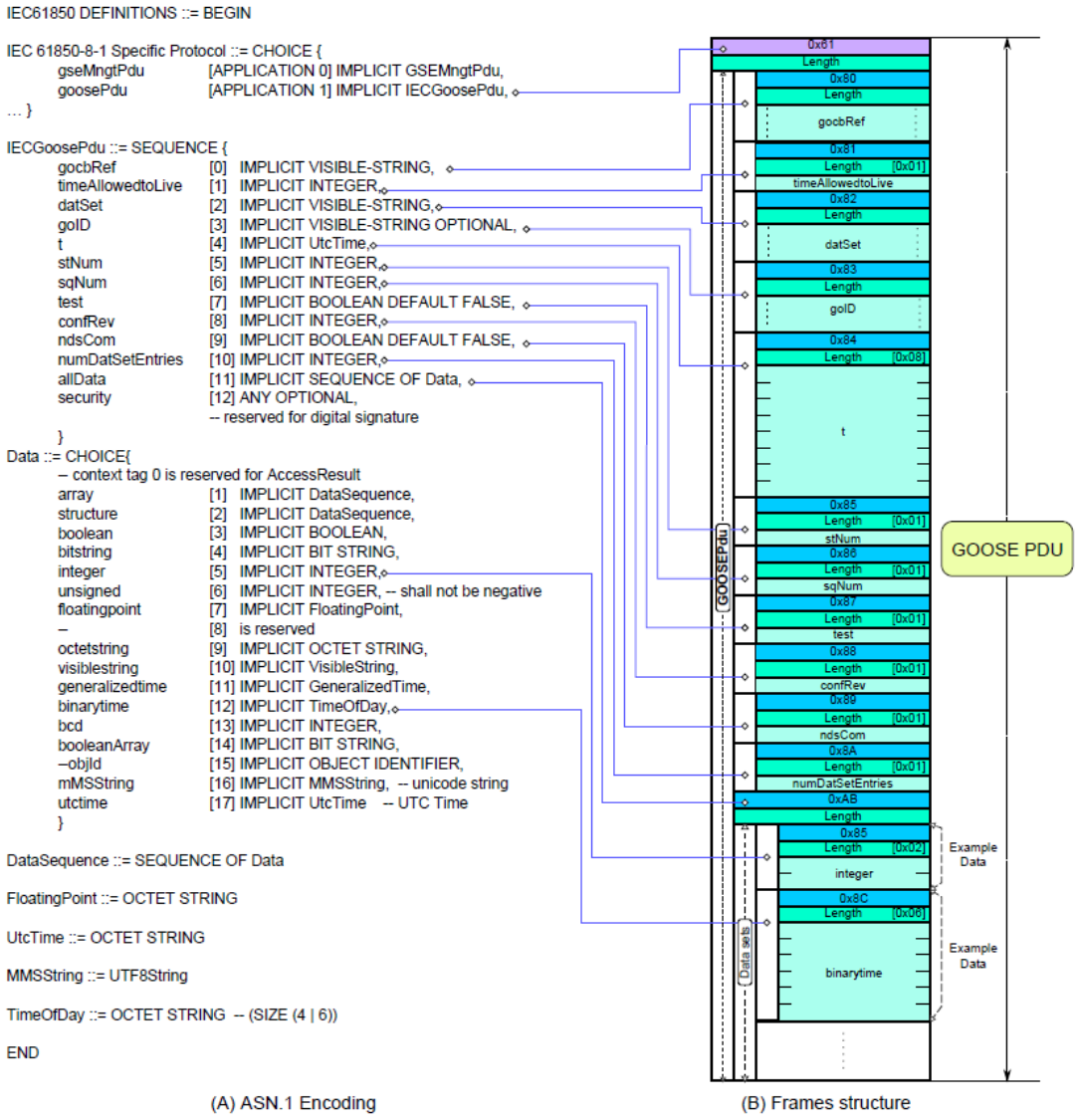


Figure 7.1: GOOSE PDU

Appendix B

The structure of SV PDU

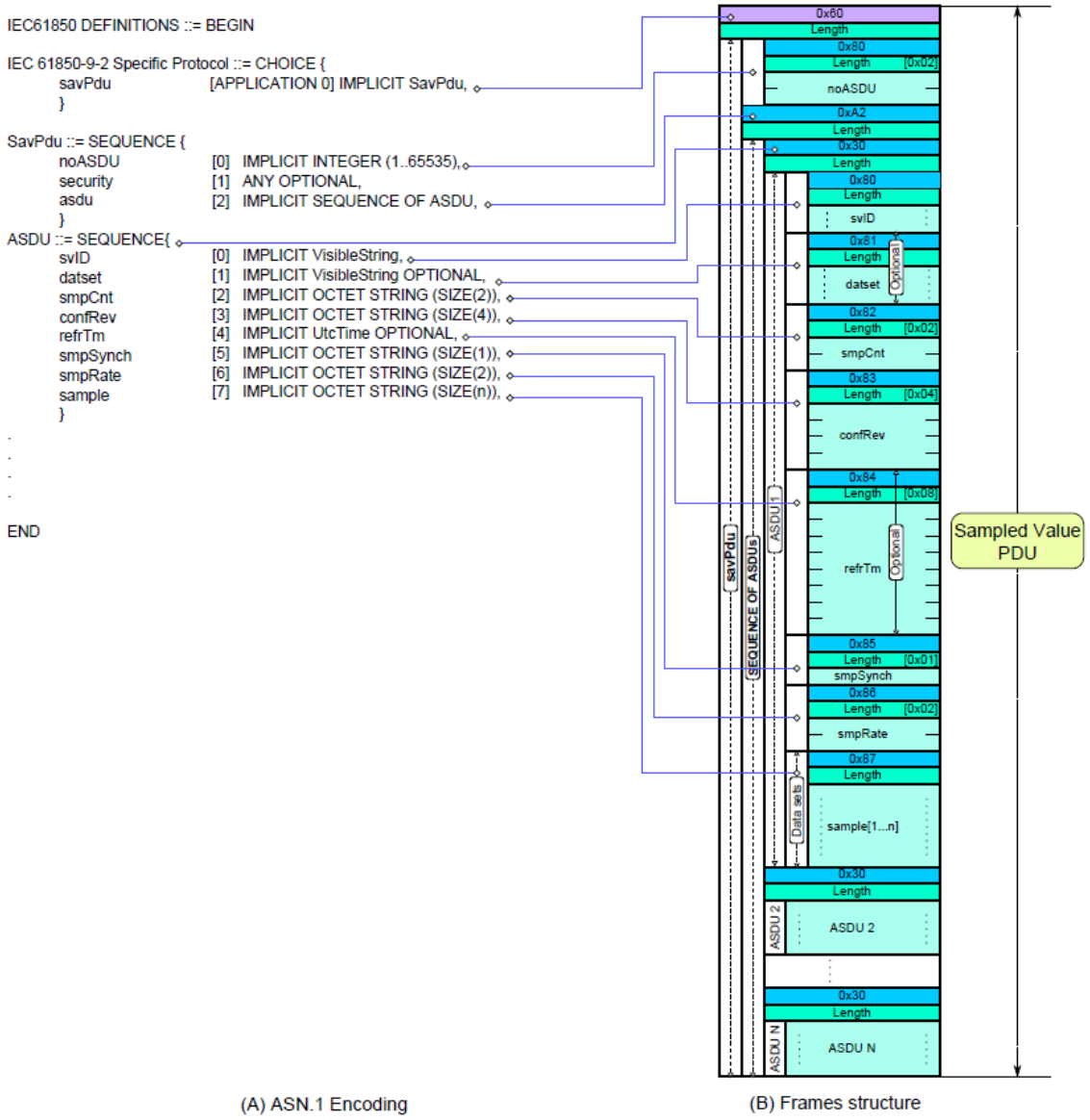


Figure 7.2: Sampled Values PDU

Appendix C

The principal configurations used in VPN Server

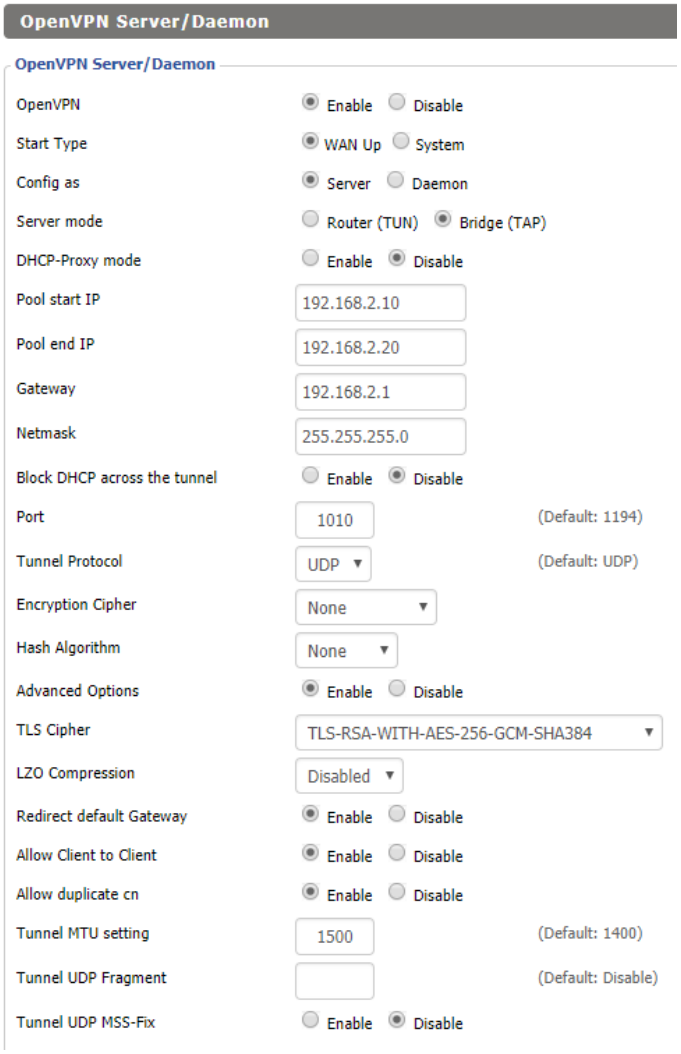


Figure 7.3: OpenVPN Server Configuration

Appendix D

Client OpenVPN configuration file

remote 192.168.2.1 1010
remote-cert-tls server
client
dev tap0
proto udp
resolv-retry infinite
nobind
persist-key
persist-tun
float
verb 3
<key>
“Certificate”
</key>
<cert>
“Certificate”
</cert>
<ca>
“Certificate”
</ca>

Appendix E

Matlab Script used to calculate the RTT and OWTT of UDP messages

```
clc
clear
close all
format long
%%
load('Workspace_02.mat')
%arrays de dados a inserir
valsServer = table2array(testServer02);
valsGateway = table2array(testGateway02);
n_messages= length(valsServer);
n_measures =length(valsServer)/2;
