POLITÉCNICO DE LEIRIA | ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

Dissertation

Master in Computer Engineering - Mobile Computing

Mobile long range rfid reader for indoor positioning system (mlrips)

Ricardo Batista dos Santos

Leiria, *November 2021*

This page was intentionally left blank

POLITÉCNICO DE LEIRIA | ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

Dissertation

Master in Computer Engineering - Mobile Computing

Mobile long range rfid reader for indoor positioning system (mlrips)

Ricardo Batista dos Santos

Dissertation developed under the supervision of João da Silva Pereira PhD, professor at the School of Technology and Management of the Polytechnic Institute of Leiria, co-supervision of Sílvio Priem Mendes PhD, professor at the School of Technology and Management of the Polytechnic Institute of Leiria and Hugo Miguel Cravo Gomes PhD, professor at the School of Technology and Management of the Polytechnic Institute of Leiria.

Leiria, *November 2021*

This page was intentionally left blank

Dedication

This dissertation is dedicated to my family.

This page was intentionally left blank

Acknowledgements

First, I would like to thank all my family, especially my parents, and friends for supporting me in this very long journey. A very special thanks to my girlfriend for supporting me every day and my dog, Mac, for being the best stress reliefer.

I would also like to thank my professors, prof. Sílvio Mendes, prof. Hugo Gomes, and prof. João Pereira, for all the patience they had with me over the course of these two long years, and for understanding how hard it is to work and study at the same time.

Lastly, I would like to thank CIIC (Computer Science and Communication Research Centre) and "Instituto de Telecomunicações" for giving me a place to work at the school.

Thank you all for your support.

This page was intentionally left blank

Resumo

A indústria está em constante evolução, encontrando-se atualmente a viver a 4ª revolução industrial. Novos desafios exigem novos conceitos como IoT (Internet of Things), tecnologias que criem redes de dispositivos capazes de monitorizar, enviar e recolher dados essenciais para a criação de novas formas de produção, melhoria e otimização de processos em tempo real. No presente projeto pretende-se utilizar tags RFID passivas para criar um sistema de localização interior, baseado na indústria dos moldes de forma a permitir a localização de moldes e ferramentas dentro do edifício industrial. É apresentada uma solução que, utilizando tags RFID passivas acopladas a moldes e ferramentas, permite a sua identificação e localização com um conjunto de leitores que ficam acoplados em locais específicos de um armazém como entradas, pontos estratégicos ou mesmo na ponte rolante do edifício. A localização dos "objetos" será transmitida usando um protocolo de comunicação industrial denominado de OPC-UA (OPC Unified Architecture).

Palavras-chave: RFID, Tags, IPS, OPC-UA

This page was intentionally left blank

Abstract

The industry is evolving and is currently living in the 4th industry revolution. It is facing new concepts, such as IoT (Internet of Things), which can create networks of devices to monitor, send and collect data. It enables new ways of production, value-creation, and real time optimization. This project makes use of passive RFID (Radio Frequency Identification) tags for indoor positioning systems (IPS), in the moulds industry allowing the localization of moulds or tools inside big warehouses. A solution was created using RFID tags and an array of readers attached to specific industrial zones, able to locate moulds or tools with a precision of half a meter. The tag position will be broadcasted using an industry standard communication protocol called OPC-UA (OPC Unified Architecture).

Keywords: RFID, Tags, IPS, OPC-UA

This page was intentionally left blank

List of figures

This page was intentionally left blank

List of tables

This page was intentionally left blank

List of acronyms

AIDC	Automatic Identification and Data Capture
AOA	Angle of Arrival
ARM	Advanced RISC Machine
ASD	Aerospace Software Developments
AUTO-ID	Automatic Identification
BLE	Bluetooth Low Energy
CAN	Controller Area Network
CEPT	European Conference of Postal and Telecommunications Administrations
CNC	Computer Numerical Control
COM	Communication Port
CPU	Central Processing Unit
CRIS	Centre of Railway Information Systems
DCOM	Distributed Component Object Model
DLL	Dynamic Link Library
DNA	Deoxyribonucleic Acid
EAN	Europe Article Number
EAS	Electronic Article Surveillance
EPC	Electronic Product Code
ETSI	European Telecommunications Standards Institute
FCC	Federal Communications Federation
FCL	Framework Class Library
FoF	Friend or Foe
GPS	Global Positioning System
GUI	Graphical User Interface
HDMI	High-Definition Multimedia Interface
HF	High Frequency
HMI	Human Machine Interface
IoT	Internet of Things
IPS	Indoor Positioning System
ISM	Industrial, Scientific and Medical Bands

ISO	International Standard Organization
JSON	JavaScript Object Notation
kNN	K-Nearest-Neighbour
LF	Low Frequency
LOS	Line-of-Sight
OCR	Optical Character Recognition
OLE	Object Linking and Embedding
OPC-UA	Open Platform Communication – Unified Architecture
OS	Operative System
PC	Personal Computer
PLA	Polylactic Acid
PLC	Programmable Logic Controller
POA	Place of Arrival
RAM	Random Access Memory
RDBMS	Relational Database Management System
RFID	Radio-Frequency Identification
RPI	Raspberry Pi
RSP	Received Signal Phase
RSS	Received Signal Strength
RSSI	Received Signal Strength Indicator
RTLS	Real Time Location Systems
SCADA	Supervisory Control and Data Acquisition
SDK	Software Development Kit
SHF	Super-High Frequency
SKU	Stock Keeping Unit
SOA	Service Oriented Architecture
TCP	Transmission Control Protocol
TDMA	Time-Division Multiple Access
TDOA	Time-Difference of Arrival
TOA	Time of Arrival
UHF	Ultra-High Frequency
UPC	Universal Product Code
URL	Uniform Resource Locator
USA	United States of America
USB	Universal Serial Bus

UWB	Ultra-Wide Band
VPN	Virtual Private Network
XML	Extensible Markup Language

This page was intentionally left blank

Table of Contents

1. Introduction

Our lives are constantly changing, not just because we are always getting older but there is also a lot of political, economic and technological factors involved. If we look a few centuries back, when the industry (economic activity concerned with the processing of raw materials and manufacture of goods in factories [1]) was created no one could believe we would get to this point in time.

The first industrial revolution began when people started using steam and water to create machines that could speed up the manufacturing process [2]. Not long after, electricity became mainstream and usable in the industry leading to the second industrial revolution. Then, technology started to evolve, and people found a way to use it to automate production, the third industrial revolution. Currently we are living in the fourth industrial revolution (Industry 4.0) where technology is embedded into factories to optimize production and introduce intelligence in plants.

With the rise of the Internet of Things (IoT), everyday appliances use modern control systems and embedded software systems to be able to send and receive all kinds of data. These developments open the door to new products because all these devices are interconnected, enabling new ways of production, value-creation and real time optimization.

This project makes use of these new technologies in the industry where big warehouses have all kinds of pieces, tools and moulds and there's an urgent need to locate all those parts inside a wide space. Global Positioning System (GPS) is the most used technology but unfortunately it doesn't work very well inside closed spaces and has a very poor accuracy. To solve this problem, a new solution was created using the Radio-Frequency Identification (RFID) technology, with a reader coupled to a server and several connected RFID tags attached to the parts to estimate the position of the moulds inside a warehouse. This server also uses the Open Platform Communication – Unified Architecture (OPC-UA) server, which is the current most used communication protocol in the industry, to send the information of the tag's position to all the clients.

This dissertation will present the process of creating the solution described in the last paragraph.

The first chapter is the introduction of the main objectives with a brief explanation of the project and its requirements.

The second chapter will start with an overview of RFID's technology predecessors, the tags that are used and the several types and categories of the tags and some examples on how and where it is used. It will then finish with an overview of OPC-UA protocol, from its predecessor, the need to evolve into a new protocol and an example of its use.

The third chapter is dedicated to the software as well as the hardware used for this project, specifying which antenna, tags and reader is used for the proposed solution.

The fourth chapter specifies the physical architecture, starting with the first iterations of the project and finishing with the final solution.

The fifth chapter is the development chapter, where the project is disclosed. It starts by specifying the platform where it will run, followed by the first tests made using the reader and passive tags. Next, it details how the tags are located by using a mathematical model to compute the tags position, followed by the implementation of the project and ends with the implementation of the OPC-UA server.

The sixth and last chapter concludes this dissertation, with a brief description of the major obstacles, whether the project was a success and future work.

2. State of the art

In this chapter the technology chosen for the development of this project - RFID Technology – and the communication protocol – OPC-UA - will be presented. It starts with a brief explanation of RFID's predecessors, followed by a detailed explanation of the technology, the different types of tags and frequencies of operation, the most common algorithms used for location services and end with real life examples. It then ends with an overview of OPC-UA protocol, from the OPC Classic and why it was not fit for the current days, its structure, security and an example of its use.

2.1. RFID Predecessors

RFID technology is in a category called AIDC (Automatic Identification and Data Capture). This is a term given to several technologies that are used to help machines identify objects [3]. These are often coupled with some kind of automated data capture, allowing companies to identify objects without having employees typing it in. A marker, or a tag, is coupled to the object and by using a method of capturing information, it is possible to identify, store automatically and analyse data into a computer. The following sub-chapters will make a brief explanation of the previous systems, still in use in the present days.

2.1.1. Barcode Systems

A barcode is a machine-readable binary code with bars and gaps that arranged to form a predetermined pattern [4]. This sequence refers to an associated symbol. That can be interpreted both numerically and alphanumerically as well. It is usually read using an optical laser scanner that scans the barcode using the reflection of the laser beam on the

black and white gaps. There are several types of barcodes but usually they follow a standard such as EAN (Europe Article Number). This standard was designed in 1976 specifically for the grocery industry. This standard comes from the UPC (Universal Product Code) developed in the United States [5].

The EAN code consists of 13 digits. The first two or three specify the country where the code was assigned. The next five numbers identify the company, the other five identify the item and the last number is a check digit as can be seen from Figure 1 [6].

Figure 1 - Barcode EAN

These codes have many disadvantages such as the limited storage capacity and they cannot be reprogrammed. Its use in big dusty warehouses is also a problem because the barcodes can get dirty and can no longer be read and if a worker wants to scan a whole warehouse of products, he will have to scan every and each code, one by one.

2.1.2 Magnetic Strips and Chip Cards

The magnet strip was the technology used before smart cards became mainstream. It is usually identified by a magnetic strip on the side of a card, as shown on Figure 2 [7]. They were mostly used as credit cards, telephone cards, bank cards, and are still currently used,

for instance, in railroads tickets. The black strip contained only a simple number in it. These magnetic cards were easily destroyed when exposed to magnetic fields [7].

Figure 2 - Magnetic strip cards.

Smart cards consist of an electronic data storage system sometimes with a microprocessor in it with computing capabilities and an RFID antenna, as shown on Figure 3 [8]. These cards are the current technology in bank cards, or even our personal identification card. They are usually locked using a PIN-code that is then cross checked with a PIN-code stored in the card's storage. They are safer to use than the one with magnetic strips but face the same problems when exposed to magnetic fields.

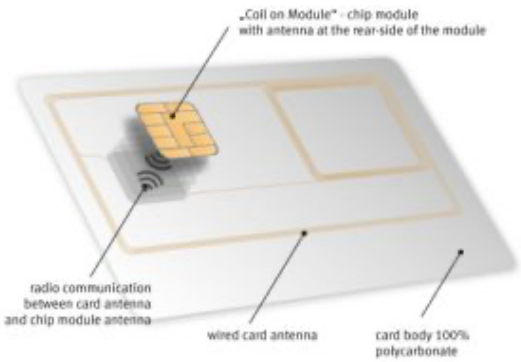

Figure 3 - Smart card

2.1.3 Biometrics

Biometrics are body measurements and calculations related to all living-beings. It is used in computer science as a form of authentication and access control. Several examples of biometric authentication can be fingerprints, face recognition, DNA (Deoxyribonucleic acid), iris/retina recognition and voice [9]. The most common biometrics systems are shown on Figure 4 [10].

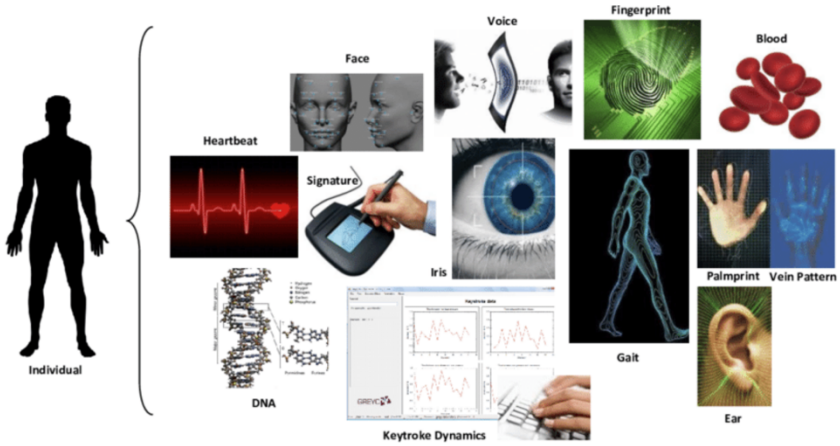

Figure 4 - Biometrics Systems

Voice recognition is currently widely used from warehouses to call centres, and even access control. One of the most used applications is converting spoken words into digital signals and comparing the speech characteristics of the individual with the one stored on a database, grating access to that specific individual.

Fingerprinting is the procedure of using fingerprints as a means of identification and it is used in modern law enforcement, and it is also used on hospitals, universities, and industries as a way of attendance control. There are also commercial uses of these systems, such as entrance verification. An individual fingerprints can be obtained from the person's fingers, this means that a person fingerprint is stored in a database, and when

a person wants to enter a specific area, he must insert his finger into a reader that cross checks his finger with the ones that can have access on the database, grating him access.

2.2. RFID

The beginnings of RFID technology are directly associated with the beginning of modern radio communication and the creation of the radar in the early 20th century [11]. RFID was born with the combination of these two technologies. The first use-case for RFID was during the 2nd World War. The allied forces used RFID in FoF (Friend of Foe) which was a system to identify a friendly aircraft from an enemy one [12]. It worked by having a transponder on the aircraft that would show up on the radar as friendly. It was developed after having some friendly fire incidents.

In the 1960s RFID was starting to become more commercially available as a counter theft measure of Electronic Article Surveillance (EAS). These systems are commonly found on shop floors. They work by utilizing 1-bit tags, where the presence of the absence of a tag can be detected. This was and still is an inexpensive and effective measure of detecting shop thefts.

In the 1980s, it was where the full implementation of this technology started having separate ways of development in the USA and Europe [13]. In the USA, the studies were mainly focused on transportation, personnel access and to a small portion of animals. In Europe, the focus was animal tagging, industrial and business applications as well as electronic toll collection [14].

Following this divergence of technologies between Europe and the USA, RFID technologies followed different standards on operation frequencies. In the USA, for example, the authority that is responsible for regulating their standard is FCC (Federal Communications Commission), which dedicates 25 channels of approximately 500kHz bandwidth in frequencies from 902 to 928 MHz. On the other hand, Europe's standard regulator is CEPT (European Conference of Postal and Telecommunications Administrations), which is based on ETSI (European Telecommunications Standards

Institute) that dedicates 15 channels, around 860 MHz, but a reader can only broadcast on 4 of them at maximum power. The channel size for is 200kHz depending on the mode.

2.3. RFID Technology

An RFID system consists of three major components:

- Antenna.
- Reader.
- Transponder (RFID-tag).

The antenna is usually coupled with the reader. This antenna can be adjusted based on the scenario and application of the project. The tag also has a small antenna and usually only depends on the system frequency. When the reader and antenna radiate, the tag to air interface activates, and the antenna can read or write values on it. This process is illustrated on Figure 5 [15]. The antennas are made in different sizes and shapes, depending on the frequency of usage, range and directivity making them usable in all kinds of applications. The antenna creates an electro-magnetic field which can be permanent when it is expected to read several quantities of tags, for instance, in a toll collection system, or it can be activated and deactivated whenever needed [16].

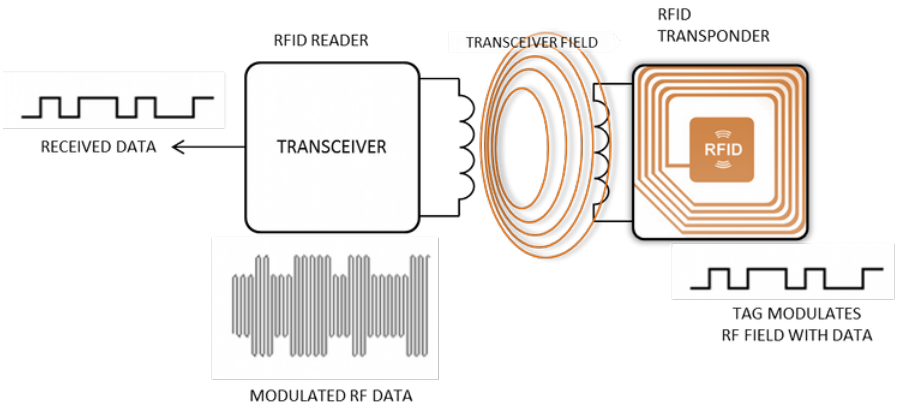

Figure 5 - RFID Architecture

A transponder, called RFID-tag in this document, is a data carrying device that consists of a coupling element (tag antenna) and an electronic chip, as can be seen on Figure 6 [17]. Some of these tags do not have their own power, making them passive tags. The power needed to send/receive and/or change data is sent by the reader to the antenna. These tags can also be made in several sizes and shapes making usable in pretty much every scenario.

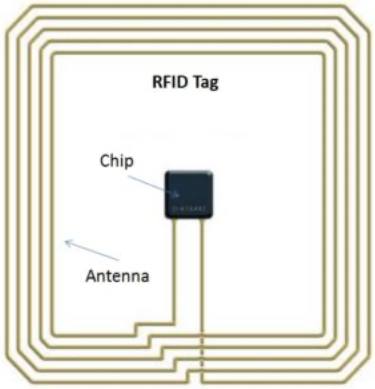

Figure 6 - RFID Tag

2.3.1. Tag Types

RFID tags can be divided into three major types:

- Active.
- Passive.
- Semi-passive.

Active tags have their own power source, usually a battery. These tags are mostly under the UHF (Ultra High Frequency), which can range from 300MHz to 3GHz and can have ranges to 100 meters in some cases. A diagram of these tags is shown on Figure 7[18]. These kinds of tags are usually much larger and expensive than passive tags and are used to track large assets. Active tags are very often equipped with sensors that can measure and transmit parameters like humidity, temperature, light and many others. These tags can also

be divided into two classes Transponders and Beacons. Transponders only “wake up” and transmit their data when they receive a radio signal from a reader’s antenna. For instance, electronic toll payment uses these tags. They are dormant and when the user goes through the toll, the tags become readable by the reader, conserving its energy. Beacons can emit signals on a pre-set interval. These tags are usually used in RTLS (Real-Time Location Systems) that are commonly used to track cargo containers on a dock or tracking wheelchairs inside a hospital.

