

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Real-time Location Systems and Internet of Things Sensors

Filipe Manuel Ferreira Cordeiro



Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Miguel Pimenta Monteiro

July 25, 2019

Real-time Location Systems and Internet of Things Sensors

Filipe Manuel Ferreira Cordeiro

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Doctor Jorge Alves da Silva

External Examiner: Doctor Helena Cristina Coutinho Duarte Rodrigues

Supervisor: Doctor António Miguel Pontes Pimenta Monteiro

July 25, 2019

Abstract

Benefits from locating and tracking assets in industry have led many manufacturers to consider implementing Real-time location systems. Real-time location systems are used for locating and tracking, in real time, objects or people in indoor environments. But unlike satellite-based systems used for outdoor locating, RTLSs still rely on several different technologies each having its own advantages and its disadvantages. Because there isn't a standard technology for real-time locating and because each industrial scenario can have different requirements and constraints, the selection of the most suitable system, by a manufacturer for its facilities, can be difficult. The use of radio frequency technologies, like Bluetooth or WiFi, in RTLS creates an opportunity to extend mobile tracking devices as IoT sensors. This dissertation aims to better understanding what are the best technologies for using an RTLS in an industrial environment, what are the current RTLS market solutions and demonstrate an integration of such solution into a Manufacturing Execution System.

Resumo

Os benefícios de localizar e rastrear ativos na indústria levaram muitos fabricantes a considerar a implementação de sistemas de localização em tempo real. Sistemas de localização em tempo real são usados para localizar e rastrear, em tempo real, objetos ou pessoas em espaços interiores. Mas, ao contrário dos sistemas baseados em satélite usados para localização no exterior, os RTLSs ainda se baseiam em várias tecnologias diferentes, cada uma com suas próprias vantagens e desvantagens. Como ainda não existe uma tecnologia padrão para localização em tempo real e porque cada cenário industrial pode ter requisitos e restrições diferentes, a seleção do sistema mais adequado, por um fabricante para suas instalações, pode ser difícil. O uso de tecnologias de radiofrequência, como Bluetooth ou WiFi, em RTLS cria uma oportunidade para transformar dispositivos de rastreamento móveis também em sensores de IoT. Esta dissertação tem como objetivo compreender melhor quais são as melhores tecnologias para utilização de um RTLS em um ambiente industrial, quais são as atuais soluções de mercado RTLS e demonstrar uma integração de tal solução em um Sistema de Execução de Fabricação.

Acknowledgements

To my parents, for the life they provided me, and the education and support they gave me over the years. To my wife, for all the love and patience and motivation. To my newborn daughter Inês, in spite of having made this dissertation much harder, also made it worth much more. To my future children, so that if they ever read this they don't get jealous of their sister. To my brother and the rest of the family, to all my friends, for making my life so good and so full. To my supervisors, Professor Pimenta Monteiro e João Santos, for all the help they provided during this dissertation.

I sincerely thank you all,

Filipe Cordeiro

*“Education is not the learning of facts,
but the training of the mind to think”*

Albert Einstein

Contents

1	Introduction	1
1.1	Context	1
1.2	Motivation and objectives	2
1.3	Document structure	2
2	Real-time Location Systems	3
2.1	Introduction	3
2.2	Locating	4
2.3	Ranging and Angulating	6
2.3.1	Received Signal Strength	6
2.3.2	Time of Arrival	7
2.3.3	Time Difference of Arrival	8
2.3.4	Angle of Arrival	9
2.3.5	Other methods	9
2.4	Technologies	9
2.4.1	RFID	10
2.4.2	Bluetooth	10
2.4.3	Ultra-wideband	12
2.4.4	Wi-Fi	12
2.4.5	ZigBee	12
2.4.6	Ultrasound	12
2.4.7	Infrared	13
2.5	Summary	13
3	Project Overview	15
3.1	Problem Description	15
3.2	Approach	15
3.2.1	Selecting RTLS commercial solutions	16
3.2.2	Proposed Solution	16
3.2.3	Integrating RTLSs with the MES	17
3.3	Expected Results	19
4	Work Developed	21
4.1	Commercial RTLS Solutions Survey	21
4.1.1	Zebra Motionworks	22
4.1.2	AiRISTA Flow	22
4.1.3	Quuppa	22
4.1.4	Cisco	23

CONTENTS

4.1.5	Aruba Networks (HPE)	23
4.1.6	Mist Systems	23
4.1.7	Ubisense	24
4.1.8	AirFinder	24
4.1.9	LitumIoT	24
4.2	Developed RTLS Solution	25
4.2.1	Radar	25
4.2.2	Server	26
4.2.3	Manager	28
4.3	MES Integration	30
4.3.1	Connect IoT Task	30
4.3.2	FabLive	32
4.4	Results	32
5	Conclusions and Future Work	35
5.1	Objectives' achievement	35
5.2	Future Work	35
	References	37
A	Commercial Solutions Survey	39
A.1	Comparison of RTLS Solutions	39

List of Figures

2.1	Intersection of multiple spheres	5
2.2	Example of trilateration in 2D	6
2.3	Example of triangulation in 2D	7
2.4	RSSI vs Distance	8
2.5	Time-of-Arrival	9
2.6	Angle-of-Arrival	10
2.7	Structure of the BLE advertisement packet including <i>iBeacon</i> , <i>AltBeacon</i> , and <i>Eddystone</i> beacon protocols.	11
3.1	Proposed solution architecture	17
3.2	Connect IoT simplified architecture	18
3.3	Example of an Automation Workflow	20
4.1	Example of RSSI fluctuation	25
4.2	Manager's location view	29
4.3	Manager's radar view	29
4.4	Manager's beacon view	30
4.5	Workflow using the RTLS Task	31
4.6	RTLS Task Settings	31
4.7	FabLive - 3D model of Critical Manufacturing Headquarters	33

LIST OF FIGURES

List of Tables

2.1	Comparison of RTLS Technologies	14
A.1	Comparison of RTLS Solutions	39
A.1	Comparison of RTLS Solutions	40
A.1	Comparison of RTLS Solutions	41

LIST OF TABLES

Abbreviations

AoA	Angle of arrival
AOD	Angle of departure
BLE	Bluetooth Low Energy
GPS	Global Positioning System
IoT	Internet of Things
LOS	Line-of-sight
MES	Manufacturing Execution System
NLOS	None Line-of-sight
RF	Radio Frequency
RFID	Radio Frequency Identification
RTLS	Real-time locating system
TDoA	Time Difference of Arrival
ToA	Time of Arrival
ToF	Time of Flight
UWB	Ultra-wideband

Chapter 1

Introduction

In the last decade the satellite-based geolocation and navigation, assisted by cellular and WiFi networks, has become such a part of our lives that some might consider it a commodity. A large percentage of the world population owns a GPS capable smartphone, and more and more cars come with GPS navigation included. One of the limitations of the satellite-based positioning systems is that they require line-of-sight (LoS) to work properly, which means that indoor locating is not feasible [BCLN05]. Also, the accuracy of these systems is around 5 to 10 meters, which might be insufficient for most indoor case scenarios.