totalTimeServer=zeros(n_measures,1);  %; %time since udp message leaves the server and receives the same data again sent by the gateway
totalTimeGateway=zeros(n_measures,1);  %time since udp message arrive at gateway and sends the same data to the server
roundTrip=zeros(n_measures,1);   %time that take the total path without processing at gateway
n=1;
for i=2:2:n_messages
    totalTimeServer(n)= valsServer(i)-valsServer(i-1);
    totalTimeGateway(n)= valsGateway(i) - valsGateway(i-1);
    roundTrip(n)= totalTimeServer(n)-totalTimeGateway(n);
    n=n+1;
end
halfRoundTrip=roundTrip/2;
halfRoundTripMS=halfRoundTrip*1000;
roundTripMS=roundTrip*1000;
format short
%% Result analysis
%% RTT
meanHalfRoundTripMS = mean(halfRoundTripMS)
```

```
stdHalfRoundTripMS = std(halfRoundTripMS)
minHalfRTTMS =min(halfRoundTripMS)
maxHalfRTTMS = max(halfRoundTripMS)
confidenceIntervalHalfRTT95= 1.96*std(halfRoundTripMS)/sqrt(n_measures) %95% intervalo de conf
confidenceIntervalHalfRTT99=2.576*std(halfRoundTripMS)/sqrt(n_measures) %99% intervalo de conf
% Set up the figure properties
fig1 = figure(1);
p1=plot(halfRoundTripMS, '.-', 'LineWidth', 0.5)
title('OWTT of Each Message', 'fontweight', 'bold', 'fontsize', 14);
xlabel('Message Number', 'fontweight', 'bold', 'fontsize', 12);
ylabel('Time (ms)', 'fontweight', 'bold', 'fontsize', 12);
saveas(fig1,'OWTT.png')
%% Gateway
totalTimeGatewayMS=totalTimeGateway*1000;
meantotalTimeGatewayMS = mean(totalTimeGatewayMS)
stdtotalTimeGatewayMS = std(totalTimeGatewayMS)
mintotalTimeGatewayMS =min(totalTimeGatewayMS)
maxtotalTimeGatewayMS = max(totalTimeGatewayMS)
confidenceIntervalGateway95= 1.96*std(totalTimeGatewayMS)/sqrt(n_measures) %95% intervalo de conf
confidenceIntervalGateway99=2.576*std(totalTimeGatewayMS)/sqrt(n_measures) %99% intervalo de conf
```