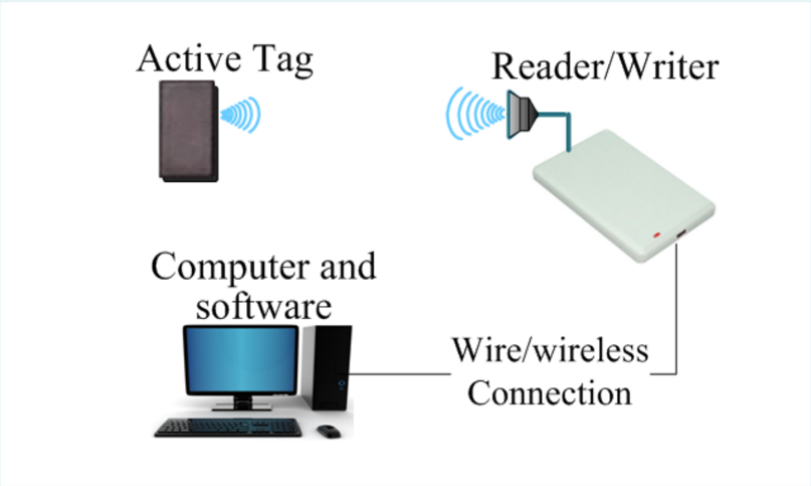

Figure 7 - RFID active tag architecture

Passive tags do not have their own power source. The signal received by the tag’s antenna must have enough energy to reflect on the tag and send the tag’s data back to the reader. These tags could use the LF (Low Frequency, 30 to 300KHz), HF (High Frequency, 3 to 30MHz) or UHF. The read ranges become shorter since the tag is limited by the power sent by the reader. These tags are usually smaller, cheaper, and more flexible than active tags. This means that they can be attached into any object. Figure 8[18] shows a diagram of a passive RFID tag.

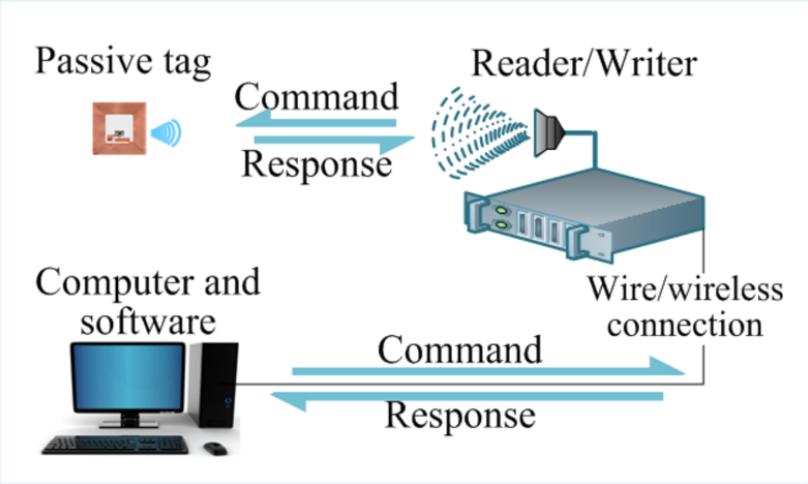

Figure 8 - RFID passive tag architecture

Semi-passive tags are a hybrid of active and passive tags. They can also have a power source built into them that helps ensure that all the energy captured from the reader can be used to reflect the signal, which can improve data transfer rates and read distance.

There are several implementations and several categories for RFID systems. Tags can be read/write and read-only tags. The data on read-only tags is programmed with a unique set of data, usually in the space of 32 to 127 bits and it cannot be modified. Read and Write tags, on the other hand, can store different information in its lifetime. RFID tags can be divided into five different classes:

- Class 0 – Passive, Read Only.
- Class 1 – Passive, Write Once, Read Many.
- Class 2 – Passive, Multi Write and Read.
- Class 3 – Active, Multi Write and Read.
- Class 4 – Active, Networking Tags.

2.3.2. Frequency bands

RFID runs at different frequency bands of the electromagnetic spectrum. These bands fall under the spectrum of ISM (Industrial, Scientific and Medical Bands), which are portions of the electromagnetic spectrum that are license free and can be used like Wi-Fi and Bluetooth, for instance. To illustrate the various frequencies used on different applications and the advantages and limitations of using these frequency bands the next table shows some commonly used applications and tag types.

Table 1 - Differences between several RFID frequencies

Band + Frequency	**Read Range**	**Advantages**	**Application**
Low Frequency (LF) 30-300 KHz	Up to ~ 50cm	+ Good penetration in moist environments + No Anticollision - Slow data rate	Animal Tracking Access Control Vehicle Locks
High Frequency (HF) 3-30MHz	Up to 1m	+ Good penetration in moist environments - Poor performance in metal environments	Item tagging Libraries Smart Cards Airline Baggage
Ultra-High Frequency (UHF) 300-3000MHz	Passive up to ~12m Active up to ~6m	+Fast data rates + Good performance in metal environments	Supply chain uses Baggage Handling Toll Collection
Super-High Frequency (SHF) 3-30GHz	Up to 2m	+Fast data rates + Good performance in metal environments	Item Tracking Toll Collection

		- Poor performance in moist environments - High Cost	

2.3.3. Standards

RFID tags can have several identification standards, depending on their use and can even have the manufacturer's own identification that does not follow any standard. The following standard is the most used in UHF tags.

UHF RFID tags are usually identified by the EPC (Electronic Product Code). This name is a standard proposed by the Auto-ID centre which has two different types, a 64-bit and a 96-bit code. The 96-bit code has a unique number to 268 million companies, with 16 million different object classes and 68 million serial number in each class [19]. The 64-bit version is a compromise between the cost of the tag and the number of different codes. This version offers low cost but fewer serial numbers.

The EPC, as shown on Figure 9 [20], is a number made up of a header and three sets of data. The header specifies the version of the EPC used, the second part specifies to manufacturer's code, and the third part identifies the type of product which is usually the SKU (Stock Keeping Unit) [21]. The main difference to the barcode is the last part, the serial number. This serial number identifies a single product, instead of a type in product.

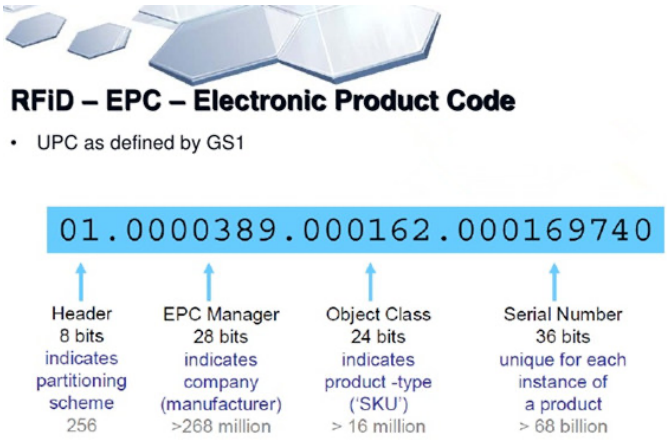

Figure 9 - Tag EPC

RFID technology also has some limitations that can be classified into 4 main categories [22]:

- **Technical problems** – This technology works using radio-waves that are always prone to reflection and refraction. For instance, radio-waves can be absorbed by water and distorted by metal. These situations can interfere, and the communication can fail. However, these problems are dependent on the environment. Interference, for example, in the same coverage area can be managed by an anti-collision scheme, such as TDMA (Time-division multiple access) used by Auto-ID centre. Also, for tag collision, when a reader can read fifty tags a second, the tags can respond one after another.
- **Standards** - When several companies try to develop new technologies, it often happens that, without standards, companies develop unique ways of communicating which can result in higher costs, effort and time. To make hardware and software compatible is very important to have standards in the industry created by ISO (International Standards Organization) and EAN.
- **Costs** – In business environments attaching RFID tags to cases, pallets of even items can be inexpensive, as the tags are very cheap. But, implementing solutions with readers, software, integration and the implementation of such technologies can have very high costs.
- **Privacy** – Companies and retailers must be careful when implementing RFID technologies as consumers can be tracked all the time and everywhere after

they purchase a tagged product. This creates fear from these kinds of technologies.

2.4 RFID in Location

Radio-waves in indoor environments always have propagation problems. These problems can be multipath, rare line-of-sight (LOS) path, absorption, diffraction, and reflection. These kinds of problems make the signal reading unprecise and can lead to errors in locating tags. To resolve this, there are several localization algorithms that can be classified into three major categories:

- Distance Estimation.
- Scene Analysis.
- Proximity.

2.4.1. Distance Estimation Algorithms

Distance Estimation algorithms use triangles to estimate the target's location. This technique consists in measuring the angle of incidence, or Angle of Arrival (AOA), with at least two references resulting in convergence point given by triangulation shown in Figure 10[23]. The position that is estimated results in the intersection of the lines defined by the angles.

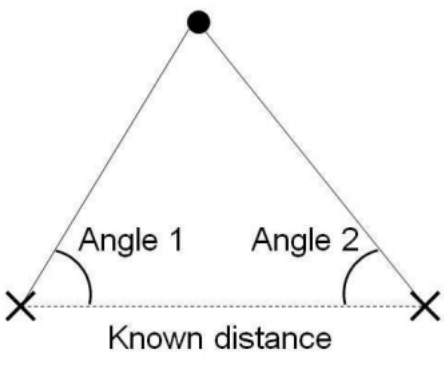

Figure 10 - RFID tag triangulation

Other techniques for locating objects can be for example Received Signal Strength (RSS), Time of Arrival (TOA), Time Difference of Arrival (TDOA), and Received Signal Phase (RSP) [22].

These techniques are:

- RSS – The location of the tag is given by the attenuation of emitted signal strength between the distance of the emitter and the receiver illustrated on Figure 11[24]. The tag can be localized with at least three reference points and the corresponding signal path losses due to signal propagation. However, this algorithm is very susceptible to environmental noise, multipath propagation, and many more making it very unreliable [25].

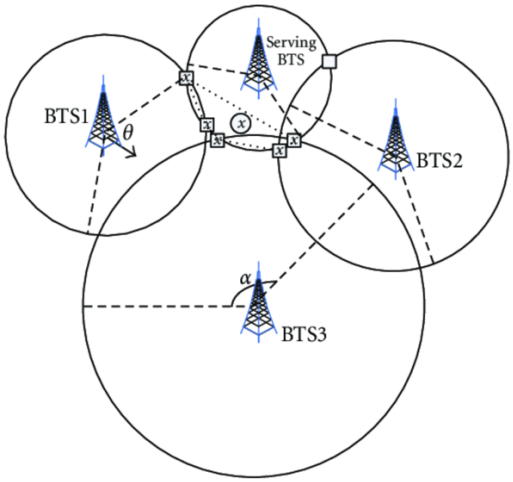

Figure 11 - RSS Technique

- TOA – This algorithm measures the time it takes for the signal to come back from the tag, this means that the distance between a reference point and the tag is proportional to the propagation time signal depicted on Figure 12[26]. TOA systems need at least three different measuring units to compute a 2-D position. It also carries the disadvantage of requiring that all receivers and transmitters to be precisely synchronized and that the signals transmitted carry timestamps to accurately compute the tag distance to the reader [27].

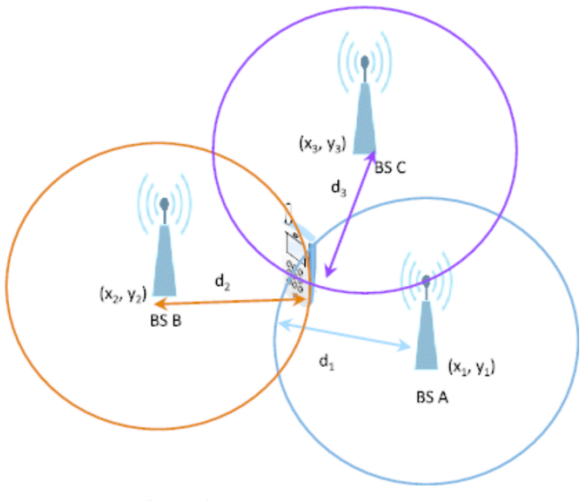

Figure 12 - TOA Technique

- TDOA – A tag is located by determining the relative location of a targeted transmitter by using the difference in time at which the signal emitted by a tag arrives at multiple measuring units. If there are three receivers, there will be two TDOAs and thus have an intersection point that can be used to estimate the location of the tag [28], portrayed on Figure 13[29]. This algorithm carries the same problems as the TOA.

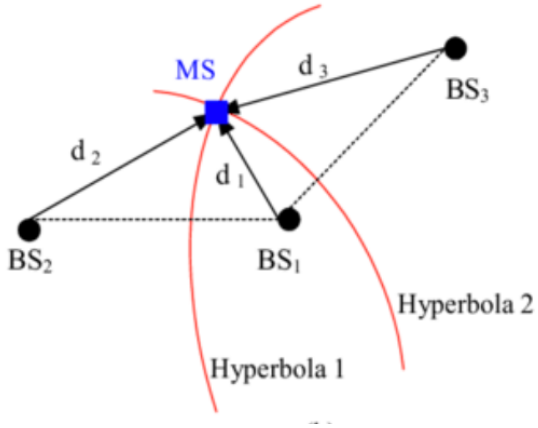

Figure 13 - TDOA Technique

- RSP – This algorithm is also known as POA (Place of Arrival). It uses a delay, expressing the signal's wavelength as a fraction to estimate the distance. The transmitters must be at specific locations, and they must emit pure sinusoidal signals. The distance is then computed using the same algorithm used on TOA or TDOA. The disadvantage is that this algorithm strongly needs a LOS to try and limit the localization errors [22].

2.4.2. Scene Analysis Algorithms

Scene analysis algorithms have two steps. The first one being the information concerning the environment (fingerprints – which are anchor tags placed in strategic points [22]) are collected. Then the location is estimated by matching the online measurements with

the appropriate set of fingerprints. The two main algorithms are kNN (K-Nearest-Neighbour) and probabilistic methods.

The algorithms are:

- kNN – This method consists in firstly measuring the RSS value of the tags at known locations to build a database of RSS, which is called radio map. After that, the measurements given by the reader are used to search for k closest matches in the radio map previously built, shown on Figure 14. In the end, root mean squares errors principle is applied on the selected neighbours to find the estimated location of the tag [30].

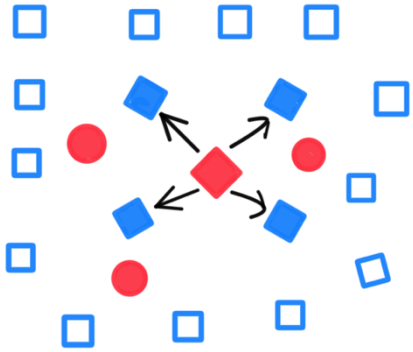

Figure 14 - kNN Technique

- Probabilistic Approach – This approach tries to find the location of a tag by assuming that *n* possible locations and one observed signal strength vector, as shown in Figure 15 [31]. So, the location with the highest probability is chosen. These approaches usually involve calibration, active learning, error estimation and tracking with history to more accurately compute the tag's location [32].

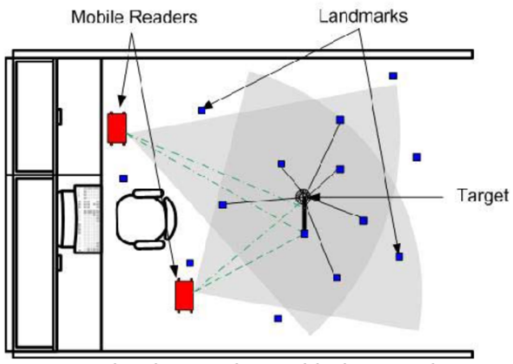

Figure 15 - Probabilistic location

- Proximity algorithms are, as the name implies, based on proximity (depicted in Figure 16 [33]). This approach relies mainly in the antenna's power. When a tag enters the range of a single antenna, the location of the tag is the same of the receiver. When one or more antennas detect the tags, the target is assumed to be inside the range of the antenna with the highest signal. This is the easiest approach to implement but it is also the least accurate one.

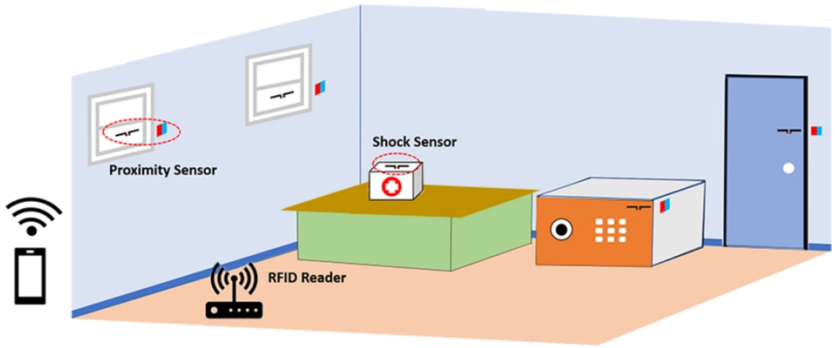

Figure 16 - Proximity Technique

2.5. RFID Location in the real-world cases

This chapter shows real world applications and commercial products that use RFID technologies.

2.5.1 FlyDubai tracks the planes seats cleaning

Airline FlyDubai is using RFID technology to track the cleaning cycles of each seat on the airplanes to maintain the best standards of flight hygiene safety. The software, called RFIDAeroTrack, is developed by an Irish company – Aerospace Software Developments (ASD) – and increases the efficiency of seat covers cleanliness by attaching an RFID tag to each seat cover, providing a unique identification and that enables the airline to keep track of the seat covers at every stage of the cleaning cycle and lifespan. The seat covers are scanned when taken of the airplane to be cleaned and the scanner tells the operator how many uses it has left. If it reaches the maximum lifespan of cleaning cycles it gets taken out of service. If not, it gets scanned at every stage of cleaning to then be put back on the airplane. It also helps maintaining the cost of managing all airplane seat covers to a minimum and it has been used to ensure to COVID-19 measures are being put in place aiding compliance with all safety regulations [34].

2.5.2 Indian Railways Installing RFID Tags to Track Entire Fleet of Wagons

Indian Railways are currently dealing with shortage of wagons, locomotives, and coaches. They are installing RFID tags on all their trains to always keep track of them. Currently, with all operations still being performed manually, a lot of errors still occur. Using cutting-edge technology like this, lowers the cost that is associated with human

errors and enables a better train operation and management. The project is being developed by Centre of Railway Information Systems (CRIS) [35].

2.5.3 Lowry solutions

Lowry, a company based in Michigan, USA, is using RFID technology to manage and locate important assets on businesses. Previously maintaining and managing a business inventory was very long and a costly task to perform. Usually, companies track their assets by using serial numbers, spread sheets, or even tracking all of the equipment with barcode scanners. This solution works on all kinds of infrastructures, from office equipment to multi-story buildings and wheelchairs to life support systems on hospitals, etc. Companies' personnel use a portable RFID scanner built into a mobile computer to easily scan one or more asset tags without having physically to scan each barcode. They do not even need to physically see the tags. An operator inside a big office room can scan the entire floor in seconds compared to a simpler barcode scanner on each asset. By using tags with built in memory, a lot more information can be stored on the asset like maintenance information, inbuilt sensor with data that can be read, and many more [36] .

2.5.4 Infsoft localization with RFID

A Germany based company, Infsoft – smart connected locations – is using RFID technologies to offer its clients, like Frankfurt Airport, Swiss Federal Railways, Siemens, Roche, and others, systems like indoor navigation using BLE (Bluetooth Low Energy) beacons, indoor analytics, indoor tracking using RFID technologies and location-based services. They have several use cases for RFID technology, like in supply chain management in the automotive industry by using RFID tags for vehicle identification, quality control in production or asset management. Also, in warehouses, forklifts are tracked using ultra-wide bands (UWB) and have a RFID scanner connected to them to

be able to read the information attached to the pallets. This allows the supply chain to be continuously monitored engaging in process optimization [37].