The role of Real-time location systems is precisely locating and tracking, in real time, objects or people within buildings or other confined areas [KY10, KBB12]. For certain areas, namely health and industry, RTLs can have enormous benefits, and it is expected that in the near future RTLs will have similar expansion as seen in the satellite-based systems [DSCD15]. But unlike outdoor systems, RTLs still rely on several different technologies. They haven't matured enough to the point that they converged to a single technology. Every technology has its advantages and its disadvantages and the diverse indoor environments (hospitals, factories, warehouses, shopping centers, etc.) may favor one over the other.

Most of RTLs use radio frequency technologies like Bluetooth or WiFi, which are often used in digital communications, turning the mobile devices to be tracked into potential IoT devices. This allows these devices to be used not only for tracking but additionally as IoT sensors - sensors that automatically synchronize their readings (e.g., temperature, pressure) with the main system.

1.1 Context

The proponent of this dissertation is Critical Manufacturing, a company founded in 2009 that develops automation and manufacturing software for Industry 4.0. Their main product is a Manufacturing Execution System (MES).

Industry 4.0 is the concept behind the so-called smart factories and the name comes from the fact that it is considered the fourth industrial revolution. It introduced interconnected cyber-physical systems based on the most recent digital technological advancements, as the *Internet of Things*, to further improve productivity in factories.

A Manufacturing Execution System can be described as:

“[...] an information system that connects, monitors and controls complex manufacturing systems and data flows on the factory floor. Its main goal is to ensure effective execution of the manufacturing operations and improve production output.” [Rou17]

An MES manages the complete production cycle including resource scheduling, overall equipment effectiveness, materials track and trace, etc. In the current competitive, fast-changing world, it is of vital importance to have an MES implementation that is able to interact with every aspect of a plant's day-to-day operations.

1.2 Motivation and objectives

IoT is already a big part of Industry 4.0, but it is believed that RTLS will also have a major role in industries in the near future. Whether it is to locate personnel, tracking assets or managing inventory, the obtained information can have great value for optimizing the industrial processes. Information is not only helpful to the decision makers but can also be useful to artificial intelligence systems. Because there isn't a "one-size-fits-all" RTLS solution, Critical Manufacturing wants to be able to recommend, if requested, the one that best adapts a certain client's context and needs. Also, it is important to be able to integrate a particular RTLS solution into the MES if a client requests so.

This dissertation has two main objectives. The first is to find one or more commercial solutions of RTLSs that best fit different industrial scenarios. These solutions should include, if possible, other types of sensors in addition to the location ones like accelerometers and gyroscopes (to register vibrations), temperature sensors, etc. The second objective is to integrate an RTLS solution into the Critical Manufacturing MES. This integration will allow, for example, to track people, equipment, and materials within facilities in real time using the existing MES 3D visualization engine. It will also allow checking the history of the readings of other sensors using the rich data visualization capabilities of the MES that include different types of charts.

1.3 Document structure

This document is composed of 4 more chapters besides this one. Chapter 2 presents the state-of-the-art regarding real-time locating systems. Chapter 3 describes the problem, the selected approach as well as the expected results of this dissertation. Chapter 4 details all the work developed under this dissertation and discusses the results obtained. Finally, chapter 5 presents conclusions and future work.

Chapter 2

Real-time Location Systems

This chapter presents a review on the state-of-the-art of the Real-time Location Systems, also referred to as Real-time Locating Systems, focusing on the several different technologies currently available, including their advantages and disadvantages, as well as the different locating methodologies.

2.1 Introduction

Real-time location systems are used for indoor locating and/or tracking within a confined area [KY10, KBB12, LLC⁺09, LDBL07] in contrast with satellite-based locating systems, like GPS, that provide global outdoor coverage [BCLN05].

Although RTLS exist for more than a decade, the recent technological advancements have improved some key aspects like accuracy, range, and power consumption while reducing overall cost contributing to the viability and widespread of these systems. Some of the Real-time Locating technologies currently available are RFID (passive or active), Bluetooth, Wi-Fi, Ultra-wideband (UWB), Ultrasound and Infrared [DSCD15, TGLP08]. Because there are several different implemented technologies, each with its trade offs [Gaf08] (there is not a “one size fits all” solution), no established standard has emerged yet [CGK⁺18]. However, some key aspects of RTLS, such as a unified Application programming interface (API), are already standardized under the ISO/IEC 24730 series [fS14].

An RTLS comprise the mobile elements to be located, hereinafter referred to as *tags*, the fixed points that constitute the *infrastructure*, hereinafter referred to as *anchors*, and the software that computes and stores the positional data generated (*engine*), to be presented directly to users or to other systems [CLCL16, KBB12]. Generally, an RTLS works by having each tag send some sort of signal periodically that is picked up by one or more anchors, that in turn send the information received to the engine to compute the positions. Anchors usually work as readers as they are listening to signals sent by tags (which work as beacons), but the inverse is also possible and common with some technologies, i.e., the anchors working as beacons and the tags as readers. Some methods even require a signal to be sent back and forth as will be seen ahead.

Some of the more important features that characterize an RTLS are:

- **Accuracy and Precision** — They are both used to determine the reliability of the obtained positions. Accuracy refers to the closeness between an estimated position and the real position while precision refers to the closeness between two or more estimated positions to each other under the same conditions;
- **Battery** — How long does the battery of tags last and if they can be recharged or are the tags disposable;
- **Scalability** — Refers to the ability of the system to handle a growing number of tags and/or anchors;
- **Range** — Corresponds to the maximum distance a tag and an anchor can communicate. The coverage of the system and its accuracy normally depend on the number and distribution of the anchors and their range;
- **Latency** — Can refer to both the time it takes to compute a position as well as the maximum rate a position can be updated;
- **Cost** — The total cost of a solution is the combination of the infrastructure cost (total cost of anchors), the cost of each tag and the expected number of tags. The costs of the infrastructure and each tag are usually inversely proportional, as the cost is related to the complexity of the hardware, and the complexity tends to shift between the infrastructure and the tags.

2.2 Locating

Locating can be divided into two different systems: proximity systems and relative coordinates systems [KPO15].