Appendix F

Matlab Script used to calculate the RTT and OWTT of GOOSE messages

```
clc
clear
close all
format long
load('Workspace_01.mat')
%arrays de dados a inserir
valsServer194 = table2array(testServer01194);
valsServer606 = table2array(testServer01606);
valsClient194 = table2array(testClient01194);
valsClient606 = table2array(testClient01606);
n_measures =length(valsServer194); %measures are round trip messages (go and come)
n_messages=n_measures *2;
totalTimeServer=zeros(n_measures,1);  %; %time since udp message leaves the server and receives the same data again sent by the gateway
totalTimeGateway=zeros(n_measures,1);  %time since udp message arrive at gateway and sends the same data to the server
roundTrip=zeros(n_measures,1);   %time that take the total path without processing at gateway
n=1;
for i=1:1:n_measures
   totalTimeServer(i)= valsServer194(i)-valsServer606(i);
   totalTimeGateway(i)=  valsClient194(i)-valsClient606(i);
   roundTrip(i)= totalTimeServer(i)-totalTimeGateway(i);
end
…
```

(Continue as the same as Appendix E)

Appendix G

Complete table with IEC 61850 parts

Reference	Publication	Title
IEC TR 61850-1	2013	Introduction and overview
IEC TS 61850-2	2003	Glossary
IEC 61850-3	2013	General requirements
IEC 61850-4	2011	System and project management
IEC 61850-5	2013	Communication requirements for functions and device models
IEC 61850-6	2009	Configuration language for communication in electrical substations related to IEDs
IEC 61850-7-1	2011	Basic communication structure - Principles and models
IEC 61850-7-2	2010	Basic communication structure - Abstract communication service interface (ACSI)
IEC 61850-7-3	2010	Basic communication structure - Common Data Classes
IEC 61850-7-4	2010	Basic communication structure - Compatible logical node classes and data classes
IEC 61850-7-410	2012	Basic communication structure - Hydroelectric power plants - Communication for monitoring and control
IEC 61850-7-420	2009	Basic communication structure - Distributed energy resources logical nodes
IEC 61850-7-5	2012	Basic communication structure - Hydroelectric power plants - Modelling concepts and guidelines
IEC 61850-8-1	2011	Specific communication service mapping (SCSM) - Mappings to Manufacturing Message Specification MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3

IEC 61850-9-2	2011	Specific communication service mapping (SCSM) - Sampled values over ISO/IEC 8802-3
IEC 61850-9-3	2016	Precision time protocol profile for power utility automation
IEC 61850-10	2012	Conformance testing
IEC 61850-80-1	2016	Guideline to exchanging information from a CDC-based data model using IEC 60870-5-101 or IEC 60870-5-104
IEC 61850-80-3	2015	Mapping to web protocols - Requirements and technical choices
IEC 61850-80-4	2016	Translation from the COSEM object model (IEC 62056) to the IEC 61850 data model
IEC 61850-90-1	2010	Use of IEC 61850 for the communication between substations
IEC 61850-90-2	2016	Using IEC 61850 for communication between substations and control centers
IEC 61850-90-3	2016	Using IEC 61850 for condition monitoring diagnosis and analysis
IEC 61850-90-4	2013	Network engineering guidelines
IEC 61850-90-5	2012	Use of IEC 61850 to transmit synchrophasor information according to IEEE C37.118
IEC 61850-90-7	2013	Object models for power converters in distributed energy resources (DER) systems
IEC 61850-90-8	2016	Object model for E-mobility
IEC 61850-90-12	2015	Wide area network engineering guidelines

Table 7.1: IEC 61850 parts

Appendix H

Modifications performed to provoke an Over Voltage faillure

```
//---------OVER VOLTAGE--------
if (IedServer_getBooleanAttributeValue(iedServer,IEDMODEL_RPi1_GGIO1_Ind2_stVal) != digitalRead(27)) {
  IedServer_updateTimestampAttributeValue(iedServer, IEDMODEL_RPi1_GGIO1_Ind2_t, &iecTimestamp);
  IedServer_updateBooleanAttributeValue(iedServer, IEDMODEL_RPi1_GGIO1_Ind2_stVal, digitalRead(27));
  //IedServer_updateBooleanAttributeValue(iedServer, IEDMODEL_RPi1_LLN0_Loc2_stVal, digitalRead(27));
  cout<<"Pin 16 changed! | digitalRead: "<<IedServer_getBooleanAttributeValue(iedServer,IEDMODEL_RPi1_GGIO1_Ind2_stVal)<<endl;
}
else{
  cout<<"No changes pin 16, no changes on OVERVOLTAGE! | digitalRead: " << digitalRead(27) <<endl;
  IedServer_updateBooleanAttributeValue(iedServer, IEDMODEL_RPi1_GGIO1_Ind2_stVal, !digitalRead(27));
};
```