2.6 OPC

Starting in the mid 1980's scholars and industries started to investigate, develop and establish new networks, protocols and BUS systems for industry communications. Within a few years, dozens of different bus systems existed. Some of these systems became universal communication systems in the industry, while others are used in very specific areas and some of them have eventually vanished. For instance, Profibus (created by the German department of education and research later mainly used by Siemens) is often used in process industries, CAN-bus (Controller Area Network) is the standard for the automotive industry and office networks are mainly based on Ethernet. In a communication system, different devices must exchange information like a Human Machine Interface (HMI) or Supervisory Control and Data Acquisition (SCADA) system. These systems must collect, analyse, display data from various devices while using different protocols and networks. As a result, to each device a software driver must be provided by the device company, making these systems complex and fault prone. A diagram of these systems is shown on Figure 17 [38]. Implementations like these mean that if a protocol specification is changed it results in malfunctions in the communication and leads to adjustments which are time consuming and expensive.

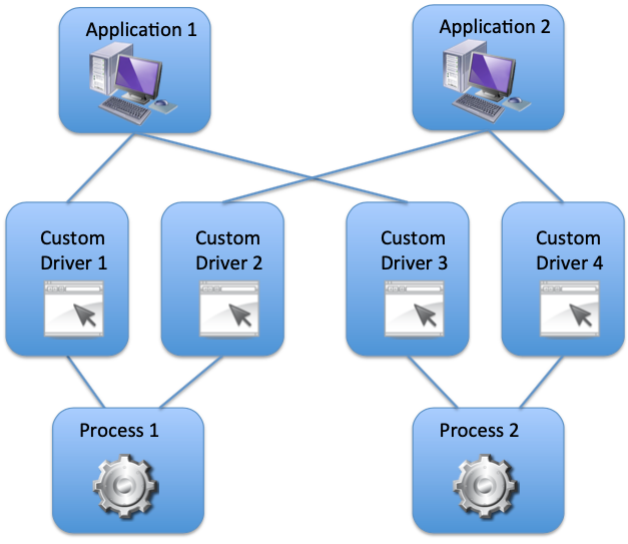

Figure 17 - Communication without OPC

In 1995, Fisher-Rosemount, Intellution, Intuitive Technology, Opto22, Rockwell, and several others, formed a task force to find a solution for these problems. After a few months, a new standard was established describing the server-client software architecture to collect real time process data from devices and pass them to, for instance, SCADA systems. The first specification for this protocol was established in August of 1996 called OPC-DA 1.0 (Data Access) and the OPC foundation was founded to continue, maintain, and market this new standard.

OPC (OLE for Process Control), OPC-UA's original name, was a client-server architecture based on Microsoft's OLE (Object Linking and Embedding) which was a technology to develop object orientated structures and was later combined in COM (Component Object Model) and DCOM (Distributed Component Object Model). Even though this protocol was widely used in the industry it also had several setbacks.

Security aspects were also not taken seriously in the 90's and the implementation of security was on the operative system, which is nowadays a very wrong practice. Data provided by two different OPC servers architectures could not be grouped together. Although the different OPC client-server architectures were and are still acceptable in process and automation industries, in 2006 a new concept was created that combined the different separate standards into one, that uses the same address space for all the different OPC client-server architectures. This new standard also integrates security and new characteristics like fault tolerance, redundancy, interoperability, and it uses web-based technologies.

2.6.1 OPC-UA

OPC-UA is the successor to the original OPC Classic specification. OPC Classic had some limitations in the early 2000's, by restricting the platform to Windows, not being able to handle network connections very well and security problems as well. This was mainly caused by OPC being defined in parts and collaborations between several groups and technologies making the various OPC facets (such as Data Access, Alarms & Events,

etc) expose their services in different ways. The OPC foundation started to work on the OPC-UA unified architecture to try and fix its predecessor's problems. So, in 2006, the first parts of the OPC-UA architecture were published. Being designed to be open-ended and configurable it was expected to be suited for all kinds of applications where the classic OPC was not.

OPC-UA is the new architecture that exposes all its facets using a single set of services. This architecture follows the Service Oriented Architecture (SOA) paradigm in which a function is based on the concept of services. These services are defined in the specification as "Client Callable operations". A server can implement all the services, or just a subset of them.

OPC-UA is also language and operative system independent, making it cross-platform. The OPC foundation provides a .NET, Java and ANSI C version but it can be, as previously said, coded in all types of languages and devices.

The most basic concepts of OPC-UA are its transport mechanisms and data modelling. For data transportation, OPC-UA, can use any protocol but the first version has two technologies already mapped:

- UA TCP (which uses the TCP protocol)
- Web Service protocol (which uses mainly the HTTP/HTTPS protocol).

Data modelling defines the tools to expose the information in OPC-UA. This specification defines an Address Space which contains the entry points and the types that are used to build a hierarchy that can also be extended by vendors and organizations.

The source code and examples are available on OPC's Foundation GitHub page [39].

OPC-UA is a communication protocol divided into many specifications [40], each different and independent from another, which are:

- Part 1 - Concepts
- Part 2 – Security Model
- Part 3 – Address Space Model
- Part 4 – Services
- Part 5 – Information Model

- Part 6 – Service Mappings
- Part 7 – Profiles
- Part 8 – Data Access
- Part 9 – Alarms and Conditions
- Part 10 – Programs
- Part 11 – Historical Access
- Part 12 – Discovery
- Part 13 – Aggregates
- Part 14 – Publish / Subscribe

The two important parts to understand the model and access information are parts 3 (Address Space) and part 4 (Services). Part 3 is the main part used for the development and design of OPC-UA applications. Which means that if we want to use this protocol, this is the only part that we need to configure to have a running OPC-UA server. Part 4 represents the interactions between the client and the server in a more abstract way. The client uses the services to find and access the information that is being provided by the server. These services are abstract because they define the information that is exchanged between UA applications.

Each of these specifications lay on each other, and every and each one of them has a unique function. Part 5, for instance, provides the framework for all information models like the entry points to the address space, the base types and the more complex types like objects and data types [41]. Our focus will be address space because it will be where we will specify our tag model. The main purpose of the address space is to provide a standard way for servers to represent their objects to all clients. It is possible to have variables, objects, classes, methods and relationships between the different objects [42]. Usually, a simple address space is represented by a folder (tree structure) (for instance, the name of a machine), and the variables which could be temperature sensors, current temperature, switch on/off, etc.

To access the address space the client only needs to specify the server's URL (Uniform Resource Locator) - opc.tcp://Server - for the standard TCP protocol (Transmission Control Protocol) and "http://Server" for web services use. When the client connects to the server it can view the server's address space which contains all the

variables given by it. The clients can now subscribe these values to monitor them. It does not work like polling or publish/subscribe. When the client subscribes to a value it can specify if it wants to watch for data changes, events of objects or aggregated values. When a value is changed, the server automatically sends an event telling everyone subscribed that the value changed, making this type of service more optimized instead of being constantly refreshing the values.

The address space, as previously said, it is the models are defined. Most PLC (Programmable Logic Controller), Sensors, and CNC (Computer Numerical Control) manufactures are now releasing their own OPC-UA certified models made in conjunction with OPC foundation [43]. This means that they release an XML file containing all the sensor available, their values and data types, that are then imported to the OPC-UA server to have access to all the data, objects, and other things from the machine to be available to all the clients.

2.6.2 Address Space

The Address Space is the collection of information that an OPC-UA server makes visible to its clients [44]. The base component of the Address Space is called a node, which is used to represent all the available information in the server. These nodes can also be connected to each other using references. For instance:

- "Organizes Reference Type"- is used to build hierarchies in the Address Space.
- "HasTypeDefinition" is used to define the instance node types.

These make the Address Space an interconnected mech of nodes connected by all these different references [45].

The structure of the Address Space, which usually represent real world entities is used organize the information. An Object can contain variables, methods which are used to invoke actions and events as can be seen on Figure 18. Each server has common nodes. The basic common nodes, as shown on Figure 19, form the base of server's object hierarchy. These

nodes are mostly used as entry points because they are present in all OPC-UA's servers. They can be browsed just like folders, and they organize the Address Space according to the NodeClass of the Node.

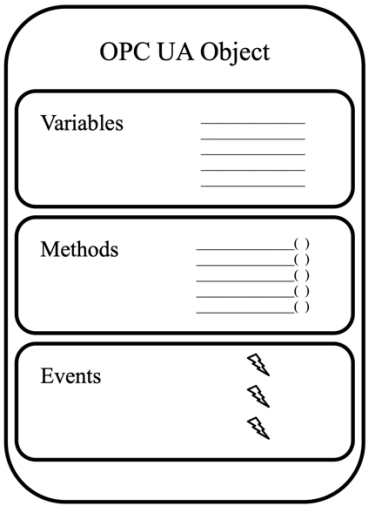

Figure 18 - Address Space Object

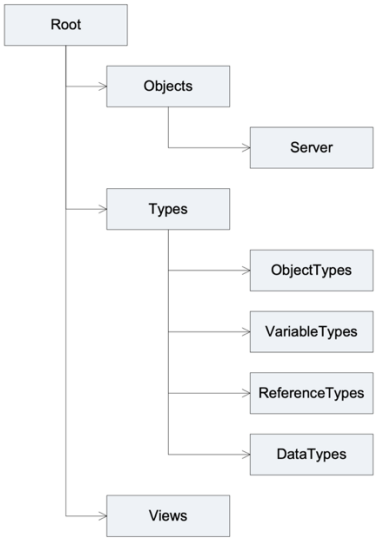

Figure 19 - Address Space's Object Hierarchy

There are eight NodeClasses that are used in the Address Space, and a node needs to always belong to one of them. These eight nodes have their own purpose:

- Objects – represent real world entities that organize the Address Space.
- ObjectTypes – are the Object's abstractions, like for instance a model. This node specifies the structure that all the Object instances should obey.
- ReferenceTypes – are the references between the nodes. Each reference has a target and a source Node and the ReferenceType defines the relation between the Nodes.
- Variables – they hold the data for the related objects and can also have sub-variables of their own.
- VariableTypes – the several variable types.

- Methods – these are the callable operations, which are the “actions” of the Object. For instance, turning on or off a machine.
- DataTypes – are the basic types that data can be represented, such as integers, strings, floats, etc. There is also the possibility to specify custom data types.
- Views – can be a subset of the whole server Address Space, which is used to reduce clutter if there is only the need to use a specific part of the Address Space.

All these nodes have a NodeClass, which is the top of the tree, and it is not extensible, meaning that no new NodeClasses above can be defined. The several NodeClasses available are shown on Figure 20.

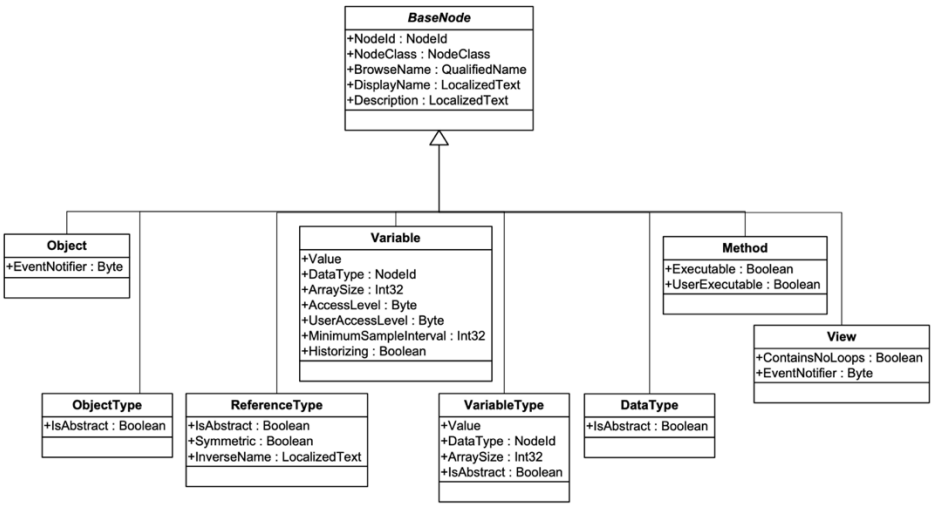

Figure 20 - NodeClasses and their abstract parents

The NodeClass field is mandatory, as are NodeId, BrowseName and DisplayName. The difference between BrowseName and DisplayName is that BrowseName is used to search the Address Space, meaning that all instances of a type will have the same BrowseName. DisplayName is not restricted, and is used to display, for instance, to the operator. The most important attribute is the NodeId one. Each of the nodes are identified by a unique NodeId, which is divided into three parts:

- Address Space Index – which is translated into an address space URI by the server.
- Identifier Type – An enumerated identifier type.
- Identifier – It can be divided into one of four types: numeric, string, GUID or opaque (byte table).
- ExpandedNodeId (optional) – address space URI in addition to the index.

2.6.2 OPC-UA Security

The OPC-UA security model and mapping is presented by part 2 and part 6 of its standards. This protocol can be used from high-level enterprise management to low-level direct process control. The use of it can involve data from both customers and suppliers, this can be an attractive target for industrial espionage or sabotage and can be exposed to all kinds off threats such as malware. These threats can be Message Flooding, Message Spoofing, Message Replay, Malformed Message, Server Profiling, Session Hijacking, Rogue Server and Compromising user credentials. Any disruption in a system like this can have an economic cost to the company and it can also have employee and public safety consequences.

The server communicates with the client by standard web services that are supported by both. These standard web services are:

- Secure Channel Service – used to build a secure communication channel and negotiate the secure strategy the server and the client.
- Session Service – builds and manages the communication between OPC-UA applications. This communication is built on a secure channel and provides communication access for all the other services.
- Node Management, View and Attribute Service – is used to operate and manage the Nodes in the Address Space. Which means that the client can get a node information from the server and can also set its value or object.

- Subscription Server – is used to subscribe the data from the server to the client. Using this service, the client can get the data from the server periodically.
- Method Service – sets the invokers implemented by the objects in the server's address space.

To guarantee the safety of the data, OPC-UA communication must meet a set of criteria, which are – Authentication, Authorization, Confidentiality, Integrity, Auditability and Availability. For this purpose, OPC-UA defines a security model illustrated in Figure 21[46].

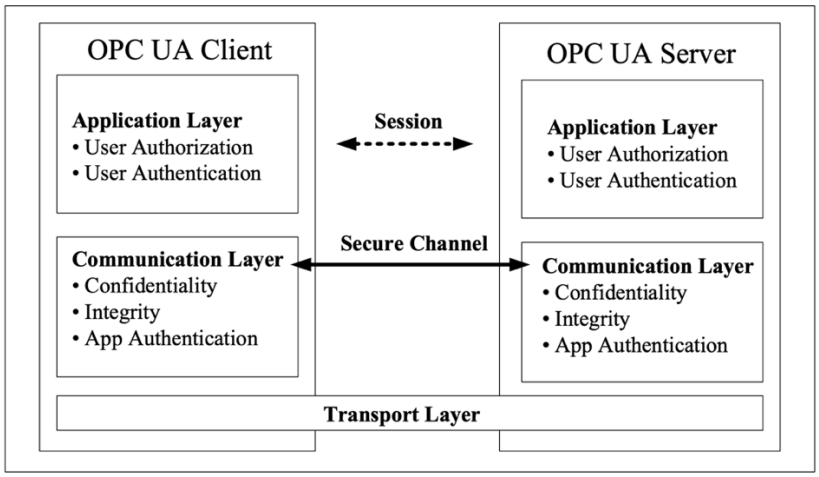

Figure 21 - OPC-UA Security Model

In this model, the communication layer provides the security functionalities that meet confidentiality, integrity, and application security as its main objectives. The client and the server negotiate the functionalities and create a secure channel. This channel provides encryption to main the confidentiality, it uses signatures to maintain integrity, as well as certificates to ensure application authentication for data that comes from the application layer and then passes secured to the transport layer. These functionalities are provided by the Secure Channel Service.

The data regarding information, settings and commands is done in a session on the application layer. The application layer also manages the security functions of user authentication and user authorization. Basically, the application layer establishes a secure

session, the data is then passed to the communication layer and is signed and encrypted (X.509 certificates) and its then passed to the transport layer to transfer it to the network.

The network environment together with the security requirements of the transferred data determine the security strategy which should be applied to the server. For the different implementations the OPC-UA server can be built on a local network, on a virtual private network (VPN) or over the internet. This implementation can also have role-based authentication and privileges as well as privileges varying by the way it's being accessed. For instance, when accessed through a local network users can have writing privileges and when accessed through the internet it can be set to read-only. This also makes possible for users to have writing privileges and users to have only read-only privileges.

2.6.3 OPC-UA Future

Currently the OPC Foundation has released 14 parts of the specification. To solve several problems encountered as the technology is used and evolves OPC foundation has released, for instance, the last part recently and the previous 2 parts after the first specifications. With the release of new technologies OPC-UA is gradually being used in the industries but the transition can be a very slow process because in many installations there is no need to upgrade to newer technologies of the upgrades can become very expensive. To fight these setbacks several wrappers were created to allow for a classic OPC server to work as an OPC-UA server, but they create problems due to maintenance and security [45]. There is also the possibility to use these wrappers to fully migrate a classic instance into an OPC-UA server [45].

2.6.4 OPC-UA Alpha Ventus

Alpha Ventus Offshore Wind Park, as shown on Figure 22, is Germany's first offshore wind farm. It is situated 45 kms north of the island of Borkum. Each of the 12 windmills is designed to be autonomous, fully automated, and self-monitored. These offshore windmill parks, built far from the shore have a small environmental impact. The conditions that they run can be very extreme, because of the salt, humidity, and extreme temperatures, so these control systems must always guaranty the most effective operation. Even though these are fully automated, operators need to constantly monitor the plant and the equipment status remotely from a control room. OPC-UA was the protocol chosen to connect their windmills to the control room, because of the inbuilt security and authentication mechanisms. The windmills are controlled using PLC's that then are connected to a SCADA software using OPC-UA protocol [47].

Figure 22 - Alpha Ventus Offshore Wind Park

3. Technologies Used

In this chapter the hardware and software used in the development of the present project will be presented. It will start with the presentation of all the hardware evaluated. Subsequently, will be presented the necessary software for the correct identification of objects, information processing and position calculation. Finally, the protocol used to send information to the central server is presented.

3.1. Hardware Used

For the RFID system, a reader, an antenna, and several UHF passive tags are required. For acquiring information from the reader, processing and calculating the position of the tags, a computer is needed. First, a Raspberry PI (RPI) was chosen but due to several problems and incompatibilities encountered with the reader's proprietary software, it was later replaced by a PC (Personal Computer) stick.

3.1.1. Raspberry Pi

A Raspberry Pi, as seen on Figure 23, is a series of small single-board computers developed by the Raspberry Pi Foundation. The project first started to aid the promotion of teaching computer science in schools and in developing countries. Its success was so great that people started using it for other applications like robotics as it was very small/compact, and its processing power was very good for such a small computer. The one used of this project was the version 2 Model B, released in 2016. It runs on an Advanced RISC Machine (ARM) architecture clocking on 0.9GHz and had 1GB of Random-access memory (RAM). The cost is on average 50€. It counts with 4 Universal Serial Bus (USB) interfaces to connect peripherals (for this project, the reader), a High-Definition Multimedia Interface (HDMI)

interface and Ethernet to be able to connect to the network. The next version already supported Wi-Fi and Bluetooth without the need of external hardware.

This computer was used to run the reader's software to read ID from the tags. It was chosen because it is low cost and easy to use. It was later discarded because, as it is runs on ARM architecture and our software needs a x86 processor to run it. This will be later discussed on the development chapter.

Figure 23 - Raspberry Pi

3.1.2. PC Stick

A PC stick is as single-board computer that normally as an end with a male HDMI connection to be connected directly to a computer monitor, television set, etc. Inside its elongated case it has a CPU (Central Processing Unit), memory, disk (usually eMMC), USB ports for peripherals, Wi-Fi and many other features. This computer is based on Intel's Compute Stick. It was made to be used as a media centre for applications.