In **Proximity Systems** the signal from a moving tag is received by a single fixed anchor¹, thus acknowledging the tag is within range of the anchor [RPE05, LDBL07]. The accuracy is therefore inversely proportional to the anchor's range. Usually, if more than one anchor receives the signal from the same tag the system only acknowledges proximity to the anchor that received the strongest signal [ZYC⁺16]. To provide real-time location and tracking with good accuracy these systems require a dense grid of anchors, thus they are more useful for room-level accuracy (in facilities with many small divisions like hospitals it may be enough to know the room the tag is located).

In **Relative Coordinates Systems**, the signal from a tag is received by a multiple number of anchors² and its position is estimated using one or more locating algorithms, such as trilateration,

¹For simplicity, it is assumed that tags work as beacons and the anchors work as readers but it should be noted that the inverse is also valid.

²Same as the previous footnote

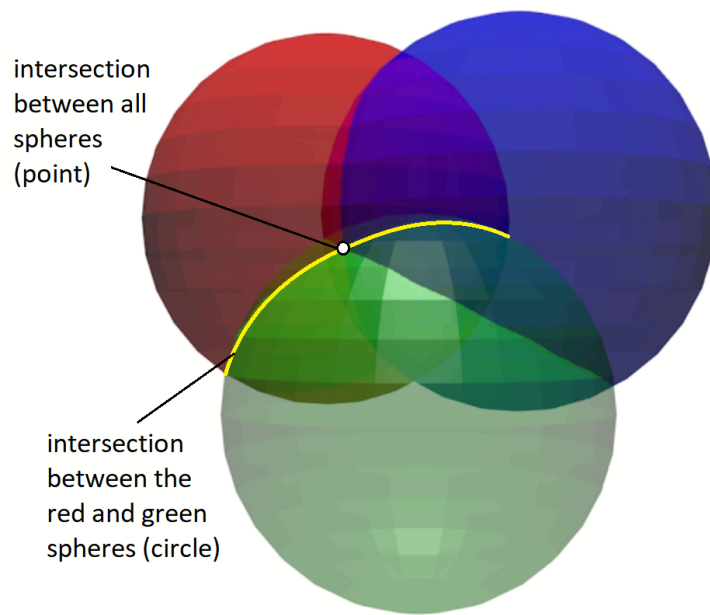


Figure 2.1: Intersection of multiple spheres

multilateration, or triangulation. These systems require anchors with good range and carefully positioned, so their range areas overlap.

Trilateration and **True Range Multilateration** algorithms use estimated distances from each anchor to a tag to calculate its approximate location based on geometry [LDBL07]. The position of the tag is the intersecting of multiple spherical surfaces (or circumferences if only considering a 2-dimensional problem) where each of their centers and radius correspond to an anchor's position and its distance to the tag respectively. The intersection of two spherical surfaces results in a circumference, that, if intersected with a third spherical surface, results in two points at most (see Fig. 2.1). Usually, the second point called ambiguous point can be easily dismissed (e.g. in GPS one of the points is on the surface of the earth and the other is in the outer space) resulting in three spherical surfaces being enough to obtain a position. More spherical surfaces (anchors) are useful for improving accuracy. Figure 2.2 exemplifies trilateration in a 2-dimensional scenario.

Triangulation works similarly to trilateration but uses angles instead [LDBL07, RPE05]. Considering a 2 dimensional problem, it is possible to obtain the position of a tag with only two anchors³ and the direction of the tag relative to each one of these anchors (see Fig. 2.3).

The methods used to obtain said distances and angles are presented in section 2.3

³If the anchors and tag are aligned, an additional anchor that is not in the same alignment is needed to obtain the tag's position

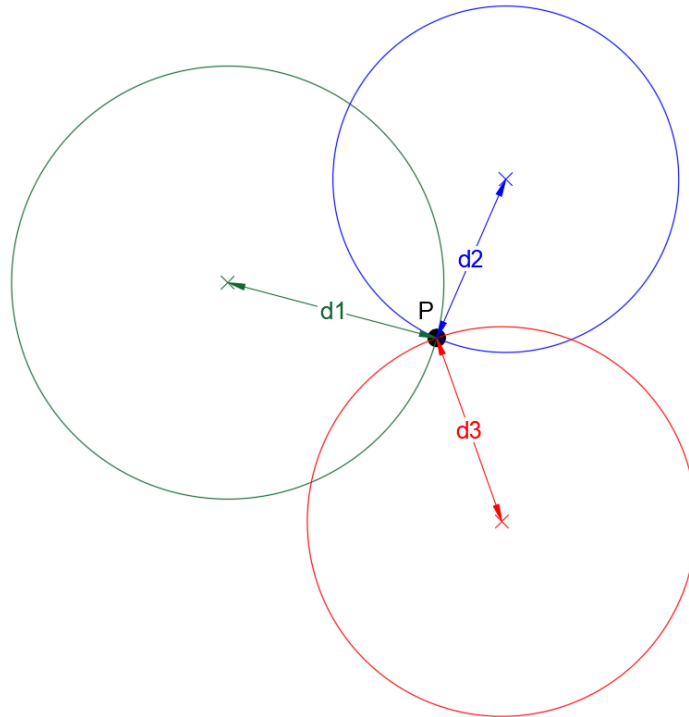


Figure 2.2: Example of trilateration in 2D

2.3 Ranging and Angulating

There are several different methods for determining distances (ranging) and for determining angles (angulating) used in locating. The method (or combination of methods) used depends largely on the technology in question. Some of these methods are briefly described below.

2.3.1 Received Signal Strength

Received Signal Strength (RSS) is based on the principle that the signal strength decreases with the distance between two devices. It is one of the most commonly used methods due to its availability (it is inherent to wireless systems) and low cost. It also doesn't require synchronization of clocks but is sensitive to multipath interference (reflected signals) [DSCD15, Gaf08].

The Received Signal Strength Indicator (RSSI) is usually calculated using the log-distance path loss model [Gaf08] (see Fig. 2.4) and is expressed as:

$$RSSI = T_x - 10n \log d \quad (2.1)$$

where d is the distance between both devices, n is the path-loss exponent that is related to the transmission environment (a value between 1 and 5 where a lower value represents a less obstructed environment), and T_x is a constant that represents the reference RSSI value at one meter distance.