Appendix I

Location of mobile antennas in Portugal for the different mobile operators

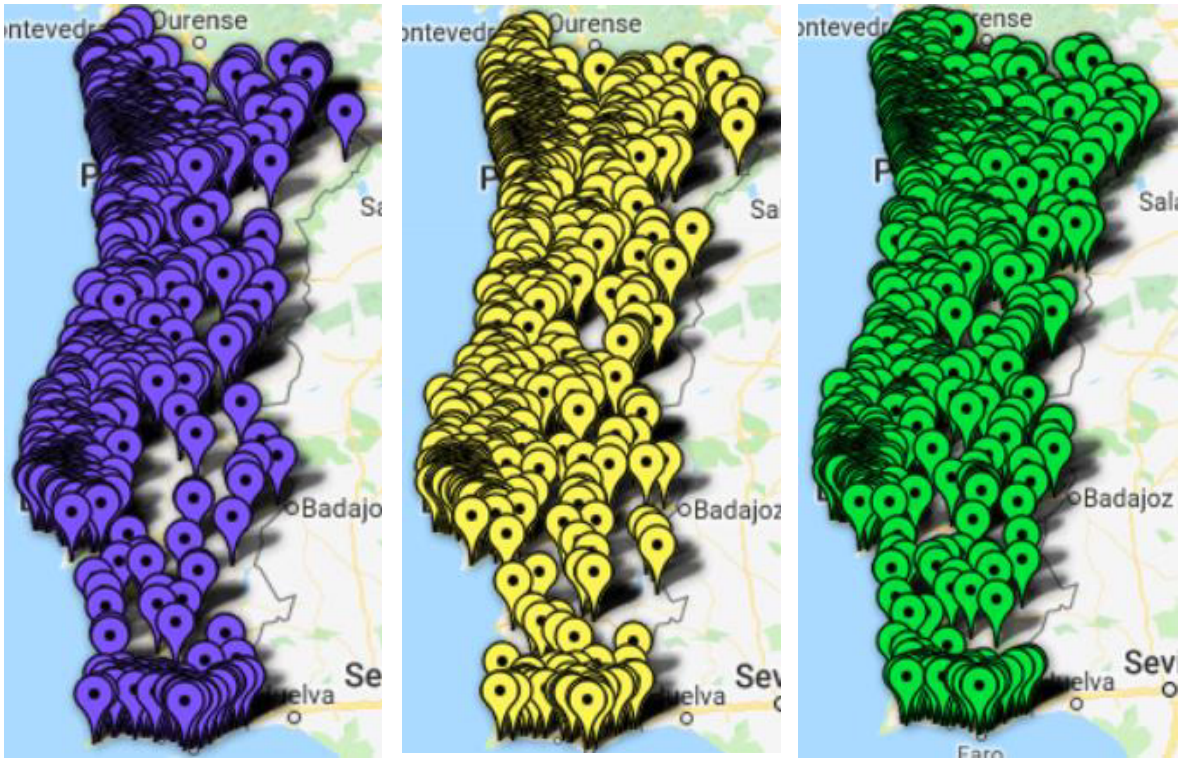


Figure 7.4: From the left to the right: Vodafone, NOS and MEO

Appendix J

Procedure to perform the Cross-Compiling

Loading the project, it is necessary to:

1. Press with right click over the Project and open the Properties.
2. Going to Build tab and insert IP of the Raspberry Pi where the code will be built, by pressing the three dots.
3. Next, is necessary to link the libraries. The libraries are: lpthread, liec61850 and lwiringPi. Looking to the Figure 7.6, added them all like it is shown.
4. Last but not least, to confirm that everything is correct, on top of NetBeans, press Tools, go to C/C++ tab and check if Building Host is the one that was inserted before and others options are the same like the Figure 7.7, showing the localization of the C/C++ compilers in Raspberry PI.