The first prototype used a stick PC, shown in Figure 24. This PC had a x5-Z8350 Intel Atom that can reach 1.92GHz. It had a 2GB of RAM and 32GB of eMMC storage. It also counted with Wi-Fi/Bluetooth, MicroSD slot for expansion and two 2.0 USB ports. The CPU ran on x64 architecture which can run Windows 10 very easily considering its small size (111x38x13.1mm).

Figure 24 – PC Stick for development

The final product used a stick PC, shown on Figure 25, like the one used for the prototype but with different characteristics. The stick has a CPU x5-Z8300 capable of reaching 1.84GHz, 2GB of RAM and 32GB of eMMC storage. It measures 113x38x12mmand it is slightly less powerful, but it can run the project's software easily.

Figure 25 - Stick PC for production

3.1.3. RFID Reader

The reader module is the one responsible for reading the RFID tags. It can be connected to any type of antenna, from short to long-range depending on the application. In the present project, it is connected to a long-range antenna. The chosen reader is made by a

Chinese company called Chafon specialized in making RFID technologies. Their product line ranges from antennas, to readers, tags as well as face recognition products and access control technologies.

The chosen reader is present in Figure 26, and the model is MU904 [48]. It works in band frequency from 865-868MHz (EU) or 902-928Mhz (US) which are the RFID standard UHF frequencies. This model can read up to 50 tags per second and has a range of 0 to almost 15 meters depending on the antenna. To attach the device, a USB interface, or a serial communication (RS232) can be used. Chafon has a Software Development Kit (SDK) and a demo, to help us developers to easily communicate with the reader.

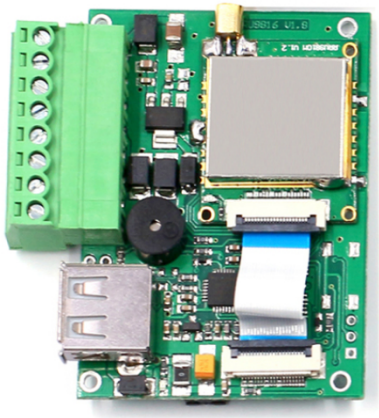

Figure 26 - RFID Reader CF-MU904

3.1.4. Antenna UHF 12dBi (CF-RA1202)

To use with our reader an antenna is required to extend the reader's signal to read the tags. For this purpose, Chafon's CF-RA1202 12dBi [49] antenna was used (Figure 27).

Figure 27 - Antenna UHF 12dBi CF-RA1202

It offers a reading range of 860~960MHz, well according to our reading standards, and a gain of 12dBi. Its structure is rigid, made in fiberglass and its well protected to withstand the elements as its widely used on highways for tolls payment. The theoretical radiation diagram of the antenna, shown on Figure 28[34], will be further analysed in development chapter.

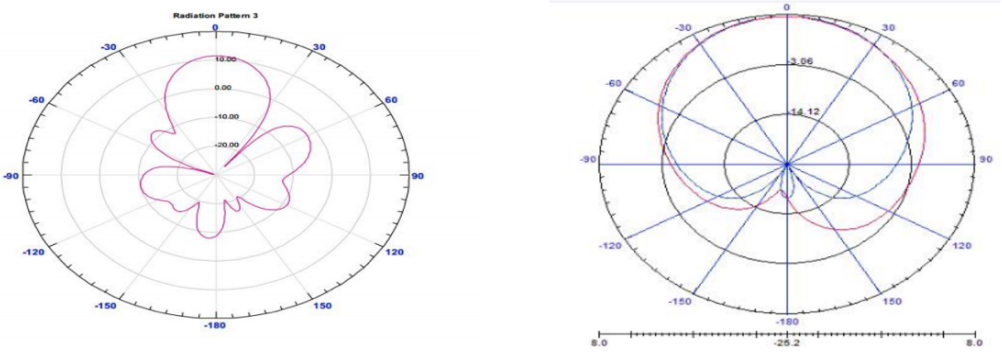

Figure 28 - Theoretical radiation diagram of the antenna

3.1.5. Tags

The RFID tags used in this project, as seen in the Figure 29, are made by the same manufacturer of the reader and antenna, Chafon. The tags operate in the same radio spectrum of the antenna, which is UFH (860-960MHz). These tags are of the passive type. The prototype uses a long range 12dBi antenna, that can read the tags within 15 meters. The tags have a size of 44x18mm. They come in a roll and have a sticker to be easily stuck to a mould even tough that is not how it is used in this project.

Figure 29 - RFID Tags used in the project

To easily attach the tag to a mould, a 3D printed enclosure was developed using Polylactic Acid (PLA), shown in Figure 30. This 3D printed piece has an enclosure where the tag can enter and stand vertically and at the bottom, it has a magnet to easily attach it to a metallic mould.

Figure 30 - 3D Printed Enclosure for RFID Tag

3.2. Software Used

The software used for this project is .NET Framework to manage the RFID reader, MySQL database to locally store the read tags and OPC-UA for creating an OPC server on the PC stick.

3.2.1. .NET Framework

.NET is a software framework made by Microsoft that runs primarily on x86 Windows systems. It was first released in 2002 and includes a large class library called Framework Class Library (FCL) that provides language interoperability which means each language can use code written in other languages. .NET Framework

leads a family of .NET platforms such as mobile computing, embedded devices, other operative systems, and web-browser plugins. The main programming language is C#.

For the present project, this technology was used because the software for the RFID reader, given by Chafon, is built on C# running on .NET. An SDK has been supplied with several examples, can be opened in Visual Studio (the main code editor for .NET Framework) to test the application.

3.2.2. MySQL

MySQL is an open-source Relational Database Management System (RDBMS). SQL comes from "*Structured Query Language*". It organizes data into one or more data tables in which that data can be related to table. It is used for creating, modify and extract data as well as controlling its user access. MySQL, logo shown on Figure 31[50], is currently owned by ORACLE. It is free and open source and works on pretty much every operative system. It is one of the most used database systems in the world today and it will be used to store the position of the tags.

Figure 31 - MySQL

3.2.3. OPC-UA SDK

OPC UA is the successor of OPC. The first one had several drawbacks like frequent configuration issues, it only worked with Microsoft Windows, it had lower security, etc. This second version came to fix all these issues making it better by adding multi-platform, scalability, multi-thread, and security. The main communications protocol it implements is TCP.

The implementation of this communication protocol is a requirement of the project, and its implementation was made using .NET Framework. In the beginning of the project an attempt of implementing it using the language Python was tried, but later it was changed to C# (also used in the RFID reader), to decrease the project's complexity.

OPC Foundation has examples posted on Github for .NET, Java, and C++. The OPC-UA SDK is a library that supports general people in writing OPC-UA clients and servers. The SDK simplifies UA stack APIs, implements commonly used functionalities for most OPC-UA applications, provides base functionality and helper functions, implements security handling and provides samples.

4. Implementation scenarios

This chapter will disclose the project's implementation scenarios, specifically the diagrams in which the several hardware components will be mounted, and which components are going to be used. Four main diagrams are shown, the first three diagrams represent the initial drawings of the current project and the last is the current implementation of the project. One of them has only one reading module and antenna and the communication to the main server is made by an http request using a TCP-IP protocol. The second one uses only one antenna and reader but sends the data collected by the reader to main server by radio waves using the NRF24L01 module. The third uses three antennas, three readers and NRF24L01 modules that communicate with the main one that sends the data to server. The fourth, and last, was the one selected for development which is composed by two antennas with two readers and a computer in the bridge (later used a PC stick instead of a raspberry PI) with the purpose of reading, computing the tag's places, and sending the data through an OPC-UA server.

4.1. First Scenario

In the first scenario has the antenna set at a 45-degree angle on the bridge's crane and the reader directly behind it. The reader is connected to the PC stick and sends the data to the server through the Wi-Fi network or a NRF24L01 using radio frequency. The server then saves the data in its database to later show to users where the tags are.

The biggest complication with this approach is that an X and Y sweep is required. Since the antenna can read a tag approximately 14 meters away, this only happens when the tag is comprehended between 2 meters (approx.) of the Y position of the antenna. So, a factory employee must make approximately three sweeps to successfully read all the tags in the warehouse. Because the crane takes a very long time to move, this process can take a very long time. Figure 32 shows a schematic of this approach.

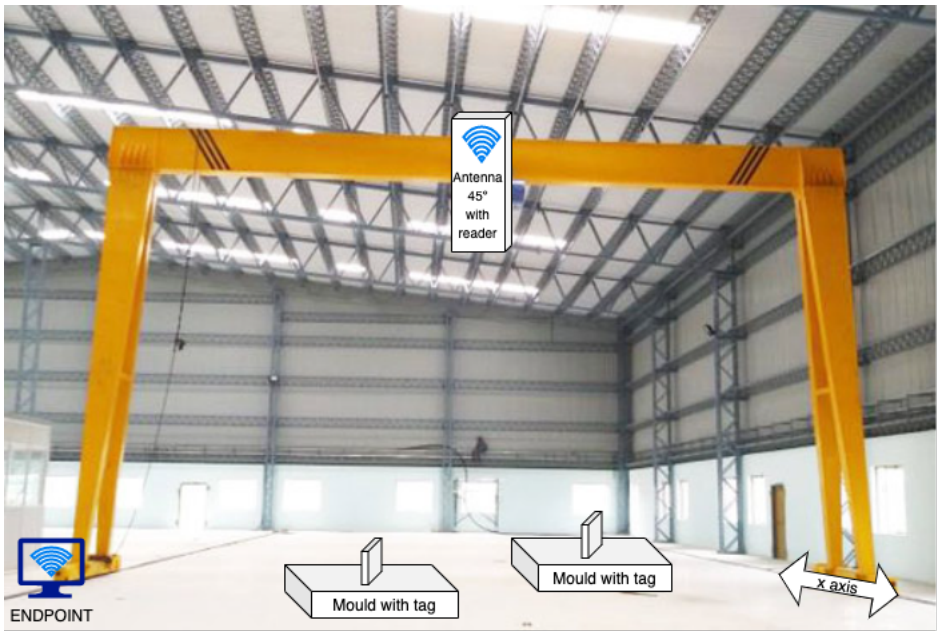

Figure 32 - 1st scenario schematic

4.2. Second Scenario

As shown in Figure 33, by using three antennas, the Y axis sweep is removed. This way, the controller on the bridge only needs to make on sweep on X axis to be able to successfully collect the data from all the warehouses. To be able to make this solution work, NF radio-waves modules were used for each of the readers. The secondary readers (the ones located in the ends of the crane) send their data to the main reader that is in the centre of the crane. This main reader is the one responsible of collecting the data from the other two and send it to the endpoint by a radio wave / Wi-Fi signal.

Figure 33 – 2nd scenario schematic

4.3. 3rd Scenario

This is the chose implementation and its diagram is shown on Figure 34. In this case, two readers and two antennas are placed on the top side of the crane mitigating the need for a second or a third sweep. This way the crane only needs to make one sweep of the whole warehouse to get the tags location. The antennas are set with a 25-degree angle and are two meters apart (This is explained in further detail on the development chapter).

Figure 34 - Final scenario schematic

5. Development

In this chapter all the steps of the project are going to be presented. It will follow a chronological order and it will start with the first steps and lead to the final product.

5.1. Raspberry PI Software

One of the main requirements for this project was that a solution must have been found using low-cost hardware and technologies. The reader and antenna were already chosen, but the remaining of the hardware was not. A Raspberry PI was initially used to begin this project, as the interface between the reader and the server.

Chafon's SDK [51] was developed using .NET Framework and it is only available for Windows x86 systems. This was the first problem. Raspbian (the most

used operative system for Raspberry PI) is a Unix based system and the provided SDK only runs on Windows Systems. One other problem is that the software only runs on Windows x86 systems and RPI is an ARM system.

Various solutions can be used to run a Windows x86 software on an RPI which are software's emulation or, and given the reader has serial interface, a serial communication can be made using that interface.

The first software used is called QEMU [52]. This software is a hypervisor that allows the user to virtualize other OS's – like Windows, Ubuntu, etc - inside the main OS (Raspbian). In order to make this work, QEMU virtualizes an Ubuntu distribution inside a chroot environment with x86 architecture. .NET Framework can run inside a Linux environment even though it only runs on Windows OS's. In order to run a windows application in Ubuntu a software called Wine must be installed adding a layer over the host OS to allow it to run Windows applications. To make it work, gecko and mono which are the required libraries to run .Net software must also be installed. These libraries are an Open-Source implementation of Microsoft's .NET Framework. They are required to run Chafon's software on Linux and to map the COM ports that the reader communicates with the operative system. Unfortunately, a successful mapping between the reader's COM port and the QEMU environment was not achieved. The main disadvantage of using a software like QEMU is that since the CPU of the raspberry PI is not as fast as an x86 system and the software is limited to only one core, the system becomes very slow.

The second software used was Eltechs Exagear Desktop. This program uses that same method as QEMU does, but the main difference is that it maps the whole OS to the virtualized OS making the virtual machine run alongside Raspbian. The process to install is the same and since, the whole machine is mapped to the virtualized system, the COM ports are easily mapped too. This software is better than QEMU because it is made by a company to specifically run Windows software like Office, Skype, Chrome x86, etc. The main disadvantage of it, though, is that since it is made specifically by a company it is paid and because it is an emulation a consistent connection to reader could not be accomplished. The first time the RPI booted it worked with success, but if the connection was closed it could not be restored leaving only the option of rebooting the device. This problem could not be overcome and it

was decided to leave this approach as it could leave the application with problems on a production environment. Currently, this software is discontinued and no longer exists.

Another approach that was attempted that was communication with the reader chip using serial communication bypassing its USB Interface. As shown in Figure 35 the reader contains two interfaces – USB and Serial (RS232) on the pins 3 and 4.

No.	Symbol	Comment
1	VCC	External +9V power supply
2	GND	Ground
3	TXD	RS-232 serial data output
4	RXD	RS-232 serial data input
5	GND	Ground
6	GPIO1	GPIO1 or Wiegand Data 0
7	GPIO2	GPIO2 or Wiegand Data 1
8	GND	Ground

Figure 35 - Reader's connection pins

This approach was found to be difficult to implement as it needed to program the whole SDK by reading bytes and sending bytes to reader. This approach was not pursued.

Finally, and given all these problems, it was decided that Windows would be the solution. So, a small computers was needed (as it will be attached to a moving crane) that could run Windows with the x86/x64 architecture. The SDK comes prepared to work out of the box. A controller must be installed to communicate via USB with

the reader [53], the SDK from Chafon's website needs to be downloaded and the demo project can be run.

5.2. Reading Tests

At the beginning of the current project, tests using Chafon's reader were made using their antenna to figure out the Received Signal Strength Indicator (RSSI) changes by distance and to confirm if the antenna's signal equal to the one Chafon provides.

The first tests were conducted with the tags straight with the antenna, with the antenna resting on a plastic table (to avoid interferences) approximately 1 meter from the ground. The tags were stuck to the top of a wood broom, also to avoid interferences. The major purpose of these initial tests was to try and understand the range in front and on the sides of the antenna. A maximum range of 15 meters with very low accuracy was achieved using this method. It was estimated that the antenna had the best reception with a straight path with the tags and if the length between the antenna and one tag was increased, the antenna could only read with some angle in approximately 5 meters. After the 5 meters the angle of the reading began to close, and it could only read in a straight line-of-sight (as shown in Appendices I).

It also allowed to understand that the tags only work if they are levelled with the antenna's magnetic field (polarization). Which means that, if the antenna is horizontal the tag must also be horizontal, otherwise it will not be read. Since the tags had a vertical stand to be coupled to the mould with the aid of magnets, the antennas had to be mounted on the cranes on a vertical stand.

Since the crane is 5 meters high of the ground, tests needed to be conducted with the antenna on high ground. The tests were conducted on an open area with the antenna set at a 45-degree angle and 3 meters above the ground. A 5x5-meter square was drawn on the ground with intervals of 25cm. A representation can be seen on Figure 36 (please note that this representation is not at scale).

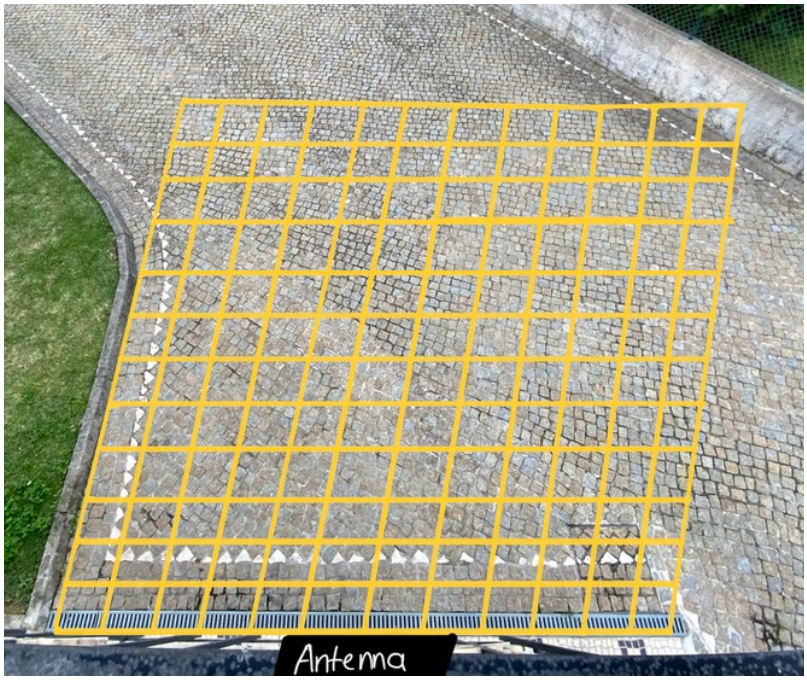

Figure 36 – Antenna testing representation

After that, the RSSI average on each of the intersection points was measured to get the RSSI signal average. The results are shown in Appendix I. As shown on the appendix 1 and Figure 28 (radiation diagram), a pattern in the results can be seen that resembles Chafon's antenna diagram. If the values from Appendix I and the original Chafon's antenna are overlayed we get a very similar radiation graphic, as show in Figure 37 (the magenta line represents Chafon's, and blue line represents testing). These values were later used to better tune the location model explained in the next subchapter.

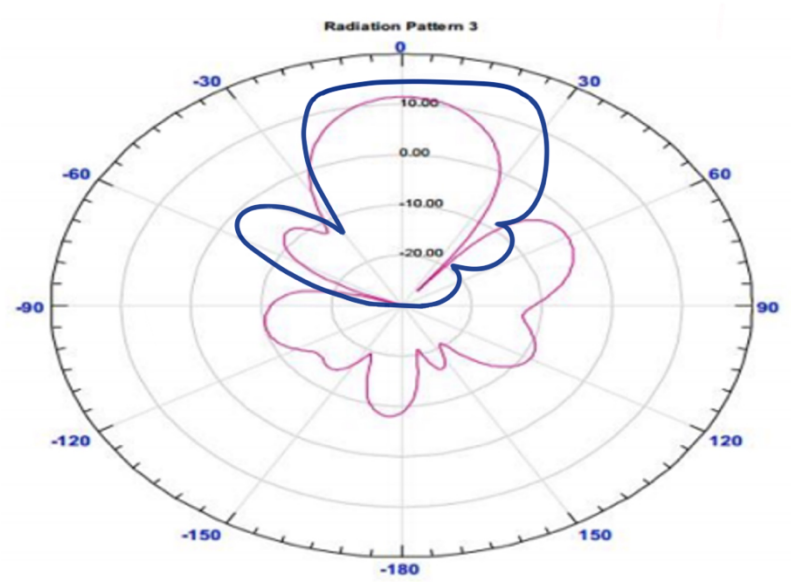

Figure 37 - Chafon's Antenna and testing overlay

5.3. The location model

The mathematical model used to return the location of the tags works by estimating the RSSI value by finding the intersection between the two curves of the antennas [54]. To get to the final equation the test values specified on the subchapter above were used. This equation computes the RSSI value (z) in a bi-dimensional space (x, y) and is given by:

$$y = \frac{L}{\alpha - \beta}\sqrt{(\alpha - z)^2 - \gamma^2(x - x_0)^2}$$

Equation 1 - RSSI Value (z)

The following table specifies the variables in the equation:

Table 2 - Equation values

x	Cartesian X coordinates from the tag to the reader referential (x_0, 0) in meters.
y	Cartesian Y coordinates from the tag to the reader referential (x_0, 0) in meters.
z	RSSI value obtained by the RFID reader.
α	Maximum RSSI value at the coordinate (x_0, 0)
β	RSSI value at the coordinate (x_0, L).
L	maximum range of the RFID antenna.
γ	an integer value depending on the type of the antenna used (short and long range). In this case, the value is 20 for a long-range antenna.
x_0	shift value of the antenna distance over the *x* axis.