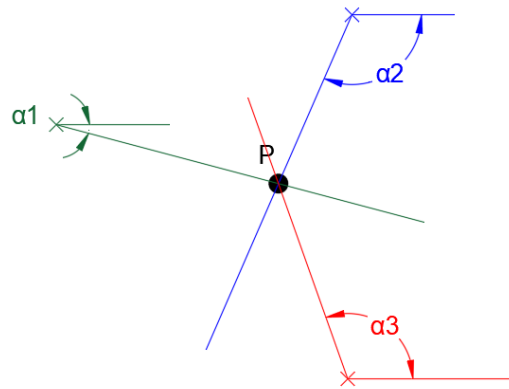


Figure 2.3: Example of triangulation in 2D

The distance can be obtained from the RSSI by changing expression 2.1 to:

$$d = 10^{\frac{T_x - \text{RSSI}}{10n}} \quad (2.2)$$

Since T_x is device dependent this method usually requires calibration.

Because radio waves are easily influenced by external factors such as absorption, interference, or diffraction, the measured RSSI value can be quite inconsistent, which affects the accuracy of this method. The inconsistency of RSSI increases with distance. There are some methods for improving accuracy, such as using Kalman filters and/or neural networks [DSCD15, Gaf08].

2.3.2 Time of Arrival

Time-of-Arrival (ToA) or Time-of-Flight (ToF) calculates the distance based on the time a signal takes to travel between devices and the signal's speed. It can be a one-way ToA, which requires clock synchronization between sender and receiver, where the receiver knows the instant the signal is sent and calculates distance using the following expression [Gaf08]:

$$d = (t_1 - t_0) v \quad (2.3)$$

where t_0 is the instant the signal is sent, t_1 is the instant the signal is received (see figure 2.5-a) and v is the traveling speed of the signal (e.g., speed of light in the case of radio-frequency signals or infrared).

ToA can also refer to two-way ToA or Round Trip Time (RTT) where the distance is calculated based on a signal round-trip delay time, i.e., the total time it takes for a signal to be sent and for an acknowledgment of that signal to be received [DSCD15]. This method doesn't require clock

Real-time Location Systems

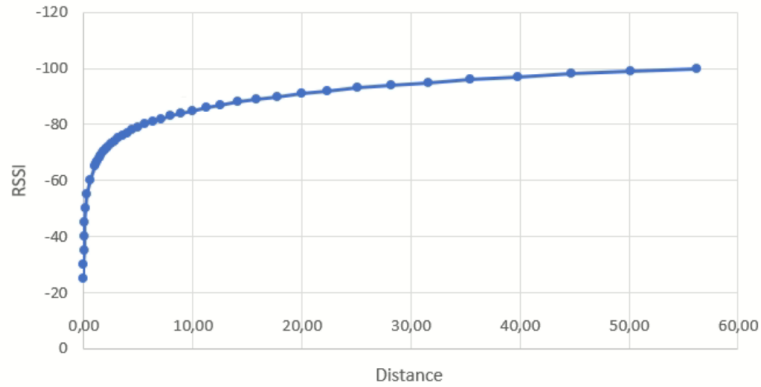


Figure 2.4: RSSI vs Distance

synchronization but requires the receiver to reply to the sender. The expression to calculate the distance is the following:

$$d = \frac{(t_3 - t_0) - (t_2 - t_1)}{2} v \quad (2.4)$$

where t_0 is the instant the signal is sent, t_1 is the instant the signal is received, t_2 is the instant the reply signal is sent, t_3 is the instant the reply signal is received and v is the traveling speed of the signal (see figure 2.5-b). Because there is no clock synchronization, the difference between t_2 and t_1 is sent in the reply signal.

There is another way of calculating distance using ToA without clock synchronization which involves sending two signals with different traveling speeds (see figure 2.5-c). The system of equations would be the following:

$$\begin{cases} d = (t_1 - t_0) v_1 \\ d = (t_2 - t_0) v_2 \end{cases}$$

where the two unknowns would be the instant the signals are sent (t_0) and the distance (d). This method can also be used to synchronize clocks.

2.3.3 Time Difference of Arrival

Time-Difference-of-Arrival (TDoA) method is based on the ToA of different synchronized fixed nodes. It has two different approaches whether the fixed nodes (anchors) are the signal senders (beacons) or receivers (readers). In the case of senders, all fixed nodes send a synchronized signal and the mobile node calculates a TDoA based on the ToA of each sender's signal. In the case of receivers, all nodes share their ToA relative to the same signal sent by a mobile node and calculate the TDoA [DSCD15].

Real-time Location Systems

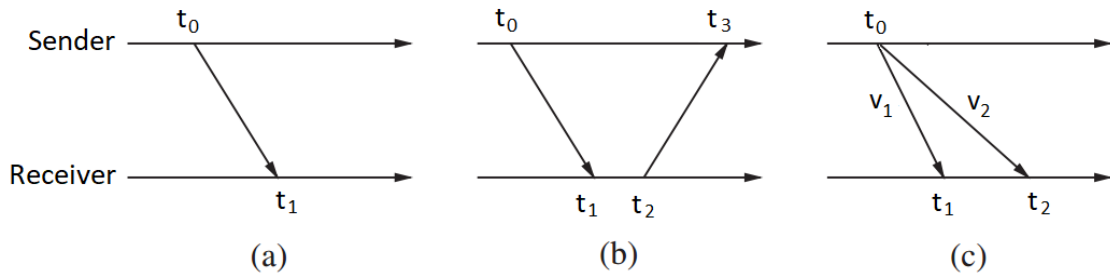


Figure 2.5: Time-of-Arrival

TDoA-based approaches require the clocks of the anchors to be tightly synchronized (but not the clocks of the tags) and can obtain very accurate measurements. TDoA is similar to the method used in GPS except that in GPS, satellites send their respective position in addition to the timestamp.

2.3.4 Angle of Arrival

Angle-of-Arrival (AoA) is a method used to calculate angles by using an array of antennas and comparing the different ToA of each antenna for the same signal. The angle of incidence of a signal can be obtained by the following expressions:

$$\delta = v \Delta t \quad (2.5)$$

$$\delta = d \sin \theta \quad (2.6)$$

$$\theta = \arcsin \frac{v \Delta t}{d} \quad (2.7)$$

where the expression 2.7 is obtained from expressions 2.5 and 2.6, and Δt is the time difference of arrival of the signal between two antennas in the array, v is the traveling speed of the signal and d is the distance between those antennas (see figure 2.6).

This method is usually used in combination with others as it is sensitive to multipath [DSCD15].

2.3.5 Other methods

Other not so used methods but worth mentioning are *Phase-Difference-of-Arrival*, *Line-of-sight (LoS)*, and *Proximity* [DSCD15].

2.4 Technologies

This section presents the most common technologies used in RTLS. The majority is based on radio frequency, but there are some alternatives like infrared and ultrasound.