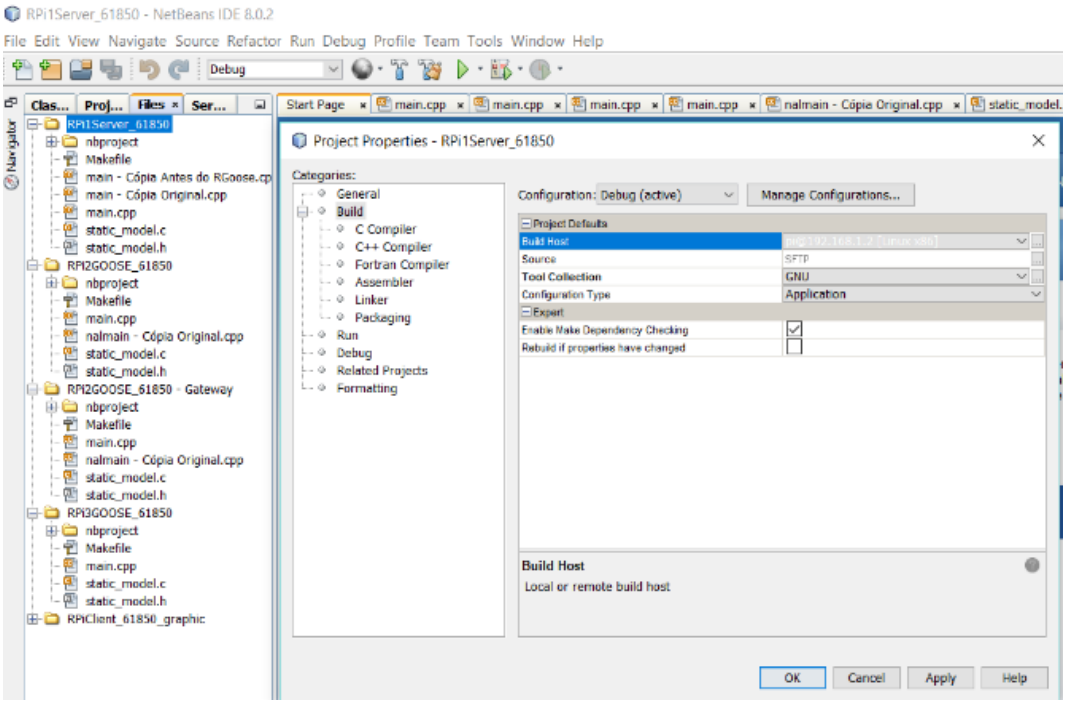


Figure 7.5: Image a) for Cross-Compiling using Netbeans

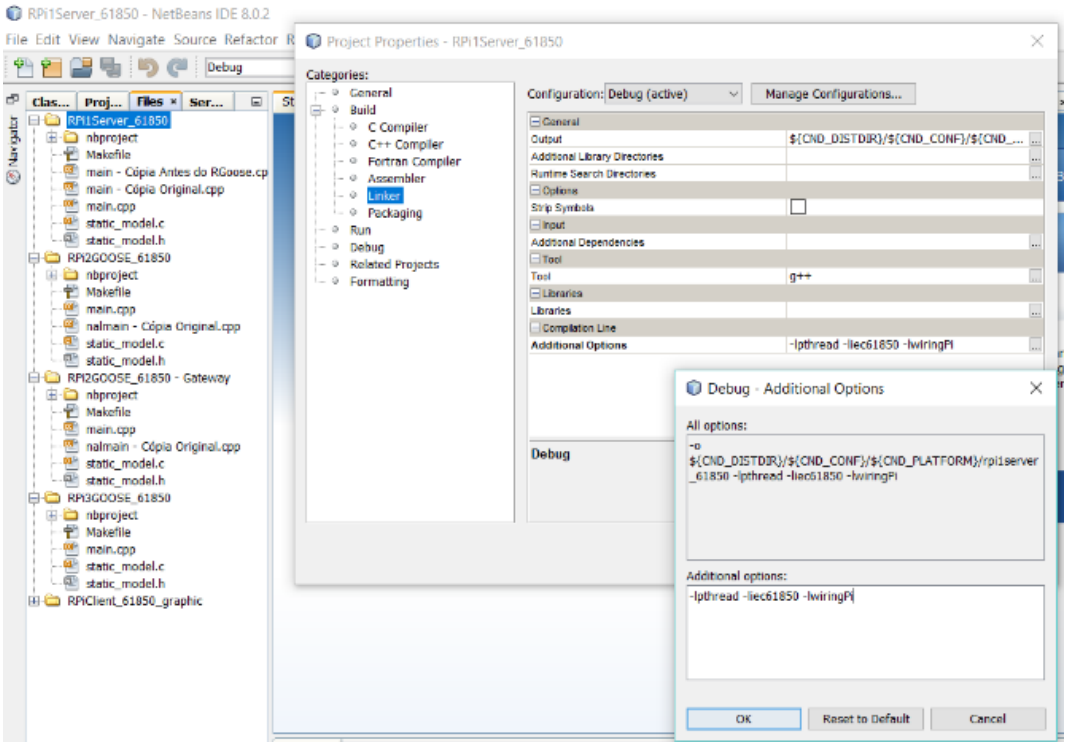


Figure 7.6: Image b) for Cross-Compiling using Netbeans

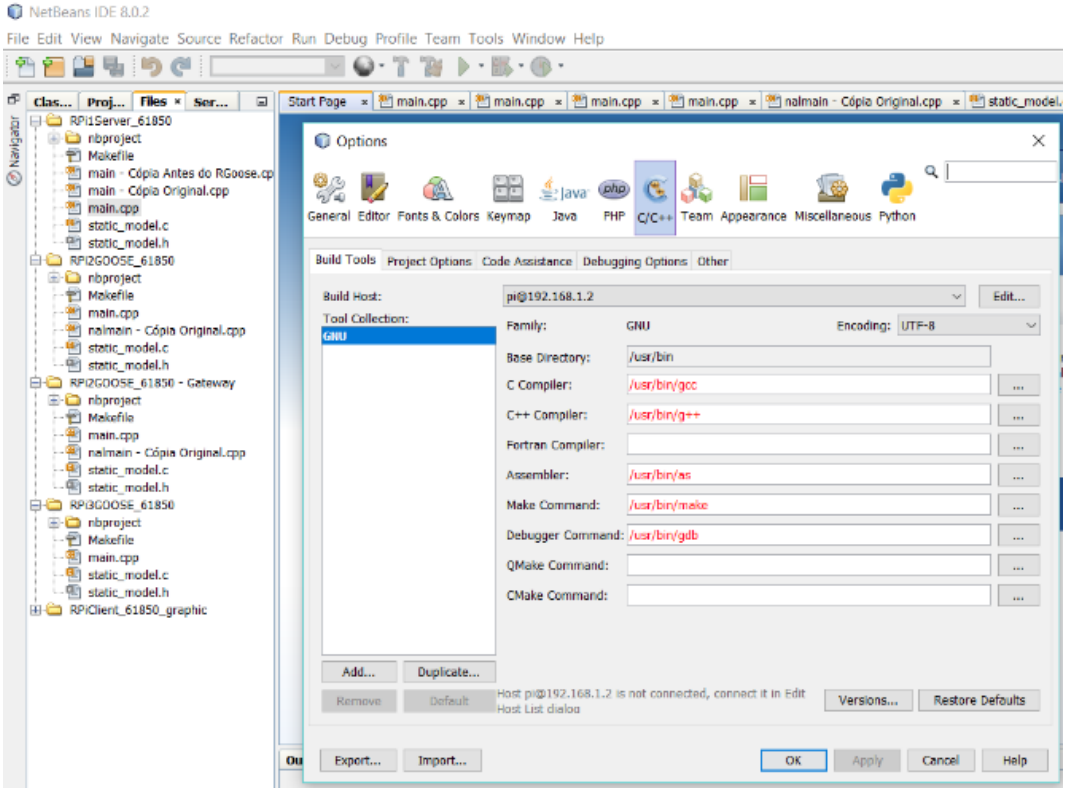


Figure 7.7: Image c) for Cross-Compiling using Netbeans

Appendix K

Implementation of Gateway Sender in *goose_publisher.c* file

```
397  void die(char *s)
398  {
399      perror(s);
400      exit(1);
401  }
402
403  void RGoose(uint8_t* buffer, int packetSize){
404      struct sockaddr_in si_other;
405      int s, i, slen=sizeof(si_other);
406      char buf[BUFLEN];
407
408      if ( (s=socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) == -1)
409      {
410          die("socket");
411      }
412      memset((char *) &si_other, 0, sizeof(si_other));
413      si_other.sin_family = AF_INET;
414      si_other.sin_port = htons(PORT);
415      if (inet_aton(SERVER , &si_other.sin_addr) == 0)
416      {
417          fprintf(stderr, "inet_aton() failed\n");
418          exit(1);
419      }
420          //send the message
421          if (sendto(s, buffer, packetSize, 0 , (struct sockaddr *) &si_other, slen)==-1)
422          {
423              die("sendto()");
424          }
425          //receive a reply and print it
426          //clear the buffer by filling null, it might have previously received data