Solving the last equation (Equation 1), the RSSI sensitivity *z* is given by the following equation:

$$z = \alpha - \sqrt{\left(\frac{\alpha - \beta}{L}\right)^2 y^2 - \gamma^2 (x - x_0)^2}$$

Equation 2 - RSSI Sensitivity Z

Given Equation 2 and using 0.5 meters step until a maximum of 5 meters the following theoretical and real curves can be represented by Figure 38 [38] and Figure 39 [38]:

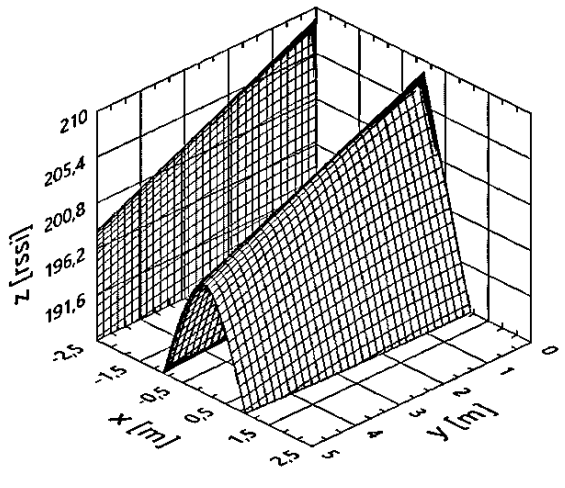

Figure 38 - Theoretical RSSI surface

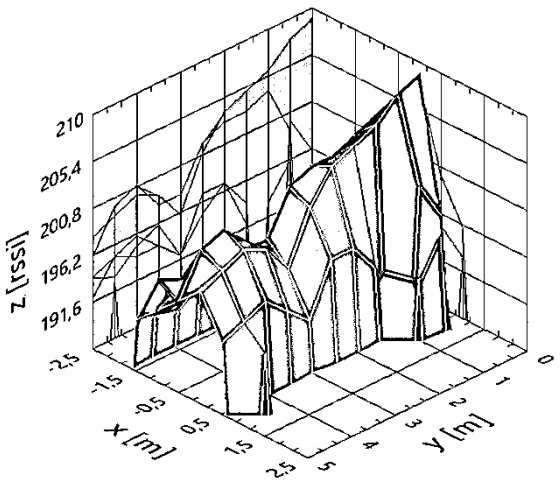

Figure 39 - Real RSSI surface

The above figures were computed using the following values:

Table 3 - Graphic values

α	209 RSSI (max RSSI measure)
β	198 RSSI (min RSSI measure)
L	5 m
γ	20
x_0	0 m and 0.5 m

This mathematical RSSI surface of the antenna is used to estimate the location of the passive tags, when two similar antennas are placed side by side in front of the tags, as shown on Figure 40 [38].

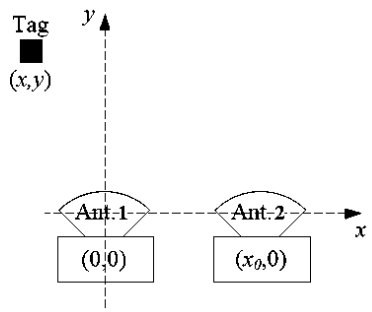

Figure 40 - Two antennas space representation

5.3.1. Intersection curves for tag location

To compute the location of the tags (using Equation 2) a model to represent the intersection of the antenna curves was used, with the antennas side by side. The first antenna is on the coordinate (0,0) position having an RSSI value of z_1 (Equation 3) as shown on Figure 41 [38].

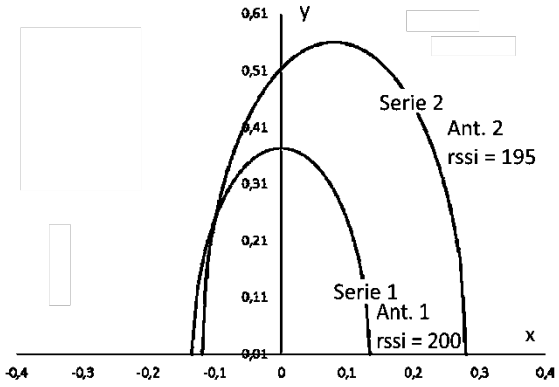

Figure 41 - RFID intersection curves

$$y = \frac{L}{\alpha - \beta}\sqrt{(\alpha - z_1)^2 - {\gamma_1}^2 x^2}$$

Equation 3 - RSSI Value of z1

In figure 24, ”Serie 1” is the RSSI curve of antenna 1, located at (0,0) when its RSSI $z_{1.} = 200$. “Serie 2” is the RSSI curve of antenna 2, located at (0, 0.08) when its RSSI $z_2 = 200$. The tag location is (-0.10, 0.25) in meters in this specific scenario.

The second antenna is located at (x_0, 0) with an RSSI value of z_2 (Equation 4) and is defined by:

$$y = \frac{L}{\alpha - \beta}\sqrt{(\alpha - z_2)^2 - \gamma^2(x - x_0)^2}$$

Equation 4 - RSSI Value of z2

To get the intersection of these two curves (Equation 3 and Equation 4):

$$x = \frac{z_1(z_1 - 2\alpha) + z_2(2\alpha - z_2) + \gamma^2 x^2}{2\gamma^2 x_0}$$

Equation 5 - Intersection of the two curves

5.3.2. Model Results

The next table represents the results when the model was run with real life values. For this example, consider that the antennas are side by side with a distance between them of 50 centimetres.

Table 4 - RFID results

Tag Reading	Antenna at (0,0)	Antenna 2 at (0.5, 0)	x'	y'	x real	y real	Error
1	201	202	0.3	2.5	0.0	2.5	0.3
2	198	196	0.1	4.9	0.0	5.0	0.2
3	194	199	0.6	4.5	0.5	3.5	1.0
4	197	204	0.5	2.2	0.5	2.0	0.2
5	201	195	-0.1	3.6	-0.5	4.0	0.6
Error Average = 0.5 meters							

In Table 4 there are five readings, for five different tags in the space of (x, y). The readings of the antennas are the RSSI values of the tag in that position. The values of (x', y') are the values computed using our model. The error is computed using the real position values and the computed ones. The model outputs an average error of 0.5 meters. There is also the need to have in mind that this test was conducted under very good conditions and not in real world (warehouse floor) conditions. But the results are pretty good, given the small 50-centimeter error. A paper was published to show the results found. The paper is called "Long Range RFID Indoor Positioning System with Passive Tags" [38]. It was co-written with a colleague of mine that is doing the same work as me but in short range, and the guiding professors.

5.4. Implementation

In this chapter the steps made while implementing our solution will be covered. It will start with the implementation of the reader's software followed by the MySQL database for data persistence and OPC-UA Server for publishing the results.

5.4.1. Reader's Software

As previously said, Chafon's software was used for an easier implementation as their software already has all the settings required for communicating with the reader. To use it, it must be downloaded from Chafon's product website [48]. Then, the reader needs to be connected to a computer's USB port and find the communication port by opening device manager.

Figure 42 is the default page when opening the software, bear in mind all the software shown is already changed to suit the needs of the project.

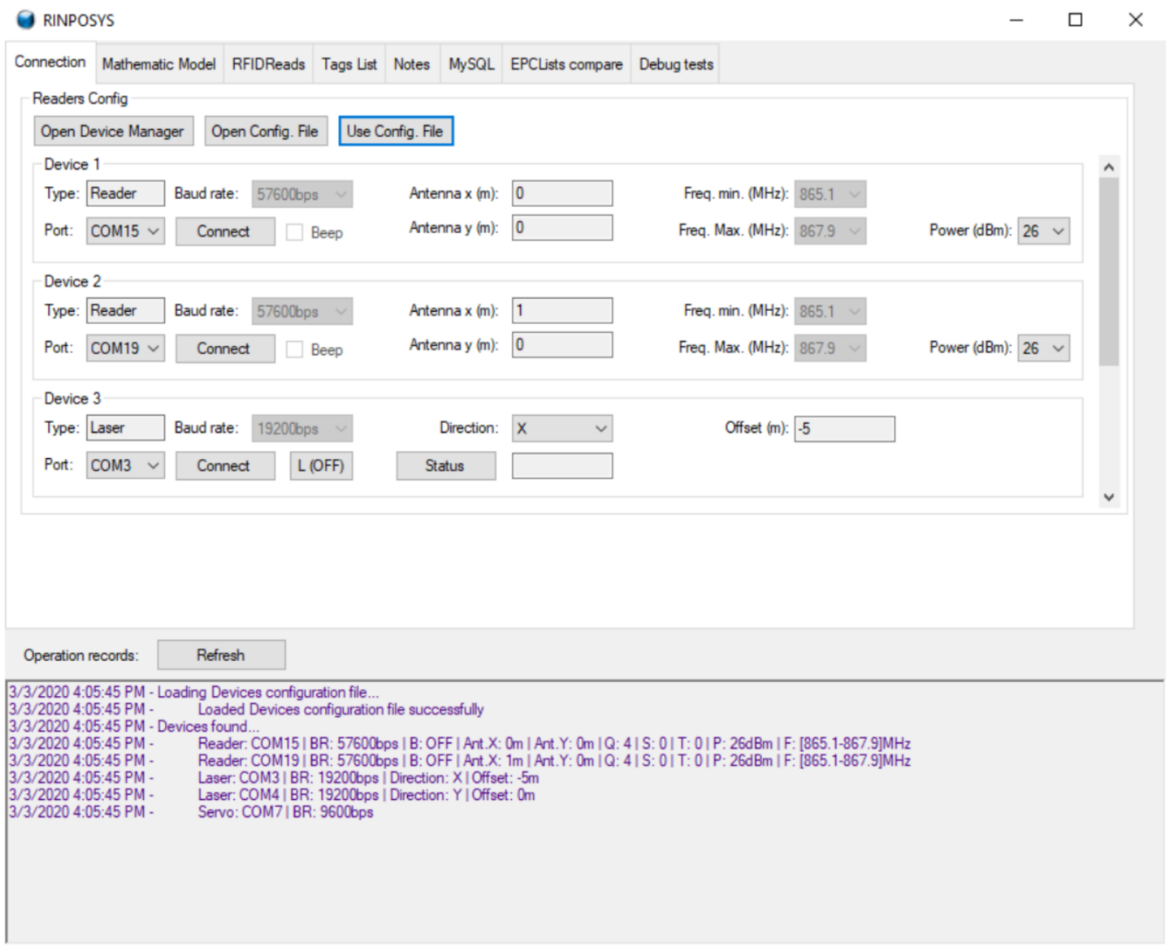

Figure 42 - Rinposys first page

In this tab, the reader's connection information is shown. Previously (using Chafon's demonstration software) all of Windows COM ports were available, and the port that the reader was connected to be the one used in the software. The demo software had a problem, being that the software must first be ran and then the port specified to make the connection. In other words, the software could only be connected to a reader at a time. The software was then changed to allow the connection of one or more. For this purpose, a configuration file was created that stores the readers information and when it runs it reads the file and connects automatically to the specified readers, as shown on Figure 43.

```
##Example UHFReader
#type: <{Reader}>
#port: <{COM[1;50]>
#beep: <{0,1}>
#antenna_pos_x (m): <{Number}>
#antenna_pos_y (m): <{Number}>
#q_value: <[0,15]>
#session: <{0,1,2,3}>
#target: <{0,1}>
#power (dBm): <[0,26]>
#height: <[4,12]>
#angle: <[5,45]>
#freq_min (MHz): <[865.1,867.9]>
#freq_max (MHz): <[865.1,867.9]>
```

Figure 43 - Reader's configuration file example

In this file, the COM port that the OS sets for its connection with the reader, the beep value which makes a beep sound when a tag is read and the antenna position in correlation with the other ones are defined. It can also set the power to be used on the antenna by the reader giving it more reading range or less range depending on the value and the height and angle of the antenna. If all these parameters are correctly set, when the program runs, a successful connection to the reader is made, and the reader's information is shown in the screen.

After that, moving on to the second tab which is the tab responsible for the parameters required to run the mathematical model. As shown from Figure 44, there is also the possibility to create a configuration file to set the values. The parameters, such as, the "L Max" which corresponds to the maximum value of the distance to minimum RSSI value, the

"RSSI min" which is the minimum RSSI value observed during the testing phase as well as the "RSSI Max" and the antenna type value which represents the antenna used can be set.

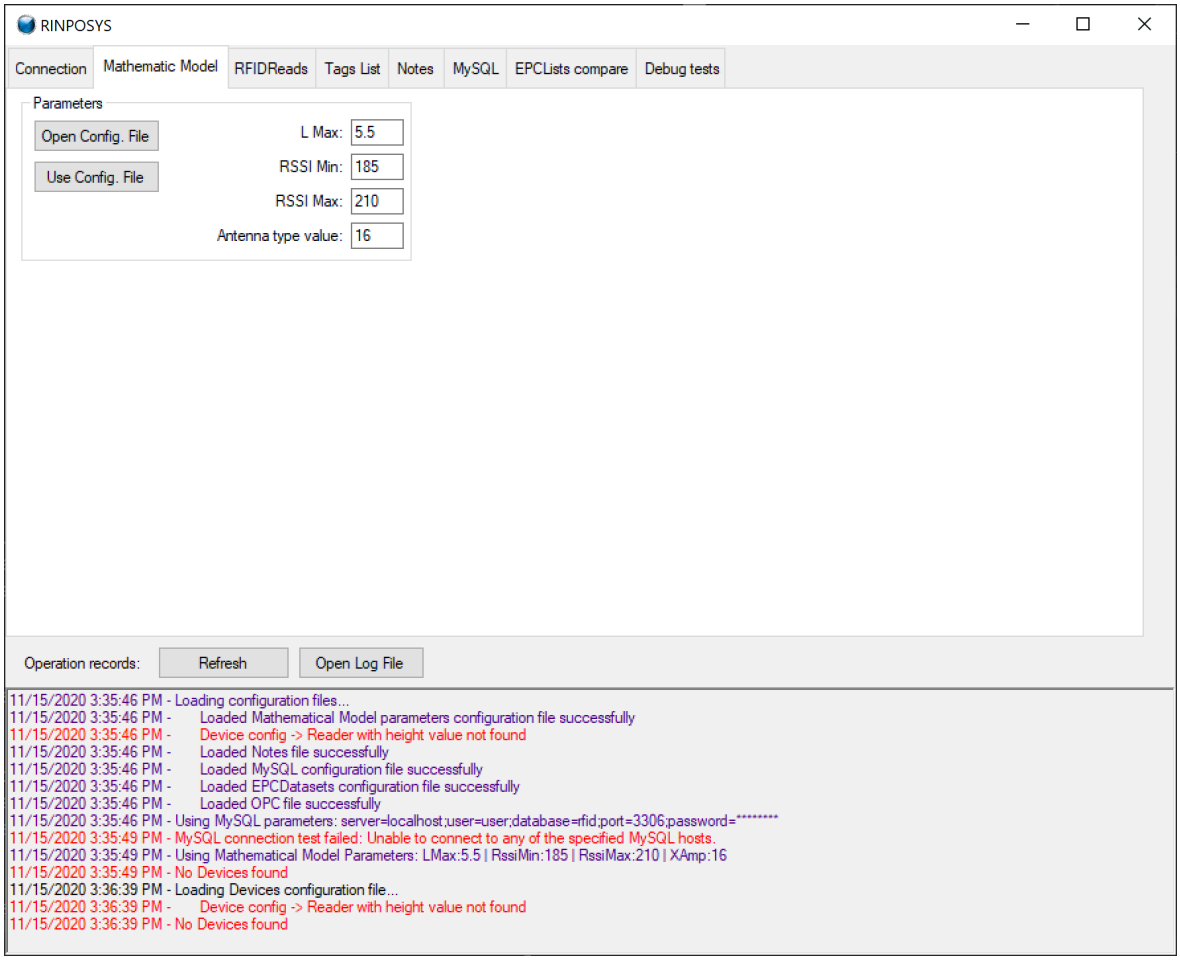

Figure 44 - Mathematical model configuration tab

The third tab "RFIDReads" is where the program can read the tags. In Figure 45, if one or more readers are connected, when the "Start" button is clicked the program will start to read all the tags it can find. The time in which the software reads the tags can also be set, using "Use Time" checkbox and the mathematical model is used using the "Use MM" checkbox which will be explained in the next sub-section.

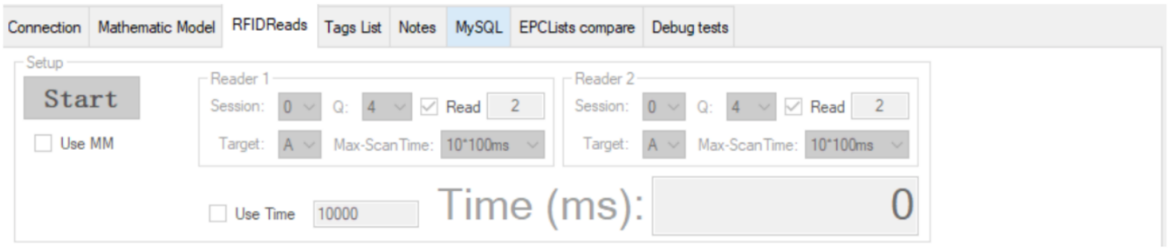

Figure 45 - "RFID Reads" tab

Clicking the start button the read RFID tags start to appear, as shown in Figure 46.

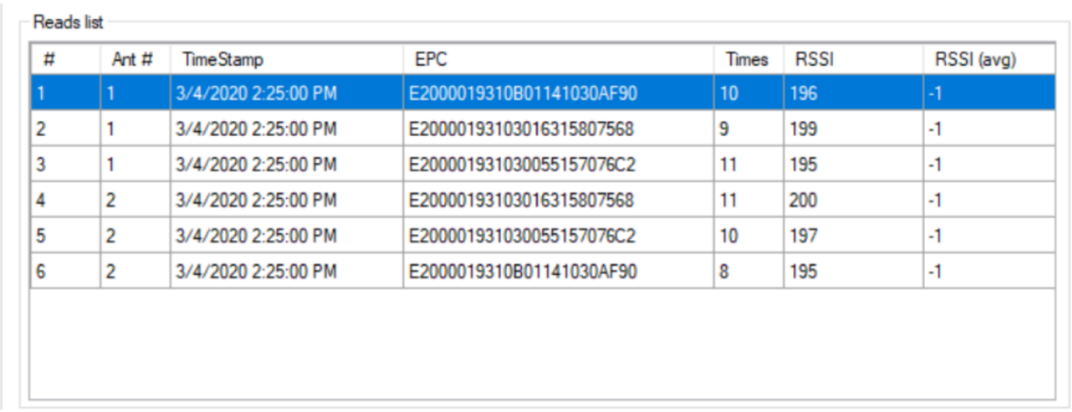

Figure 46 - Read tags list

In the table of Figure 46, there are several columns:

- Ant# - which antenna read the tag.
- Timestamp – the last time this tag was read by that specific antenna.
- EPC – representing the tags unique identifier ID.
- Times – How many times this tag has been read.
- RSSI – The last read RSSI value.
- RSSI (avg) – representing the average RSSI values for that tag. “-1” if there are not enough values and an average value if we have enough RSSI values.