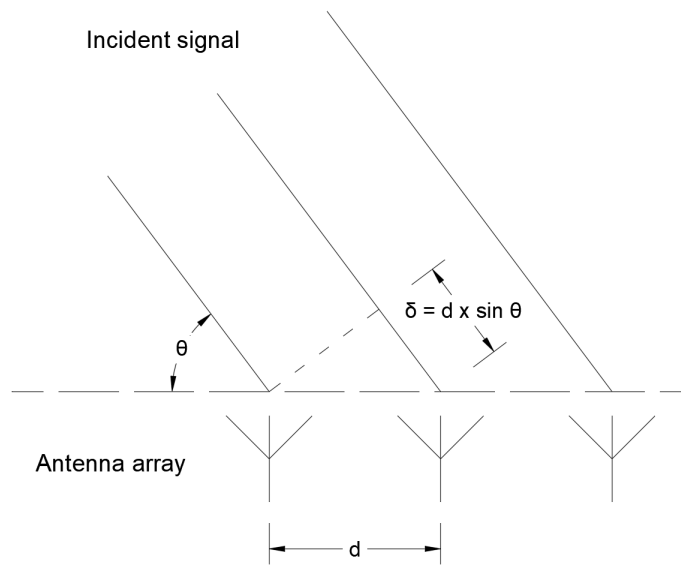


Figure 2.6: Angle-of-Arrival

2.4.1 RFID

Radio frequency identification consists of small integrated circuits called tags that store an identification number which can be retrieved by a compatible reader with the use of electromagnetic radio frequency waves. RFID technology has been around for some decades and is commonly used to replace barcodes for labeling items as it doesn't need line-of-sight between the reader and the tag [TGLP08]. RFID tags can be passive or active [TGLP08].

Passive tags don't have a battery as the radio waves emitted by the reader provide enough energy to transmit a response. This, of course, limits the reading range to a few meters, but in return reduces the size and cost of each tag. In an RTLS point-of-view, the limited range eliminates the possibility of proper tracking as it would require a prohibitive number of readers both in terms of cost and space. Another potential problem of passive tags is tag collision, which happens when a reader can't process simultaneous responses from different tags, thus compromising scalability.

Active tags have their own power source effectively increasing the reading distance to several meters (up to 100m) but also increasing in size and cost [TGLP08]. Active tags don't activate in the presence of an RFID reader, but transmit a signal periodically instead. The major drawback of the active tags in terms of RTLS is the average accuracy of the technology.

2.4.2 Bluetooth

Bluetooth is a short-reach radio frequency technology used for short-range data communications. It is used in wireless personal area networks (WPANs). It implements adaptable variable frequency which allows switching between frequency channels to minimize interference from other radio sources. Bluetooth range depends on the device class, with class 2 having a typical range of around

Real-time Location Systems

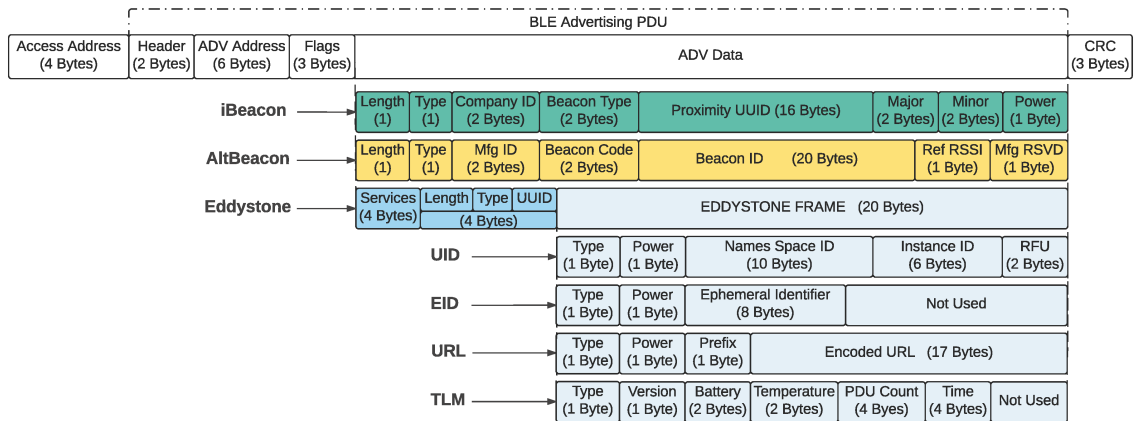


Figure 2.7: Structure of the BLE advertisement packet including *iBeacon*, *AltBeacon*, and *Eddystone* beacon protocols.

10 meters, while the more industrial focused class can reach up to 100 meters at the expense of higher power.

Bluetooth version 4.0 introduced the Bluetooth Low Energy (BLE) protocol that allows significantly reduced power consumption without sacrificing the communication range. Bluetooth subsequent versions have introduced other features specifically aimed at IoT. The most recent version 5.1 specification has a special focus on indoor tracking by introducing direction estimation (using the AoA method), and improving the advertising mode.

BLE advertisement consists of a device periodically sending information to any BLE peripherals within range without explicitly connecting to them (broadcasting). This information is usually used for devices to discover each other and establish connections but can be used for other purposes namely location. Devices such as beacons only transmit data and don't establish connections with other devices. Apple's proprietary protocol *iBeacon*, as well as Radius Network's *Altbeacon*, and Google's *Eddystone* use the advertising packets to broadcast information. Figure 2.7 shows how each of the protocols makes use of the BLE advertisement packet [HRFCLE17].

BLE advertising packets contain the MAC address that identifies the device. A feature known as BLE Privacy allows replacing the MAC address within the advertising packets with randomly generated values at determined intervals. This MAC randomization prevents listeners from using MAC addresses to build a history of device activity, thus increasing user privacy. Recent versions of Android and iOS implement BLE MAC randomization.

From an RTLS point-of-view, Bluetooth offers low cost and low power consumption with an average range. Range can be improved at the expense of battery. It also offers good accuracy [RPE05, RLJ⁺15] if enough fixed-point devices are used. Since the infrastructure cost is relatively low that shouldn't pose a problem. The new standard will allow to further improve its accuracy (based on the direction feature), scalability and reliability (based on the improved advertising).

2.4.3 Ultra-wideband

Ultra-wideband is a radio frequency that transmits a signal over several frequency bands simultaneously making it resistant to interference from other radio sources and multipath.

UWB has been used recently in RTLS because it can send short low energy pulses - which allows very precise delay estimates, that translate in great location accuracy [MHN⁺18, LLC⁺09] - with a large signal range (around 30 meters). It also means it has very good response time and can be even more power efficient than RF active tags, with a much larger signal range. The UWB although, can be susceptible to metal interference and the infrastructure is costly and requires complex installation [CLCL16].