427          memset(buf,'\0', BUFLEN);
428          //try to receive some data, this is a blocking call
429          if (recvfrom(s, buf, BUFLEN, 0, (struct sockaddr *) &si_other, &slen) == -1)
430          {
431              die("recvfrom()");
432          }
433      close(s);
434  }
```

Figure 7.8: Functions RGoose and die

Appendix L

Implementation of Gateway Receiver

```
64  int
65  main(int argc, char** argv) {
66
68      struct sockaddr_in si_me, si_other;
69
69      int s, i, recv_len = 0;
71      socklen_t slen = sizeof (si_other);
72      uint8_t* buf = (uint8_t*) malloc(BUFLEN);
73
73      //create a UDP socket
75      if ((s = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) == -1) {
76          die("socket");
77      }
78
79      // zero out the structure
80      memset((char *) &si_me, 0, sizeof (si_me));
84      si_me.sin_family = AF_INET;
85      si_me.sin_port = htons(PORT);
87      si_me.sin_addr.s_addr = htonl(INADDR_ANY);
88
89      //bind socket to port
90      if (bind(s, (struct sockaddr*) &si_me, sizeof (si_me)) == -1) {
91          die("bind");
92      }
93
94      //keep listening for data
95      while (1) {
96          cout << endl;
97          printf("Waiting for data...");
99          cout << endl << endl;
100         fflush(stdout);
```