The fourth tab, show in Figure 47, is the tab that shows the tags that already have a computed position, which means that if a tag has been read by two antennas, its position can be computed based on the intersection of the two curves.

The columns in this table are:

- Timestamp - The last time the position has been computed for this tag.
- ID – The identification ID for the tag.
- X (mm) – The X value in correlation to the bridge in millimetres.
- Y (mm) – The Y value in correlation to the bridge in millimetres.
- RSSI (R1) – Meaning the last RSSI value when the position was calculated for the first reader.
- RSSI (R2) – Meaning the last RSSI value when the position was calculated for the second reader.

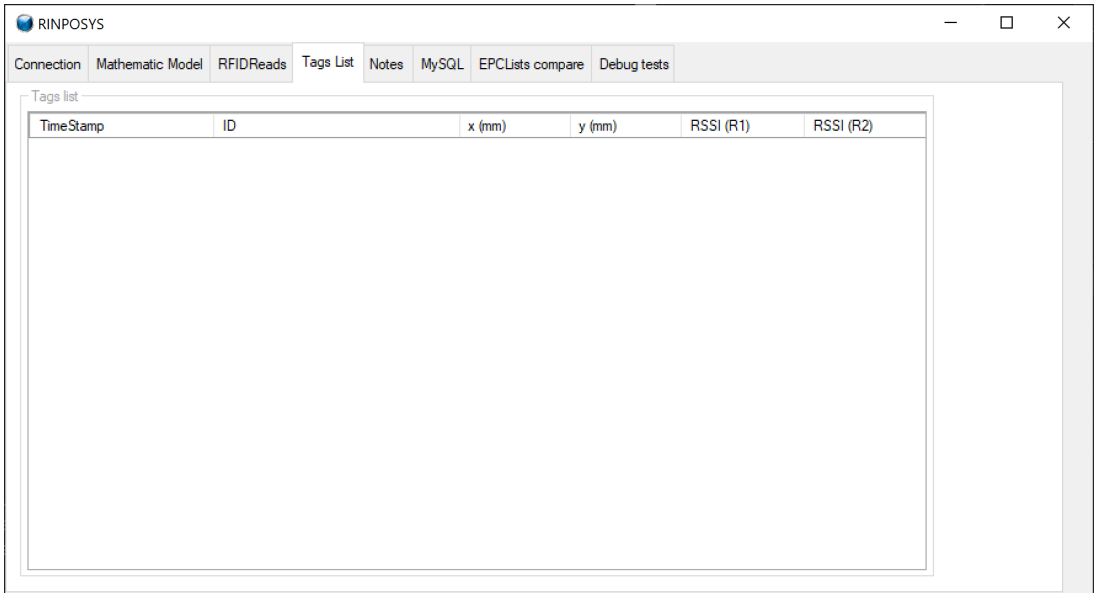

Figure 47 - Computed position tags tab

The fifth tab, “Notes”, has a field to write the notes value to be sent to the database and OPC-UA server when the position value is computed. It can be set to, for instance, “Warehouse 1” which means all these tags were read in one warehouse if there were multiple warehouses running this solution.

The sixth tab” MySQL”, is where the configuration parameters for the MySQL server are located, as shown on Figure 48. There is also the possibility to have the configuration file on a text file specifying the connection parameters to the server.

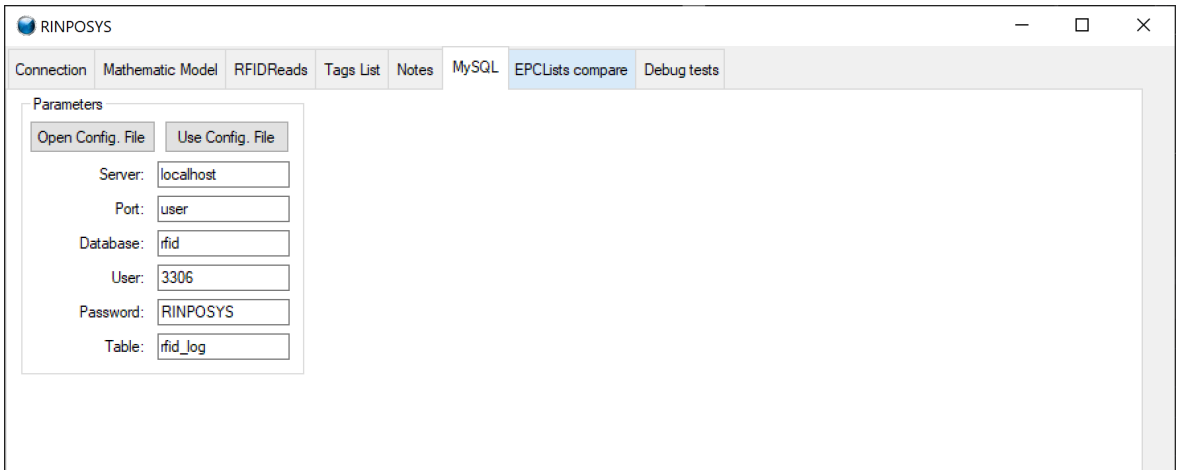

Figure 48 - MySQL configuration tab

On the list of fields, there are:

- Server - in this case "localhost" which is the host where the MySQL server is running.
- Port – the port where the server is responding from, usually is port 3306.
- Database – The database name.
- User/Password – The username and password used for the connection.
- Table – The table used to store the information.

The project's MySQL Server needs to store information gathered by the readers to guarantee some sort of persistence if something is wrong with the OPC-UA server. The server will be running on the PC stick where the readers are connected to make sure no information is ever lost.

To store this information (the values of the tags position), there is a table called "RFID_LOG" that has the following fields:

- Id – the table's primary key and the EPC of the tag to be able to rapidly identify a tags position.
- Datetime – the value of date and time at which the position was calculated.
- Pos_x – The tag's X position value.
- Pos_y – The tag's Y position value.

- Notes – The value passed in the fifth tab “notes”.

These will be the same values passed to the OPC-UA Server.

5.4.2. Tag Position

As seen in the previous chapter if “Use MM” checkbox is selected the software will use the mathematical model to compute the tag positions. Initially there was a problem, that when the software began reading the tags it would start reading tags, but the system was only working on a single thread, which meant that it could only do one task at a time. If another reader was added, or the tag’s location was being computed, there was the need to stop one reader, read from the other and then compute the tags position.

To fix this, a multi-threaded system was implemented, which a diagram can be seen on Figure 49. This means that when the software starts running a thread for each reader is created and an additional thread is started for the mathematical model computation.

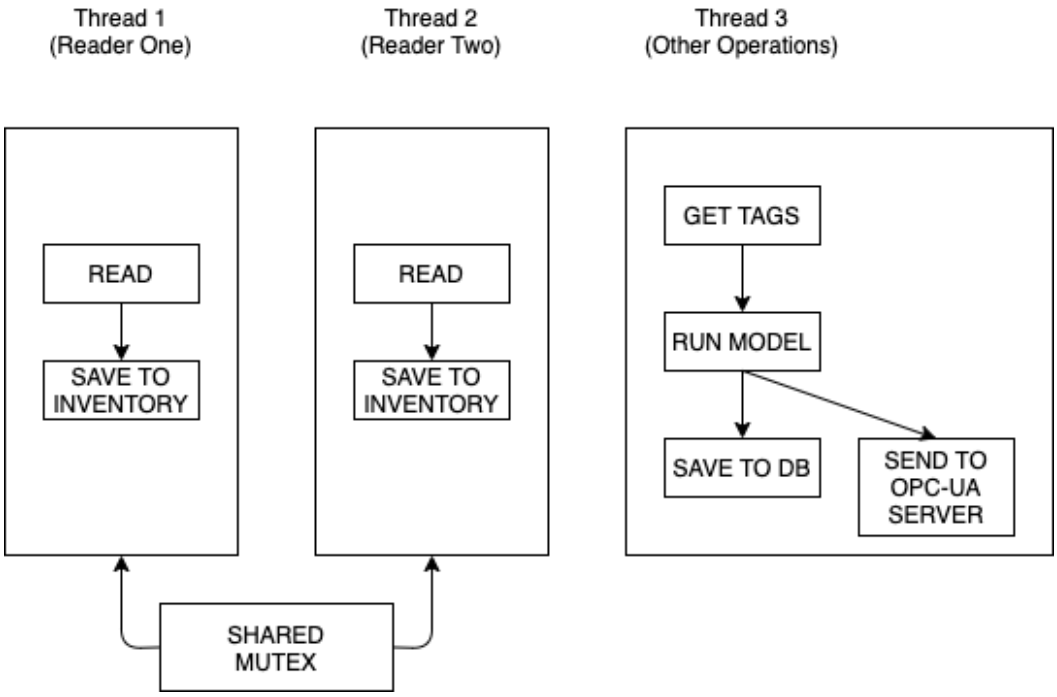

Figure 49 - Read tags algorithm diagram

This image is an example running two readers simultaneously. The two reader's are synchronized by a shared mutex to make sure their inventories (the read tags list) do not overlap. Figure 50 shows the initialization for these threads. As it is shown, several threads can be created for each reader that is connected to the software which will read tags and save them to their respective dictionary. If the mathematical model is enabled another thread is created for the purpose of computing the tags position.

```
ClearUIGridMetaTagsDel();
if (devicesConnected != null)
{
    Devices = devicesConnected;
}

LogUsedDevicesAndMMInformation();
onReads = true;

for (int i = 0; i < Devices.Count; i++)
{
    if (!Devices[i].Connected)
    {
        continue;
    }

    if (Devices[i] is UHFReader)
    {
        UHFReader r = (Devices[i] as UHFReader);
        r.Reset();

        ParameterizedThreadStart processTaskThread = delegate { UseResource(r, readerCounter++, UpdateUIGridMetaTagsDel); };
        Thread deviceThread = new Thread(processTaskThread)
        {
            Name = String.Format("R{0}", i)
        };
        deviceThread.Start();
        threadsUsed.Add(deviceThread);
    }
}

if (useMM)
{
    ParameterizedThreadStart processTaskThreadForAlgorithm = delegate { UseResourceForAlgorithm(Devices, RunAlgorithm); };

    Thread algorithmThread = new Thread(processTaskThreadForAlgorithm)
    {
        Name = "M"
    };

    algorithmThread.Start();
    threadsUsed.Add(algorithmThread);
}
```

Figure 50 - Threads Initialization

The function "*RunAlgorithm*", shown on Figure 51, is the function responsible for getting the tags from the readers dictionaries, compute their position, save them on the database and send them to the OPC-UA server. The function that parses the readers dictionaries basically compares every and each dictionary from each reader and it will check if a tag was already found in at least two dictionaries. If a tag is found on two dictionaries it will be removed from the current dictionaries and they will be saved in a temporary list for further transformation and location estimation. When a tag is saved in this temporary list it

will be checked if the tag does not yet exist in the list and if the tag already exists in this list the current timestamp for last read value is updated.

```
private void RunAlgorithm()
{

    List<TagRead> tagsReads = LocalizationAlgorithm.GetTagsReads(Devices, RemoveFromUIGridMetaTagsDel);
    List<Tag> tags = MathematicalModel.RunMathematicalModel(tagsReads);

    if (tags.Count == 0)
    {
        WriteLog("Mathematical Model applied: 0 tags obtained", 0);

        return;
    }

    tags = LocalizationAlgorithm.TagsPosteriorTreatment(tags, null, 0, false, false);
    UpdateUIGridTagsDel(tags);

    InsertTagsOnDB(tags);
    OPcWriter oPcWriter = new OPcWriter();
    foreach (Tag t in tags)
    {
        oPcWriter.write(t.ID.ToString(), t.TimeStampDateTime.ToString(), t.PositionX.ToString(),
            t.PositionY.ToString(),notes.ToString(), t.R1Power, t.R2Power);
    }
}
```

Figure 51 - Run Algorithm Function

More subjectively, as the crane moves the readers will begin to read tags with an RSSI value, but as the crane continues to move closer to the tag RSSI value will increase. This means that a tag is actively being updated in the dictionaries. When the tag is passed to the temporary list to compute the value the model could not compute the tag's location so this tag will stay in the list waiting for a new value for the RSSI and its timestamp will be updated.

The function responsible for computing the tags' location is shown in Figure 52. As shown in the image it will check if the tag was already read by two antennas, if this is the case, it will then find all of those reads to compute the tag's location.

```
/// Ignore TagRead if it wasn't obtained by at least 2 UHFreaders
if (tagRead.AntennasReads.Count < 2) return null;

/// Used to get the sum of all x coordinates values
double xx = 0;
/// Used to get the sum of all y coordinates values
double yy = 0;
/// Number of (x, y) positions found
int counter = 0;

/// Obtains every combination of AntennaReads of different UHFReaders and calculates x and y positions for the new Tag
for (int i = 0; i < tagRead.AntennasReads.Count - 1; i++)
{
    for (int j = i + 1; j < tagRead.AntennasReads.Count; j++)
    {
        /// First AntennaRead of combination of AntennaReads
        AntennaRead ant1 = tagRead.AntennasReads[i];
        /// Second AntennaRead of combination of AntennaReads
        AntennaRead ant2 = tagRead.AntennasReads[j];
        /// Obtain x by calculating its value with both AntennaReads
        double x = CalculateTagXCoordinate(ant1, ant2);
        /// Ignore AntennaReads combination if x doesn't have a valid value
        if (double.IsNaN(x)) continue;
        /// Obtain y by calculating its value with AntennaRead and obtained x coordinate
        double y = CalculateTagYCoordinate(ant2, x);
        /// Ignore AntennaReads combination if y doesn't have a valid value
        if (double.IsNaN(y)) continue;
        xx += x;
        yy += y;
        counter++;
    }
}
/// If no position was obtained, discard
if (counter == 0) return null;
/// If position value is not valid, discard
if (double.IsInfinity(yy)) return null;
/// Get average values of all x and y, obtained during the iteration of every combination of AntennaReads
double xAvg = xx / counter;
double yAvg = yy / counter;
/// Convert x and y values to a decimal with only 2 decimal digits
decimal xRounded = (decimal)Math.Round(xAvg * 100) / 100;
decimal yRounded = (decimal)Math.Round(yAvg * 100) / 100;
/// Create new Tag
Tag tag = new Tag(tagRead.ID, xRounded, yRounded, tagRead.AntennasReads);
return tag;
```

Figure 52 - Compute Tags Location Function

For computing the X coordinate it will use the following function and its implementation is shown on Figure 53.

$$x = \frac{z_1(z_1 - 2\alpha) + z_2(2\alpha - z_2) + \gamma^2 x^2}{2\gamma^2 x_0}$$

Equation 6 - Compute X coordinate

```
private static double CalculateTagXCoordinate(AntennaRead ant1, AntennaRead ant2)
{
    /**
     * Tag X Coordinate
     *
     * ( ( RSSI_1 * ( RSSI_1 - ( 2 * RSSI_MAX ) + RSSI_2 * ( 2 * RSSI_MAX - RSSI_2 ) )
     *      /
     *      ( 2 * X_AMP^2 ) * ( ANT_2_X_COORD - ANT_1_X_COORD ) )
     * +
     * ( ( ANT_1_COORD + ANT_2_COORD ) / 2 )
     */

    double var1 = ant1.RSSI * (ant1.RSSI - 2 * MathematicModelParameters.RssiMax)
        + ant2.RSSI * (2 * MathematicModelParameters.RssiMax - ant2.RSSI);

    double var2 = 2 * Math.Pow(MathematicModelParameters.XAmp, 2) * (double)(ant2.Antenna.Position.X - ant1.Antenna.Position.X);

    double var3 = var1 / var2;

    double tagXCoordinate = var3 + ((double)(ant1.Antenna.Position.X + ant2.Antenna.Position.X) / 2);

    return tagXCoordinate;
}
```

Figure 53 - Compute X Coordinate

For the calculation of the Y coordinate the following function is used and its implementation is shown on Figure 54.

$$y = \frac{L}{\alpha - \beta}\sqrt{(\alpha - z_2)^2 - \gamma^2(x - x_0)^2}$$

Equation 7 - Compute Y coordinate

```
private static double CalculateTagYCoordinate(AntennaRead ant2, double tagXCoordinate)
{
    /**
     * Tag Y Coordinate
     *
     * RAIZQ(
     *  ( RSSI_MAX - RSSI_2 )^2 - ( X_AMP^2 * ( TAG_X_COORDINATE - ANT_2_X_COORD )^2 )
     * )
     * *
     * L_MAX / ( RSSI_MAX - RSSI_MIN )
     */

    double tagYCoordinate = Math.Sqrt(
        Math.Pow(MathematicModelParameters.RssiMax - ant2.RSSI, 2)
        - (Math.Pow(MathematicModelParameters.XAmp, 2) * Math.Pow(tagXCoordinate - (double)ant2.Antenna.Position.X, 2))
    ) * MathematicModelParameters.LMax / (MathematicModelParameters.RssiMax - MathematicModelParameters.RssiMin);

    return tagYCoordinate;
}
```

Figure 54 - Compute Y Coordinate

Like previously said if the algorithm was not able to compute the tags' location it will stay on the list waiting for a new RSSI value for a new calculation. If the location is successfully computed the algorithm will then save the tags on the database and, after that, send the tags the OPC-UA server.