2.4.4 Wi-Fi

Wi-Fi is a well known radio-frequency technology based on the IEEE 802.11 standards used in wireless local area networks. Wi-Fi focuses mostly on high data rates - with the latest versions achieving speeds of over 1 Gbit/s - resulting in expensive and energy demanding hardware.

In terms of RTLS, Wi-Fi provides poor range and accuracy (3 to 5 meters) [BCLN05] and one of the few advantages is that most buildings already have an WiFi infrastructure in place and existing smartphones could be used as tags effectively reducing the investment cost. Still, software for RTLS would be needed and the accuracy and range would depend on the distribution of the Wi-Fi access points (RTLS readers). Increasing the number of access points to increase coverage and accuracy can be counter-productive as it will increase costs and probably impact the performance of the WLAN, as access points too close to each other tend to cause interference.

2.4.5 ZigBee

ZigBee is another low power radio frequency technology intended for wireless personal WPANs. Compared to Wi-Fi, ZigBee has lower power consumption and lower cost at the expense of a lower data rate. Much like Bluetooth, ZigBee's range is somewhere between 10 and 100 meters depending on power output and environmental characteristics. Although ZigBee supports other topologies, it usually uses a mesh topology - unlike Wi-Fi that uses a star topology - meaning nodes in the network are connected to each other, allowing better scalability and reliability.

The biggest drawback of ZigBee, concerning RTLS, is the poor accuracy (around 5 meters). Nevertheless, if only room-level accuracy is desired, ZigBee may be a viable choice.

2.4.6 Ultrasound

An alternative technology for RTLS is the use of ultrasounds. It works similarly to UWB except it uses sounds instead of radio waves. Tags send ultrasonic pulses (hence not audible to humans) to neighboring readers that calculate position. Ultrasound systems require line-of-sight as sound waves are more sensitive to obstacles than radio waves but on the other hand, are very accurate and less prone to multipath. Ultrasounds are also susceptible to interference from other sound

sources (e.g., noises). Ultrasounds can be a good alternative to passive RFID in environments with multiple walls.

2.4.7 Infrared

Infrared-based locating systems use infrared light pulses. Tags send infrared pulses to neighboring readers that calculate position. As ultrasound systems, infrared systems also require line-of-sight as light doesn't go through opaque obstacles. Infrared can guarantee room-level accuracy but aside that they have limited range (few meters) and are susceptible to interference from other infrared sources (e.g., sunlight).

2.5 Summary

After a comprehensive review of the current state-of-the-art in Real-time location systems, one can conclude that radio technologies are more appropriate in open spaces characteristic of warehouses and manufacturing facilities. From these technologies, UWB, Bluetooth and RFID seem the most complete options. Nonetheless, all technologies can have their use in more specific scenarios depending on the requirements of the system. Table 2.1 provides a comparison between the RTLS technologies referred to in this chapter accordingly with some of the key characteristics.

Bluetooth, from version 4.0 onwards, is becoming the technology of choice for IoT and possibly also for RTLS. The introduced features in version 5.1 are a step forward for RTLS, but the fact that this specification was only recently released means that it will take some time before it can be found in a commercial solution.

Among location methods, RSSI is the easiest to implement because it is inherent to RF systems, but it is also the least accurate. One-way ToA and TDoA are more robust methods but require clock synchronization between devices, which can be hard to accomplish and maintain (expensive setups). Two-way ToA has the robustness of the previous methods without the need for clock synchronization, but adds complexity to the devices and can also have a negative impact on battery life, latency and scalability of an RTLS. AoA provides additional accuracy but requires an array of antennas and is sensitive to multipath.

Real-time Location Systems

Table 2.1: Comparison of RTLS Technologies

Technology	Accuracy	Range	Battery Life	Cost	Strengths	Weaknesses
BLE	2-3 m	10-20 m	Good	Low	Ubiquity; Good balance between accuracy and total cost; Scalability	Shorter range than some alternatives (can be increased at the expense of battery)
Infrared	sub-meter	2-3 m	Good	Average	No false positives	Requires LoS; Sensible to interference; Short range
RFID (active)	2-3 m	20-30 m	Good	Low	Low latency	Not many advantages over BLE
RFID (passive)	2-3 m	2-3 m	N.A.	Very low	Tags don't need battery and are very cheap	Provides only proximity
Ultrasound	sub-meter	~10 m	Good	High	No false positives	Requires LoS; Sensible to interference
UWB	sub-meter	20-30 m	Good	High	Has the best accuracy	High tag and infrastructure price
Wi-Fi	3-5 m	20-30 m	Poor	High	Can use existing infrastructure and devices	Worse than other technologies in almost every aspect
ZigBee	3-5 m	20-30 m	Good	Low	Can use a mesh topology	Not many advantages over BLE

Chapter 3

Project Overview

3.1 Problem Description

Benefits from locating and tracking assets in industry have led many manufacturers to consider implementing Real-time locating systems. But because there isn't a standard technology for Real-time locating systems and because each industrial scenario can have different requirements and constraints, the selection of the most suitable system, by a manufacturer for its facilities, can be difficult.

Some manufacturers may be uncomfortable with cloud-based solutions, where others may prefer the versatility provided by them. Not all scenarios require pin-point accuracy, but in some cases, the added cost of such accuracy is justified. A scenario where several tags are to be tracked might require a solution with cheaper tags but more expensive infrastructure, whereas large facilities can require systems with a cheaper infrastructure. These are some examples of factors that can determine the appropriate RTLS.

That being said, some manufacturers have already made inquiries to Critical Manufacturing about Real-time locating systems and increasing demand for integrating these systems is expected. This results in adding such functionality to the MES, of relative importance.

3.2 Approach

This section describes the proposed solution to address the problem described in section 3.1 that is divided into three parts.

It is presented first how the process of screening, evaluating and selecting appropriate RTLS solutions for the manufacturing industry is made. Secondly, the development of an own RTLS solution for Critical Manufacturing's facilities, as a proof of concept, is described. Finally, the approach to integrate an RTLS solution into the Critical Manufacturing MES is also described.

3.2.1 Selecting RTLS commercial solutions

The task of screening and evaluating appropriate RTLS solutions had two goals. The first was for Critical Manufacturing to have a knowledge base of current commercial solutions to recommend to its clients according to their needs and context. The second goal was to find one or more vendors that would provide their solutions to test and integrate into Critical Manufacturing MES.

The work involved:

- Finding as many credible RTLS commercial solutions available as possible;
- Sorting them according to key characteristics such as technologies used or cost;
- Electing the most promising solutions and contacting their vendors for additional information and possible trials.