Figure 7.9: Code at Gateway Receiver – a)

```
104         //print details of the client/peer and the data received
            printf("Received packet from %s:%d\n", inet_ntoa(si_other.sin_addr), ntohs(si_other.sin_port));
106         cout << endl;
107
108
109         //now reply the client with the same data
            if (sendto(s, buf, recv_len, 0, (struct sockaddr*) &si_other, slen) == -1) {
111             die("sendto()");
112         }
113
114         //parseUDP
115         int bufPos = 12;
116         uint8_t* buffer = buf;
117         int numbytes = recv_len;
118
119         //too small
120         if (numbytes < 22)
121             return 0;
122
123         int headerLength = bufPos + 2;
124
125         cout << "bufPos: " << bufPos << endl;
126         cout << "headerLength: " << headerLength << endl;
127
128
129         /* check for VLAN tag */
130         if ((buffer[bufPos] == 0x81) && (buffer[bufPos + 1] == 0x00)) {
131             cout << "VLAN TAG 0x81 0X00" << endl;
132             bufPos += 4; /* skip VLAN tag */
133             headerLength += 4;
134         } else
135             cout << "VLAN TAG WRONG!" << endl;
136
137
138         /* check for GOOSE Ethertype */
139         if (buffer[bufPos++] != 0x88)
140             cout << "GOOOSE Ethertype different 1 !" << endl;
141         if (buffer[bufPos++] != 0xb8)
142             cout << "GOOOSE Ethertype different 2 !" << endl;
```

Figure 7.10: Code at Gateway Receiver – b)

Appendix M

Procedure to enable SSH and VNC

First, it's necessary to update everything using two commands:

- $sudo apt-get update
- $sudo apt-get upgrade

Later, after completed the installation of the updates, there are two things that are required to enable to control the Raspberry PIs remotely. So, there are two ways to activate SSH and VNC:

- Using GUI, going Menu Setting on left top corner:
  - Press Preferences
  - Then, press Raspberry Pi Configurations
  - Next, go to Interfaces tab
  - And finally, press enable in SSH and VNC
- Using command line, by typing:
  - $sudo raspi-config
  - Go to option 5, Interfacing Options
  - Then, enable SSH and VNC

Once both options are enabled, it will be possible to control the devices either by GUI through VNC, either by command line through PuTTY, in my PC. Both connections are made by IP address that can be obtained by typing in the command line $ifconfig.

Appendix N

Procedure to compile the libIEC61850

The procedure will be:

- Place the library inside of an easy and rememberable folder, for example in home folder. It could be downloaded from Michael Zillgith github [70] or transferred by USB flash drive.
- Open a terminal.
- Enter the inside the library's folder with:
  - $cd libiec61850-1.2
- Compiling inside the folder, if for some reason, it would be needed to try the examples:
  - $sudo make install
- Compiling to /usr/local in order that later, the files will be referred in the header of main program:
  - $sudo make INSTALL_PREFIX=/usr/local install

Later, when I tried to compile my main program in Netbeans to the Raspberry Pi, it was notice that something was missing in the headers. The header "hal_etherneet.h" was missing and it was necessary to copy it from the library to the folder where other headers were. So, after the building the lib:

- Enter in libiec61850-1.2.
- Going to the folder where is the header that is missing.
  - $cd libiec61850-1.2/src/hal/inc
- And copy the "hal_etherneet.h" where it was missing.
  - $sudo cp -r hal_ethernet.h /usr/local/include

Appendix O

Procedure to run the examples of libIEC61850

So, the procedure in any Linux environment will be:

- Opening the project root directory and build the code.
  - $sudo make
- After build the library, it is necessary to enter the folder “examples”.
  - $cd examples
- To run the GOOSE example, just enter in “goose_publisher”.
  - $cd goose_publisher
- Now, it is necessary to build the “goose_publisher”.
  - $sudo make
- To run the example, it is necessary to add as argument the network interface that it is desired to be use. If it is eth0, no argument is necessary to introduce.
  - $sudo ./goose_publisher
- Again, in new computer, repeat the procedure until enter the folder “examples”.
- Now, to add a subscriber, enter in “goose_subscriber” folder.
  - $cd goose_subscriber
- Build “goose_subscriber” code.
  - $sudo make
- Add as argument the network interface, if necessary.
  - $sudo ./goose_subscriber

Appendix P

Modifications to change the interface to Wi-Fi

```
IedServer_setGooseInterfaceId(iedServer, "wlan0");
```

(…..)

```
cout<<"GOOSE interface wlan0\n";
GooseReceiver_setInterfaceId(receiver, "wlan0");
GooseReceiver_setInterfaceId(receiver1, "wlan0");
```