Figure 55, shows the function that saves the tags location information in the database. This is used for redundancy purposes. This means that if, for some reason, the OPC-UA server is down the computed the tags will be saved to the database, and if the database is down the computed tags will be saved to a local file, guarantying that no information is lost. The values that are saved to the database or local file, are the tag id, which is the unique tag identifier, the timestamp representing the date and time that this tag's location was computed, followed by the x and y coordinates of the tag and finally additional notes.

```
public void InsertTags(List<Tag> tags, string notes)
{
    /// Initialize statement
    string Command = "INSERT INTO " + Table + " (id, datetime, pos_x, pos_y, notes) VALUES (@id, @timestamp, @pos_x, @pos_y, @notes);";
    /// Use connection to DB, and when finished, dispose the MySqlConnection object
    using (MySqlConnection mConnection = new MySqlConnection(ConnStr))
    {
        /// Do connection to DB
        mConnection.Open();
        /// Use transaction, and when finished, dispose the MySqlTransaction object
        using (MySqlTransaction trans = mConnection.BeginTransaction())
        {
            /// Use the MySQL command, with the connection and transaction. When finished, dispose the MySqlCommand
            using (MySqlCommand myCmd = new MySqlCommand(Command, mConnection, trans))
            {
                myCmd.CommandType = CommandType.Text;
                foreach (Tag t in tags)
                {
                    myCmd.Parameters.Clear();
                    /// Apply the tag values into the parameters of the statement initialized on the string vareable "Command"
                    myCmd.Parameters.AddWithValue("@id", t.ID);
                    myCmd.Parameters.AddWithValue("@timestamp", t.TimeStampDateTime);
                    myCmd.Parameters.AddWithValue("@pos_x", t.PositionX);
                    myCmd.Parameters.AddWithValue("@pos_y", t.PositionY);
                    myCmd.Parameters.AddWithValue("@notes", notes);
                    /// Execute statement
                    myCmd.ExecuteNonQuery();
                }
                /// Commit changes
                trans.Commit();
            }
        }
        /// Stop connection to DB
        mConnection.Close();
    }
}
```

Figure 55 - Insert Tags on the database

Writing on the OPC-UA server is relatively easy. As shown on Figure 51, the function "*opc.write*" is called and the values from the tag are sent to this function. The beginning of this function is call of the connection, as shown on Figure 56, when the software connects to the server and, after that, the values which are needed to send to the server need to be parsed from string to DataValue.

```
connect();

m_TypeItem1 = BuiltInType.String;
DataValue val1 = new DataValue();
val1.Value = TypeUtils.Cast(id, m_TypeItem1);
DataValue val2 = new DataValue();
val2.Value = TypeUtils.Cast(datetime, m_TypeItem1);
DataValue val3 = new DataValue();
val3.Value = TypeUtils.Cast(pos_x, m_TypeItem1);
DataValue val4 = new DataValue();
val4.Value = TypeUtils.Cast(pos_y, m_TypeItem1);
DataValue val5 = new DataValue();
val5.Value = TypeUtils.Cast("implementar", m_TypeItem1);
DataValue val6 = new DataValue();
val6.Value = TypeUtils.Cast(notes, m_TypeItem1);
DataValue val7 = new DataValue();
val7.Value = TypeUtils.Cast(r1Power, m_TypeItem1);
DataValue val8 = new DataValue();
val8.Value = TypeUtils.Cast(r2Power, m_TypeItem1);
```

Figure 56 - OPC-UA write parse

When the data is parsed to DataValue the information is now ready to be sent to the OPC-UA Server. For this, and as shown on Figure 57, the NodeId needs to be specified which is where the data is going to be saved. After that a new node is specified all that values that are going to be written are created, and finally the session created ensures that everything is successfully written to the server.

```
if (m_session == null) return null;

// Set up the browse filters.
BrowseContext context = new BrowseContext();
context.BrowseDirection = BrowseDirection.Forward;
context.ReferenceTypeId = ReferenceTypeIds.HierarchicalReferences;
context.IncludeSubtypes = true;
context.MaxReferencesToReturn = 0;

m_continueAutomatically = context.MaxReferencesToReturn <= 0;

// parse the node id.
NodeId nodeId = NodeId.Parse("ns=3;s=Posicionamento");
// this is a blocking call so show the wait cursor.
byte[] continuationPoint = null;
// browse the references (setting a 10 second timeout).
List<ReferenceDescription> references = m_session.Browse(
    nodeId,
    context,
    new RequestSettings() { OperationTimeout = 10000 },
    out continuationPoint);
/// [Browse 2]
bool isWritten = false;
// add references to control.
foreach (ReferenceDescription reference in references)
{
    ReadValueIdCollection nodesToRead = new ReadValueIdCollection();
    string TagName = reference.BrowseName.Name;
    nodesToRead.Add(new ReadValueId() { NodeId = NodeId.Parse("ns=3;s=" + TagName + ".id"), AttributeId = 13 });
    List<DataValue> results = m_session.Read(nodesToRead);
    string idReference = results[0].WrappedValue.ToString();

    if (id == idReference)
    {
        List<WriteValue> nodesToWrite = new List<WriteValue>();
        nodesToWrite.Add(new WriteValue() { NodeId = NodeId.Parse("ns=3;s=" + TagName + ".id"), AttributeId = 13, Value = val1, });
        nodesToWrite.Add(new WriteValue() { NodeId = NodeId.Parse("ns=3;s=" + TagName + ".datetime"), AttributeId = 13, Value = val2 });
        nodesToWrite.Add(new WriteValue() { NodeId = NodeId.Parse("ns=3;s=" + TagName + ".pos_x"), AttributeId = 13, Value = val3 });
        nodesToWrite.Add(new WriteValue() { NodeId = NodeId.Parse("ns=3;s=" + TagName + ".pos_y"), AttributeId = 13, Value = val4 });
        nodesToWrite.Add(new WriteValue() { NodeId = NodeId.Parse("ns=3;s=" + TagName + ".error"), AttributeId = 13, Value = val5 });
        nodesToWrite.Add(new WriteValue() { NodeId = NodeId.Parse("ns=3;s=" + TagName + ".notes"), AttributeId = 13, Value = val6 });
        nodesToWrite.Add(new WriteValue() { NodeId = NodeId.Parse("ns=3;s=" + TagName + ".r1power"), AttributeId = 13, Value = val7 });
        nodesToWrite.Add(new WriteValue() { NodeId = NodeId.Parse("ns=3;s=" + TagName + ".r2power"), AttributeId = 13, Value = val8 });
        List<StatusCode> resultsWrite = m_session.Write(nodesToWrite);
        isWritten = true;
    }

}
```

Figure 57 - OPC-UA Write

5.4.3. OPC-UA Server Implementation

The OPC-UA server, as shown in Figure 58, is responsible for receiving the data from any device, such as PLC, CNC, any control device, etc. When the data is received on the server it automatically becomes available to any kind of client. The OPC-UA clients can subscribe the information that is being broadcast by the OPC-Server.

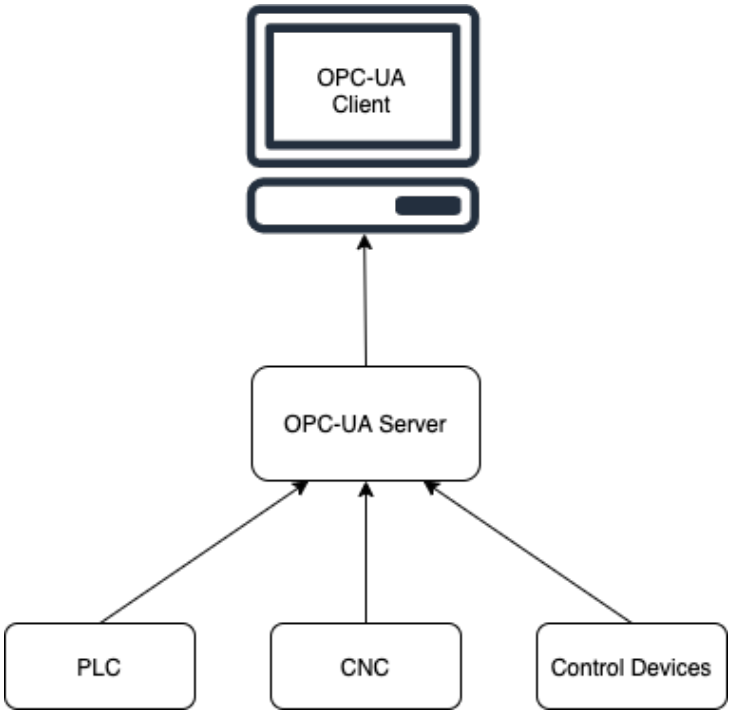

Figure 58 - OPC-UA architecture

The main file responsible for managing an OPC-UA Server is usually called "OpcServerNodeManager.cs". This file is the server's main file. It is responsible for loading our address space, create all the variables that would be available to use, read, and write values from it, and managing all the other abstract specifications.

5.4.4. OPC-UA Model

Usually the models are authority certified, which means that a specification can be used and automatically load into our address space. This specification is defined in XML (Extensible Markup Language) which is a markup language that defines a set of rules in a format that can be easily read by humans but also easily read by machines.

As shown in Figure 59, the variables for the address space are specified. When the software loads, the client specifies the node which is the set of objects and related information available to all the clients, which in this case is node “1” and then the software will load all the variables defined on our XML file. In this case, the same variables set on MySQL model are specified.

```
<UAVariable DataType="String" ParentNodeId="ns=1;i=1001" NodeId="ns=1;i=6001" BrowseName="1:id" UserAccessLevel="3" AccessLevel="3">
  <DisplayName>id</DisplayName>
  <References>
    <Reference ReferenceType="HasTypeDefinition">i=2365</Reference>
    <Reference ReferenceType="HasModellingRule">i=78</Reference>
    <Reference ReferenceType="HasComponent" IsForward="false">ns=1;i=1001</Reference>
  </References>
</UAVariable>
<UAVariable DataType="String" ParentNodeId="ns=1;i=1001" NodeId="ns=1;i=6002" BrowseName="1:datetime" UserAccessLevel="3" AccessLevel="3">
  <DisplayName>datetime</DisplayName>
  <References>
    <Reference ReferenceType="HasTypeDefinition">i=2365</Reference>
    <Reference ReferenceType="HasModellingRule">i=78</Reference>
    <Reference ReferenceType="HasComponent" IsForward="false">ns=1;i=1001</Reference>
  </References>
</UAVariable>
<UAVariable DataType="String" ParentNodeId="ns=1;i=1001" NodeId="ns=1;i=6003" BrowseName="1:pos_x" UserAccessLevel="3" AccessLevel="3">
  <DisplayName>pos_x</DisplayName>
  <References>
    <Reference ReferenceType="HasTypeDefinition">i=2365</Reference>
    <Reference ReferenceType="HasModellingRule">i=78</Reference>
    <Reference ReferenceType="HasComponent" IsForward="false">ns=1;i=1001</Reference>
  </References>
</UAVariable>
<UAVariable DataType="String" ParentNodeId="ns=1;i=1001" NodeId="ns=1;i=6004" BrowseName="1:pos_y" UserAccessLevel="3" AccessLevel="3">
  <DisplayName>pos_y</DisplayName>
  <References>
    <Reference ReferenceType="HasTypeDefinition">i=2365</Reference>
    <Reference ReferenceType="HasModellingRule">i=78</Reference>
    <Reference ReferenceType="HasComponent" IsForward="false">ns=1;i=1001</Reference>
  </References>
</UAVariable>
<UAVariable DataType="String" ParentNodeId="ns=1;i=1001" NodeId="ns=1;i=6005" BrowseName="1:error" UserAccessLevel="3" AccessLevel="3">
  <DisplayName>error</DisplayName>
  <References>
    <Reference ReferenceType="HasTypeDefinition">i=2365</Reference>
    <Reference ReferenceType="HasModellingRule">i=78</Reference>
    <Reference ReferenceType="HasComponent" IsForward="false">ns=1;i=1001</Reference>
  </References>
</UAVariable>
```

Figure 59 - Address space configuration xml file

This file was generated by graphical user interface (GUI) on a software called UaModeler that is made by a company called Unified Automation [55]. A screenshot of this

software can be seen on Figure 60. Basically, the namespace is set and then all the variables that are needed can be added. A folder called “Posicionamento” was created which is the folder where all the computed tags are found and then an object inside of this folder called “tag” is created and inside this tag object all the variables are added. When done, the project can be exported and a file like the one on Figure 59 is generated.

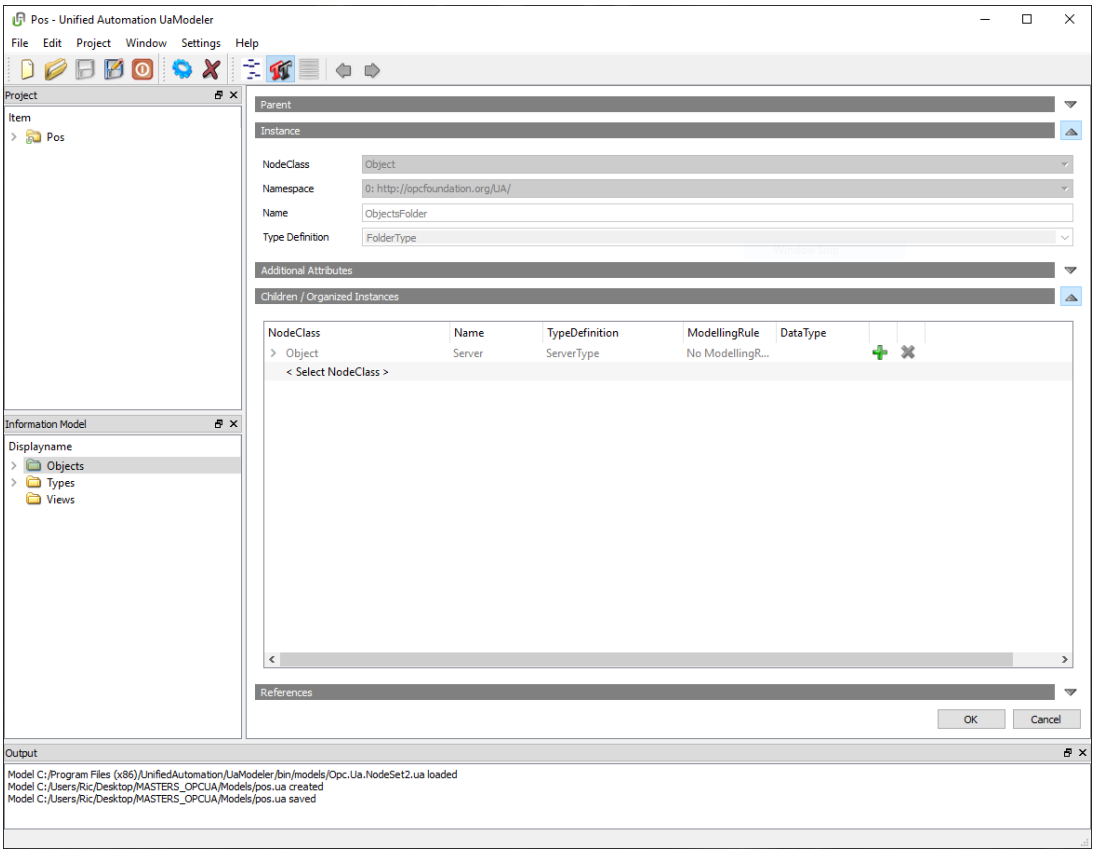

Figure 60 - UaModeler interface

5.4.5. OPC-UA Implementation

The implementation of this server was the most difficult part of this project because there is not much information on the internet about this subject or solutions to the usual errors encountered. For an easier implementation, the SDK [56] provided by "Unified Automation" was used as it is almost plug and play and much easier to understand than the one provided by the OPC Foundation. To decrease the projects complexity of using another programming language, the server was implemented using .NET Framework because the software the readers use was already implemented in .NET. This SDK comes with a pre-installed version of an OPC-UA server and client.

The server start-up function is shown on Figure 61. It is where the tags are initialized, the namespace and the address space are loaded, and it is where the main folder "Posicionamento" (where the server can then write) is created.

```
1 reference
public override void Startup()
{
    try
    {
        Console.WriteLine("Insira o numero de tags:");
        string ntags = Console.ReadLine();
        Console.WriteLine("Starting OpcServerNodeManager.");

        base.Startup();

        // save the namespaces used by this node manager.

        TypeNamespaceIndex = AddNamespaceUri(IPL.BA.Namespaces.BA);
        InstanceNamespaceIndex = AddNamespaceUri("http://ipleiria.pt/TAGS/");

        // load the model.
        Console.WriteLine("Loading the Controller Model.");
        ImportUaNodeset(Assembly.GetEntryAssembly(), "buildingautomation.xml");

        m_system.Initialize();


        // Create a Folder for Controllers
        CreateObjectSettings settings = new CreateObjectSettings()
        {
            ParentNodeId = ObjectIds.ObjectsFolder,
            ReferenceTypeId = ReferenceTypeIds.Organizes,
            RequestedNodeId = new NodeId("Posicionamento", InstanceNamespaceIndex),
            BrowseName = new QualifiedName("Posicionamento", InstanceNamespaceIndex),
            TypeDefinitionId = ObjectTypeIds.FolderType
        };
        CreateObject(Server.DefaultRequestContext, settings);


        createNodes(Convert.ToInt32(ntags));
        Console.WriteLine("Criadas " + ntags + " tags");
    }
    catch (Exception e)
    {
        Console.WriteLine("Failed to start OpcServerNodeManager. " + e.Message);
    }
}
```

Figure 61 - OPC-UA Server Initialization

All the variables used on the address space need to be initialized. To surpass this problem, before the server starts running, the user is asked to insert how many tags are available for reading. This prompt is shown in the initial lines of the start-up function (Figure 61) and the creation of the tags are shown on Figure 62. This way the server knows how many tags it must read, and it can initialize the server with this number of tags.

```
private void createNodes(int numberOfTags)
{

    // Create controllers from configuration
    foreach (BlockConfiguration block in m_system.GetBlocks())
    {
        // set type definition NodeId


        NodeId typeDefinitionId = new NodeId(IPL.BA.ObjectTypes.Tag, TypeNamespaceIndex);
        //++ntags;
        for (int i = 1; i <= numberOfTags; i++)
        {
            // create object.
            CreateObjectSettings settings = new CreateObjectSettings()
            {
                ParentNodeId = new NodeId("Posicionamento", InstanceNamespaceIndex),
                ReferenceTypeId = ReferenceTypeIds.Organizes,
                RequestedNodeId = new NodeId(block.Name + i.ToString("000"), InstanceNamespaceIndex),
                BrowseName = new QualifiedName(block.Name + i.ToString("000"), TypeNamespaceIndex),
                TypeDefinitionId = typeDefinitionId
            };
            CreateObject(Server.DefaultRequestContext, settings);
        }
    }

}
```

Figure 62 - OPC-UA Create Nodes

After having the value of all the tags, that the server is going to read, the XML file containing the address space objects is loaded and the server creates as many tags as previously prompted. After this the server initializes and it is ready to go. As can be seen, for the creation of an OPC-UA server there is only the need the define the namespace, address space and the main NodeId, after that the clients can connect and write data.

The reader software needs to be able to connect to the server to write the tags value, for this to happen the SDK's OPC-UA client DLL (dynamic link library) needs to be imported and the server has all the methods required to connect and write to the OPC-UA server. It is also needed to import the XML file containing the address space configuration to be able to write the data in a way that the server is expecting. The server's default listen URL is "opc.tcp.//localhost:4841". Now, when the tags' location is computed, the server writes the tags location for the clients to read.

5.4.6. OPC-UA Client

The client for an OPC-UA can be any client. For this project a client was not created to view the data because it was only needed to create the server. So, to watch the data being changed and added it was used Unified Automation's "UaExpert" [57] which is an OPC-UA client made by this company.

When UaExpert starts, there is the possibility to connect to the server by inserting its URL and then it shows all the address space on the lower left side of Figure 63. There, it exists the "Posicionamento" folder which contains all the tags that created initially.

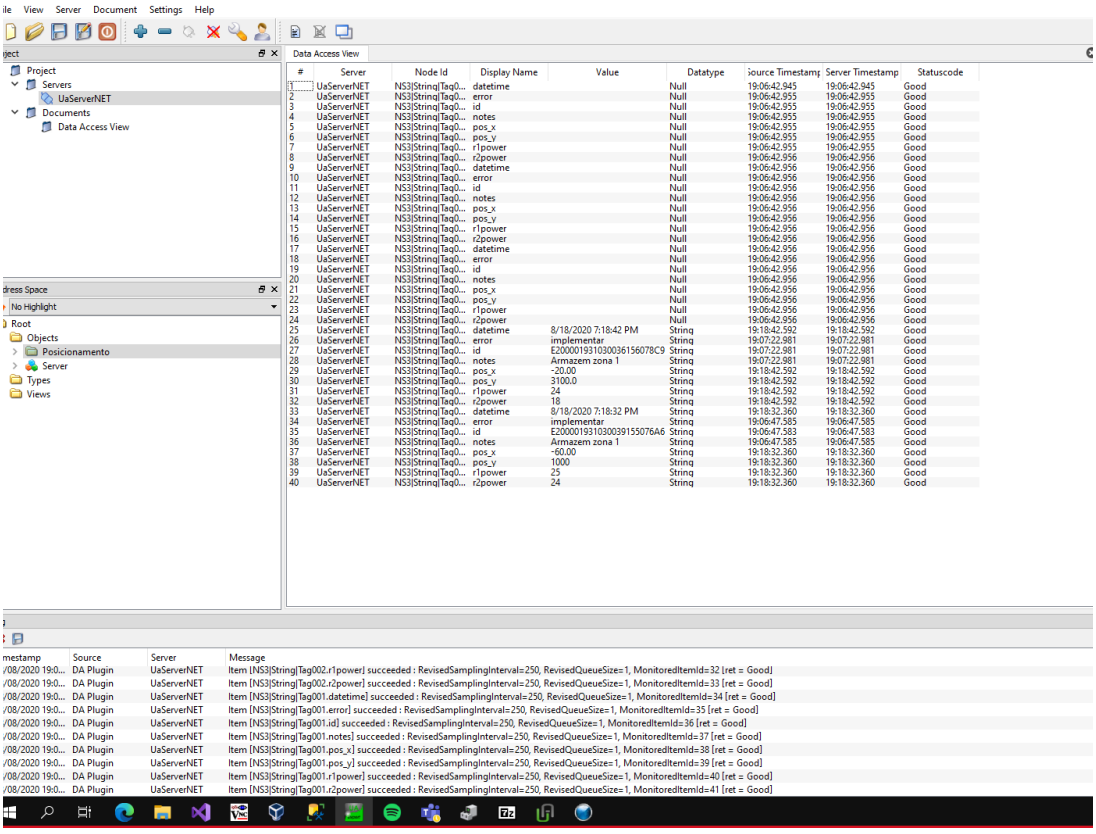

Figure 63 - UaExpert client with tags position

To see the tag values, there is the possibility to drag the tags to centre of the application. These tags are now subscribed. Every time their values change this interface is updated showing the real time value. The position of the tags is shown and their identification which is the information the client is most concerned about.