The information to be obtained for each solution included:

- **Engine** — Whether it is on-premises or based on the cloud or either;
- **Technologies** — What technologies are used;
- **Accuracy** — What is the minimum accuracy claimed by the vendor;
- **User Interface** — Whether it has an UI for managing the system and is system dependent;
- **API** — Whether it has an API for integrating with the MES and what protocols are supported;
- **Tags** — What types of tags does it provide and do they include additional sensors;
- **Pricing** — What are the prices for the infrastructure, for each tag and for the software.

3.2.2 Proposed Solution

Since there was no guarantee that there would be a vendor that would provide their solution for testing and to integrate into the MES, an own solution was designed that fit the test scenario (tracking personnel at Critical Manufacturing's facilities) and so that no major investment in hardware was needed. The solution involved BLE as technology and RSSI as the method for ranging. The choice of Bluetooth was decided because each company employee has a laptop computer with this technology and the open-plan design of the office provides an excellent distribution of the laptops to work as anchors. The choice of RSSI was by elimination, since ToA and TDoA require tight clock synchronization (which wasn't realistic), AoA requires specialized hardware, and two-way ToA doesn't allow using normal beacons as tags.

The proposed solution comprises:

Project Overview

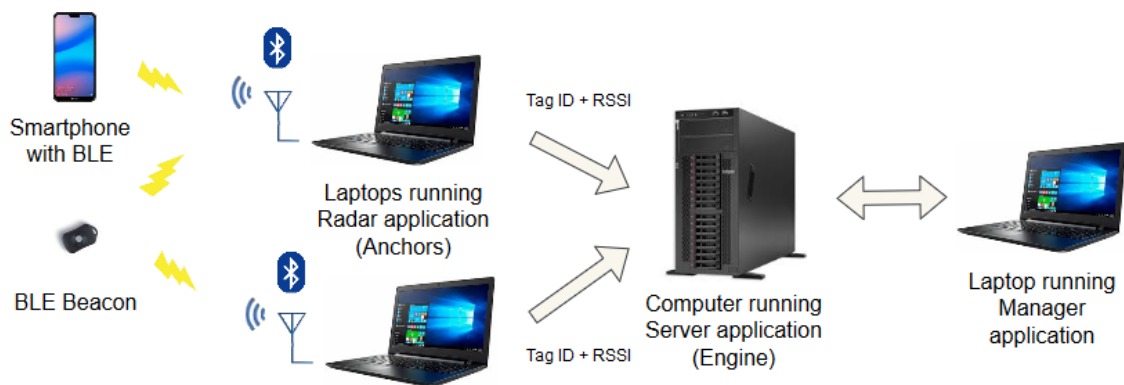


Figure 3.1: Proposed solution architecture

- **Radar Application** — a windows compatible application running on the laptop computers that scans BLE advertisement packets and sends information to the engine;
- **Server Application** — a cross-platform application that works as the RTLS engine (receives information from the radars and computes positions) and provides an external API for integration;
- **Manager Application** — a windows compatible application with a GUI for managing the RTLS and visualize the location of tags.

Figure 3.1 illustrates the architecture of the proposed solution.

3.2.3 Integrating RTLSs with the MES

Critical Manufacturing describes its MES as an innovative software platform with a deep set of modular applications that provides manufacturers in complex industries with maximum agility, visibility, and reliability. It is built on Microsoft application development layers, and HTML5 and Angular user interface technologies [Man].

The Critical Manufacturing MES has a built-in module named *Connect IoT* that serves as a lightweight connectivity layer for any type of equipment or device, with any protocol. Its goal is to dramatically reduce the time and effort to implement equipment or IoT integration [Man].

An MES can have multiple instances of the Connect IoT, each corresponds to an **Automation Manager**. Each *manager* contains the following processes:

- **Automation Monitor** — is responsible for starting and monitoring all the other processes. Each *manager* has exactly one *monitor*;
- **Automation Controller** — is a programmable component that allows processing messages received from equipment/IoT devices and/or sending them messages. Each *manager* can have several *controllers*;

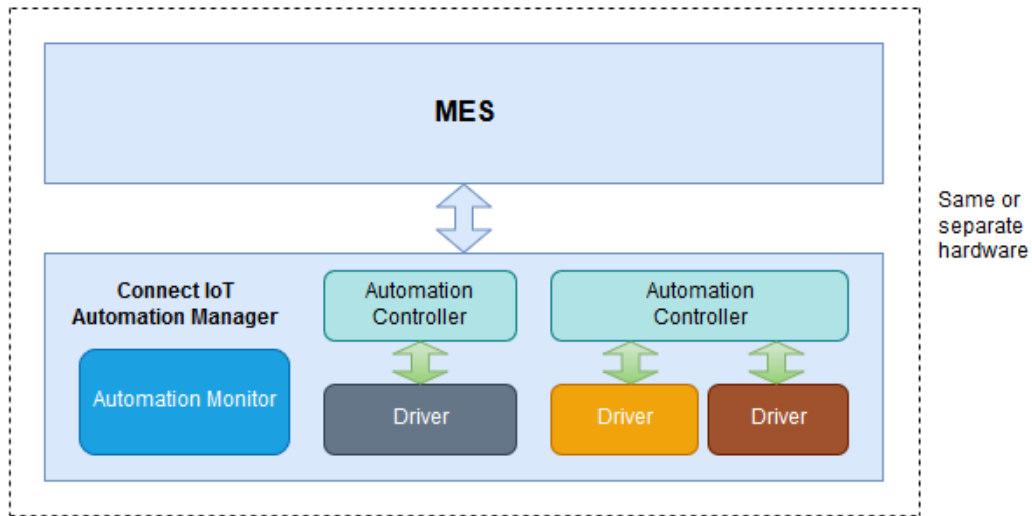


Figure 3.2: Connect IoT simplified architecture

- **Automation Driver** — is responsible for translating messages between a *controller* and an equipment/device using a specific protocol. Each *driver* is associated with a single *controller* and a *controller* can have multiple *drivers* associated with it.

Figure 3.2 illustrates the architecture of ConnectIoT.

The module contains drivers for some of the most common protocols but also allows adding new protocols. The following list contains some of the protocols included out-of-the-box:

- Serial
- TCP/IP
- BLE (Bluetooth Low Energy)
- MQTT
- SECS/GEM
- OPC-UA
- OPC-DA

The behavior of an *Automation Controller* is defined by **Automation Workflows**. Workflows are programmed using a graphical interface with easy drag-and-drop logic and require no code as shown in figure 3.3.

Workflows are completely abstracted from the specific driver details. Each *workflow* is composed of logic elements called **Tasks** that process a specific code and contain inputs and outputs that can be connected to other *tasks*.

Project Overview

This part of the project involves developing a *Connect IoT Task* that processes messages received from an RTLS and updates the respective entities coordinates on the MES database. Although the *task* works independently of the used protocol, it should be able to receive differently formatted messages.