5.4.7. Further work (Power Variation)

There was also the need to aid my colleagues Ricardo Barbara and João Machaqueiro in their project. Their project was adding the variation of power in the antenna to better try and read the RFID tags. For this purpose, to the code of the reader, five new variables were added. These variables were "AntennaHeight", "AntennaAngle", "AntennaHypotenuse", "MaxPowerDbm" and "MinPowerDbm". With these values the power variation in the antenna could be automated, because if knowing the angle of the antenna in the bridge (a), and the antenna's height from the floor (h), it was possible to estimate the distance in LOS, by computing the value of the hypotenuse X, being (as illustrated in Figure 64):

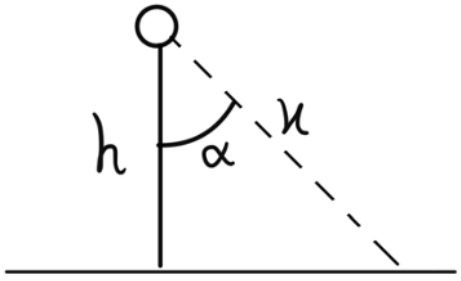

Figure 64 - Hypotenuse Variation

After having these values computed the code was changed to support the new variable value for the power. This meant that at every iteration of the algorithm, the reader's power would be initialized to the new value from top to bottom. Two new parameters on the OPC-UA server were also added to be able to show the values of the power when the tag's location was computed, as well as adding two new fields on the database to store these two new values. This is further explained on their project report [58].

6. Conclusion

To conclude this project, there is the need to go back and think about the beginning of it. My area of expertise is mostly software, as it is where most of my studies are. I choose this project because it had a hardware component which was more of a challenge for me.

Over the course of these two years, we faced several obstacles, but the most difficult thing for me was the OPC-UA server implementation. This technology has little or no documentation on the internet, and the documentation found online is only made by the OPC Foundation with examples on how the technology works. To make it work it was mostly trial and error. We first had the OPC-UA server running and sending the data to the server using JSON (JavaScript Object Notation) objects. After that, we eventually made it work with our model.

The overall objective of the project was having the tag positions with an acceptable error for a big warehouse, and in the testing phase we got an average error of 0.5 meters, which we find it to be a success.

For future work, we would have to test this project in a real warehouse and really consider all the external variables to really know the accuracy of our model. We never got to test it because when we had the chance, we still did not have a working prototype and eventually a global pandemic appeared that could not let us test the project. Another thing we would add to project was a way of knowing were the bridge was in correlation to the warehouse to really know were tags are, because right now we only know the position of the tags to the bridge. So, we needed a way of always knowing the whereabouts of the bridge to really know the position of the tags in the warehouse. For this purpose, we would probably use an array of lasers that could measure the distance on the X and Y to the warehouse walls.

References

[1] O. Dictionary, “What is the Industry?,” 2020. [Online]. Available: https://google.com/search?q=industry.

[2] K. Schwab, 14 1 2016. [Online]. Available: https://www.weforum.org/agenda/2016/01/the-fourth-industrial-revolution-what-it-means-and-how-to-respond/. [Accessed 2020].

[3] “CVision,” [Online]. Available: http://www.cvisiontech.com/library/document-automation/data-entry/automatic-data-capture.html. [Accessed 2020].

[4] “DENSO,” [Online]. Available: https://www.denso-wave.com/en/adcd/fundamental/barcode/barcode/index.html. [Accessed 2020].

[5] “Compute a Label,” [Online]. Available: http://www.computalabel.com/aboutupc.htm. [Accessed 2020].

[6] Matthews Intelligent Identification, “Breaking it down - EAN-13 Barcode,” [Online]. Available: https://www.matthews.com.au/ean-13-barcode. [Accessed 10 8 2020].

[7] “K&J Magnetics,” [Online]. Available: https://www.kjmagnetics.com/blog.asp?p=magnetic-stripes. [Accessed 2020].

[8] VariusCard, “DUAL INTERFACE CHIP CARDS,” [Online]. Available: https://www.variuscard.com/plastic-cards/smartcard-rfid-nfc-mifare/dual-interface-chip-card/?lang=en. [Accessed 30 7 2020].

[9] D. Bhattacharyya, R. Ranjan, P. Das, T.-h. Kim and S. Bandyopahdhyay, “Biometric Authentication Techniques and its Future Possibilities,” 2009 Second International Conference on Computer and Electrical Engineering, India and Korea, 2009.

[10] DreamLab Technologies, “Attacking Biometric Systems with 3D Printing,” 31 8 2020. [Online]. Available: https://dreamlab.net/en/blog/post/attacking-biometric-systems-with-3d-printing/. [Accessed 2 9 2020].

[11] “ElectronicsNotes,” [Online]. Available: https://www.electronics-notes.com/articles/connectivity/rfid-radio-frequency-identification/development-history.php#:~:text=While%20RFID%20history%20can%20be,Watt%20when%20he%20invented%20radar.. [Accessed 2020].

[12] “Arizona's University,” [Online]. Available: http://www.u.arizona.edu/~obaca/rfid/history.html. [Accessed 2020].

[13] S. Hofmayr, “Analysis and comparison of the potential of RFID-technology in European and U.S. retail supply chains”.

[14] M. P. Bach, J. Zoroja and M. Loupis, “RFID usage in European enterprises and its relation to competitiveness: Cluster analysis approach,” International Journal of Engineering Business Management Volume 8: 1–11, 2016.

[15] “Ecorfid,” [Online]. Available: https://ecorfid.electronicacerler.com/index.php/en/rfid-tag-electronic-product. [Accessed 2020].

[16] “Lowry Solutions,” [Online]. Available: https://lowrysolutions.com/blog/a-guide-to-understanding-uhf-passive-rfid-antennas/#:~:text=RFID%20antennas%20perform%20two%20very,data%20from%20multiple%20tags%20simultaneously.. [Accessed 2020].

[17] “AnalogicTips,” [Online]. Available: https://www.analogictips.com/rfid-tag-and-reader-antennas/. [Accessed 2020].

[18] Y. B. Bai, S. Wu, H. Wu and K. Zhang, “Overview of RFID-Based Indoor Positioning Technology,” RMIT University, Australia, 2012.

[19] “The Free Dictionary,” [Online]. Available: https://encyclopedia2.thefreedictionary.com/Electronic+Product+Code. [Accessed 2020].

[20] “cxjrfidfactory,” 15 8 2016. [Online]. Available: https://www.cxjrfidfactory.com/what-is-epc-rfid-stardard/. [Accessed 2020].

[21] “Tutorial's Point,” [Online]. Available: https://www.tutorialspoint.com/electronic-product-code-epc#:~:text=Electronic%20Product%20Code%20(EPC)%20is,and%20people%2C%20and%20track%20them.. [Accessed 2020].

[22] P. Darcy, P. Pupunwiwa and B. Stantic, “The Challenges and Issues Facing the Deployment of RFID Technology,” Institute of Integrated and Intelligent Systems, Griffith University, Australia.

[23] M. Bouet and A. L. d. Santos, “RFID Tags: Positioning Principles and Localization Techniques,” France, Brazil .

[24] T. Tugcu, G. Gür, F. Alagoz and O. Türkyilmaz, “Environment-Aware Location Estimation in Cellular Networks,” 2008. [Online]. Available: https://www.researchgate.net/figure/Location-estimation-via-circle-intersection-using-RSS-values-from-serving-and-neighbor_fig5_220057312.

[25] K. Chawla, C. McFarland, G. Robins and C. Shope, “Real-Time RFID Localization Using RSS,” University of Virginia, Charlottesville, Virginia, USA.

[26] F. Gagnon, S. Yousefi, G. Kaddoum and A. Tahat, “A Look at the Recent Wireless Positioning Techniques With a Focus on Algorithms for Moving Receivers,” 2016. [Online]. Available: https://www.researchgate.net/figure/Location-by-ranges-such-as-RSS-or-TOA-measurements_fig1_309846689.

[27] U. H. KHAN, H. RASHEED, B. ASLAM, A. FATIMA, L. S. Y. AMIN and H. TENHUNEN, “Localization of Compact Circularly Polarized RFID TagUsing ToA Technique,” University of Turku, Turku-20520, Finland, 2017.

[28] Y. L. Zhongjin Ai, “Research on the TDOA Measurement of Active RFID Real Time Location System,” Nanchang University, Nanchang, China .

[29] M. Zaidi, R. Tourki and R. Ouni, “A new geometric approach to mobile position in wireless LAN reducing complex computations,” 2010. [Online]. Available:

https://www.researchgate.net/figure/Position-determination-techniques-a-TOA-b-TDOA-c-AOA_fig1_241168823.

[30] C. Zhanga and Y. Lib, “RFID Localization System Based on K-Nearest Neighbor Algorithm and Extreme Learning Machine Algorithm with Virtual Reference Tags,” Tianjin Polytechnic University, Tianjin,China.

[31] A. Bekkali, H. Sanson and M. Matsumoto, “RFID Indoor Positioning Based on Probabilistic RFID Map and Kalman Filtering,” 2007. [Online]. Available: https://www.semanticscholar.org/paper/RFID-Indoor-Positioning-Based-on-Probabilistic-RFID-Bekkali-Sanson/410fe145c5ea56a96938dfb5755ba01eae3a8dda.

[32] E. Vahedi, V. Shah-Mansouri, V. W. Wong and I. F. Blake, “A Probabilistic Approach for DetectingBlocking Attack in RFID Systems,” Communications (ICC), 2010 IEEE International Conference, Vancouver, Canada, 2010.

[33] W. Wang, C. Asci and S. Sonkusale, “Using RFID tags for home security monitoring,” 2020. [Online]. Available: https://www.embedded.com/using-rfid-tags-for-home-security-monitoring/.

[34] Zawya, 2020. [Online]. Available: https://www.zawya.com/mena/en/press-releases/story/flydubai_uses_cuttingedge_technology_by_ASD_to_track_and_clean_aircraft_seat_covers-ZAWYA20200909091444/. [Accessed 2020].

[35] A. K. Das, “swarajyamag,” 14 8 2020. [Online]. Available: https://swarajyamag.com/news-brief/indian-railways-installing-rfid-tags-to-track-entire-flefloet-of-wagons-coaches-and-locomotives. [Accessed 2020].

[36] “Lowry Solutions,” [Online]. Available: https://lowrysolutions.com/blog/how-does-an-rfid-asset-tracking-system-work/. [Accessed 2020].

[37] “InfSoft,” [Online]. Available: https://www.infsoft.com/technology/positioning-technologies/rfid. [Accessed 2020].

[38] Schwarz M.H. and Börcsök J., “A Survey on OPC and OPC-UA About the standard, developments and investigations,” in *2013 XXIV International*

Conference on Information, Communication and Automation Technologies (ICAT), Sarajevo, Bosnia and Herzegovina, 2013.

[39] "GitHub OPC-UA," [Online]. Available: https://github.com/OPCFoundation.

[40] "OPC-UA Specifications," [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture. [Accessed 2020].

[41] "OPC-UA Information Model," [Online]. Available: https://documentation.unified-automation.com/uasdkhp/1.0.0/html/_l2_opc_ua_specifications.html.

[42] "OPC-UA Address Space," [Online]. Available: https://documentation.unified-automation.com/uasdkhp/1.0.0/html/_l2_ua_address_space_concepts.html. [Accessed 2020].

[43] "OPC-UA Models," [Online]. Available: https://opcfoundation.org/developer-tools/specifications-opc-ua-information-models/opc-unified-architecture-for-cnc-systems/.

[44] O. Foundation. [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-3-address-space-model/. [Accessed 2021].

[45] O. Palonen, "OBJECT-ORIENTED IMPLEMENTATION OF OPC UA INFORMATION MODELS IN JAVA," aalto university School of Science and Technology, Espoo, 2010.

[46] H. Renjie, L. Feng and P. Dongbo, "Research on OPC-UA Security," College of Computer and Information Science, ChongQing, China.

[47] "Unified Automation News," 2009. [Online]. Available: https://www.unified-automation.com/news/news-details/article/opc-ua-connects-wind-turbines-in-germanys-first-offshore-wind-park.html. [Accessed 2021].

[48] [Online]. Available: http://www.chafon.com/productdetails.aspx?pid=565.

[49] “Chafon,” [Online]. Available: http://www.chafon.com/productdetails.aspx?pid=590. [Accessed 2020].

[50] ORACLE, “MySQL,” 2021. [Online]. Available: https://www.mysql.com/.

[51] “Chafon's SDK,” [Online]. Available: http://www.chafon.com/DownLoadFile.aspx?fid=222. [Accessed 2020].

[52] “Qemu,” [Online]. Available: https://www.qemu.org/. [Accessed 2020].

[53] “Usb to Uart Driver,” [Online]. Available: https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers.

[54] J. Pereira, H. Gomes, S. Mendes, R. Santos, S. Faria and C. Neves, “Long Range RFID Indoor Positioning System with Passive Tags,” Industry4.0Summit, Manchester - UK, Abr. 2019, com publicação num livro intitulado ‘Industry 4.0 – Shaping the Future of the Digital World’ (Taylor & Francis, HB ISBN 978-0-367-42272-1, E-Book 978-0-367-82308-5), 2020.

[55] “Ua Modeler,” [Online]. Available: https://www.unified-automation.com/products/development-tools/uamodeler.html.

[56] “Unified Automation SDK,” [Online]. Available: https://www.unified-automation.com/downloads/opc-ua-development.html. [Accessed 2020].

[57] “Ua Expert,” [Online]. Available: https://www.unified-automation.com/products/development-tools/uaexpert.html. [Accessed 2020].

[58] J. Machaqueiro and R. Bárbara, “Tooling4G - Localização remota de Tags RFID passivas em ambientes Indoor. Módulo 1 – Desenvolvimento de uma aplicação com leituras RFID usando uma ponte rolante,” 2020.

[59] A. S. D. Abin M Sabu, “A Survey on various Optical Character Recognition Techniques,” Proc. IEEE Conference on Emerging Devices and Smart Systems (ICEDSS 2018) , Kannur, India, 2018.

[60] J. Zhou, H. Zhang and L. Mo, “Two-dimension Localization of Passive RFID Tags Using AOA Estimation,” State Key Laboratory of Industrial Control

Technology, Department of Control Science and Engineering, Hangzhou, P.R.China.

[61] "What is OPC-UA," [Online]. Available: https://www.opc-router.com/what-is-opc-ua/. [Accessed 2020].

[62] A. Serra, "UHF GEN2 RFID CHANNELS FOR EUROPE AND THE USA," TimingSense, 1 8 2018. [Online]. Available: https://timingsense.com/en/blog/uhf-gen2-rfid-channels-europe-usa/#diferencias. [Accessed 2 10 2020].

This page was intentionally left blank

Appendices

Appendices I

	-2,5	-2,25	-2	-1,75	-1,5	-1,25	-1	-0,75	-0,5	-0,25	0	0,25	0,5	0,75	1	1,25	1,5	1,75	2	2,25	2,5
0	0	0	0	0	0	0	0	0	192	209	210	206	192	0	0	0	0	0	0	0	0
0,25	0	0	0	0	0	0	0	0	188	205	215	205	198	0	0	0		0	0	0	0
0,5	0	0	0	0	0	0	0	0	192	205	211	201	199	191	0	0	0	0	0	0	0
0,75	0	0	0	0	0	0	0	0	195	205	208	202	195	194	192	0	0	0	0	0	0
1	0	0	0	0	0	0	187	198	203	205	207	200	0	196	193	190	0	0	0	0	0
1,25	0	0	0	0	0	0	189	198	204	205	205	202	188	190	192	189	0	0	0	0	0
1,5	0	0	0	0	0	187	189	194	199	201	204	201	192	0	187	189	188	189	0	0	0
1,75	0	0	0	0	0	0	188	194	200	201	201	198	193	0	0	188	188	190	189	0	0
2	0	0	0	0	189	192	199	202	202	202	200	197	191	0	0	188	187	188	0	0	0
2,25	0	0	0	0	192	195	199	200	199	201	199	198	194	188	0	0	0	0	0	0	0
2,5	0	0	0	0	191	195	198	199	201	202	201	200	198	192	0	0	0	0	0	0	0
2,75	0	0	0	0	192	195	198	201	201	202	202	201	199	192	0	0	0	0	0	0	0
3	0	0	0	0	191	194	199	199	201	202	201	200	198	192	0	0	0	0	0	0	0
3,25	0	0	0	0	191	193	196	199	200	200	200	199	197	193	0	0	0	0	0	0	0
3,5	0	0	0	0	195	192	192	195	198	199	199	197	193	190	0	0	0	0	0	0	0
3,75	0	0	0	0	0	192	192	191	190	194	196	195	194	193	0	0	0	0	0	0	0
4	0	0	0	0	194	191	190	188	188	190	191	187	0	0	0	0	0	0	0	0	0
4,25	0	0	0	0	189	194	194	189	190	192	197	193	190	189	188	0	0	0	0	0	0
4,5	0	0	0	0	192	193	196	196	197	198	194	191	195	191	189	0	0	0	0	0	0
4,75	0	0	0	0	0	189	190	187	190	189	190	190	188	190	0	0	0	0	0	0	0
5	0	0	0	0	189	188	189	191	195	195	194	193	194	190	189	0	0	0	0	0	0

This page was intentionally left blank

Glossary

Transponder	A device that, upon receiving a signal, emits a different signal in response.
Transceiver	A device that can both transmit and receive communications, in particular a combined radio transmitter and receiver.
Coupling	A device for connecting parts of machinery.
Anti-Collision	An algorithm to prevent collisions.
Multi-Path	The propagation phenomenon that results in radio signals reaching the receiving antenna by two or more paths.
Line-of-Sight (LOS)	A straight line along which an observer has unobstructed vision.
Triangulation	The process of determining the location of a point by measuring only angles to it from known points at either end of a fixed baseline.
Shadowing	If two tags are on a straight line, the furthest one can become shadowed by the first one and not being able to communicate with the transceiver.
Fingerprints	Also known as “Anchor Tags” are tags put in pre-determined places to increase the algorithm’s accuracy.
Chroot Environment	Operation that changes the apparent root directory for the current running process and its children.
Mutex	Synchronization mechanism for enforcing limits on access to a resource in an environment where there are many threads of execution
Polling	Actively sampling the status of an external device by a client program as a synchronous activity (always listening).