3.3 Expected Results

In the end, it is expected to be able to visualize in real-time the location of a given object in the existing Critical Manufacturing 3D fab visualization engine (Fablive) and also show a data series in real time of a sensor's readings. For the RTLS, the results will be evaluated by comparing location in the real world with the representation in the 3D visualization engine. For other types of sensors, the comparison will be made between readings at the sensor and information stored in the MES.

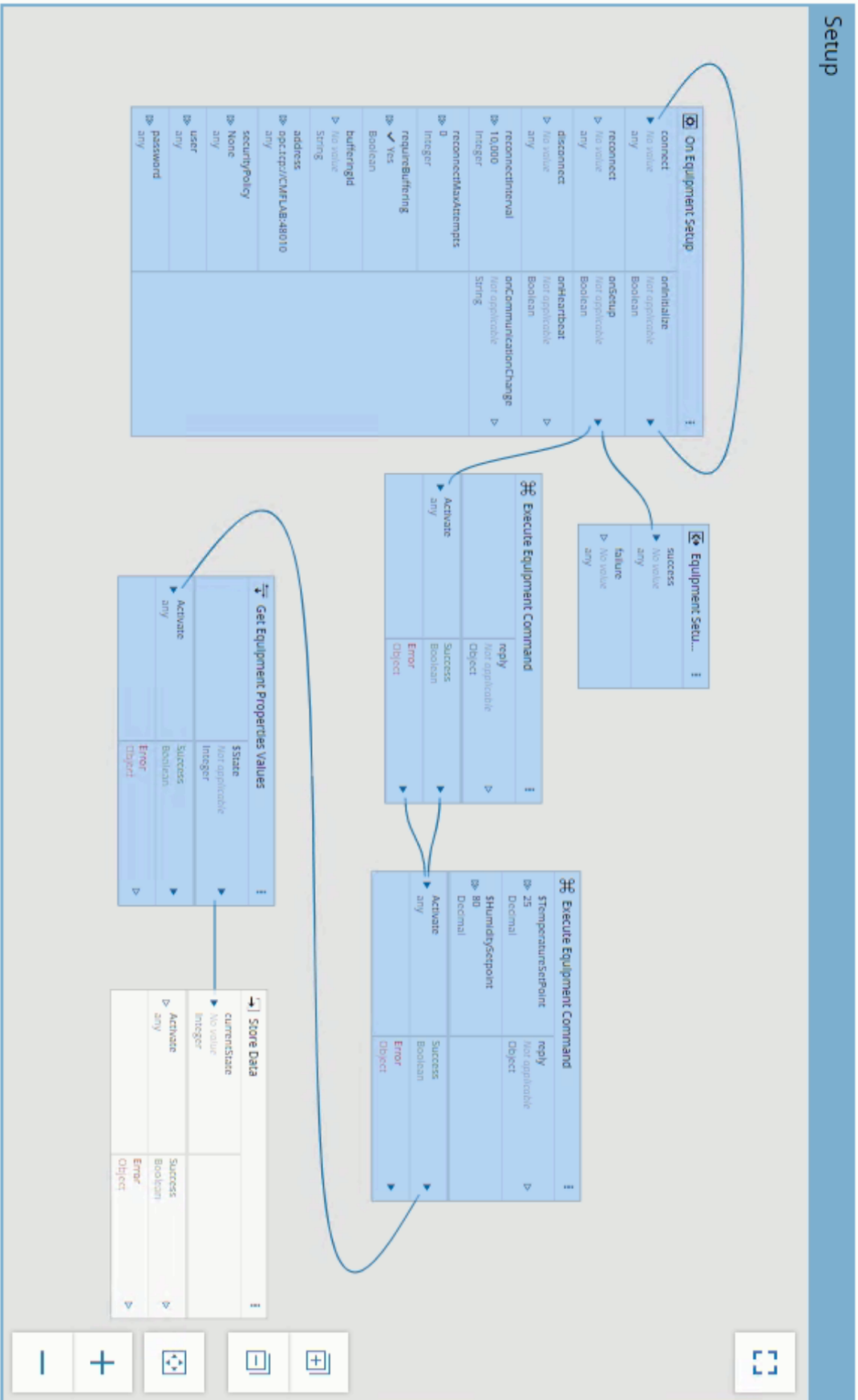


Figure 3.3: Example of an Automation Workflow

Chapter 4

Work Developed

This chapter presents in detail all the work developed for this dissertation, as well as an appreciation and discussion of the results obtained.

4.1 Commercial RTLS Solutions Survey

The first part of the work developed consisted of a survey on RTLS solutions currently on the market. The first step of the survey involved collecting commercial solutions available in the market using an Internet search engine. One of the search results was a publication regarding Indoor Location Services of a major advisory firm that provided some information on the biggest RTLS vendors worldwide. Several of the selected vendors were obtained through this publication; the others came from the top results of the search engine.

The next step involved obtaining more information for each of the selected solutions by researching on their respective websites. The research focused on the technologies used, type of engine, advertised accuracy, APIs, types of tags, etc. Based on the obtained information, the solutions were then sorted accordingly with appropriateness to different RTLS scenarios.

The final step involved contacting the vendors with the most relevant solutions for additional information - especially costs since any seller only discloses prices upon consultation - and the possibility of a trial.

The search resulted in a selection of 19 RTLS solutions. Regarding technologies used, it has been found that all solutions use at least BLE or UWB. There are solutions that use Wi-Fi or RFID in addition to one of these. Most vendors offer the option to run the engine on-premises or on the cloud, while some are restricted to the cloud and a few to on-premises. Most solutions offer an open API for third-party software, being REST and Pub/Sub the most common types. Concerning tags, most vendors offer many choices including different formats, such as badges, optional rugged design (i.e., shock, water and/or dust-proof), or optional sensors. Most solutions use proprietary

Work Developed

tags but some are compatible with third-party tags. Also worth noticing is that some solutions used tags as beacons, while others used the anchors. A few solutions could even use both methods.

Information retrieved for each solution can be seen in table A.1 in the Appendix.

Pricing is a more delicate issue. Vendors only provide prices upon contact and most of them did not respond to the inquiries. Only a few were able to provide prices, and for a specific scenario. In the matter of providing a trial for testing and integration, the prices presented were considered too high given the purpose. A tendency can be observed from the prices provided: vendors are using subscription models for the software component of their systems. These annual license fees can either be in the form of a fixed price for the engine, or additional fees for each tag and/or anchor, or even based on the area to cover. This can reflect that vendors are investing more in the software part to differentiate from the competition, such as using artificial intelligence for improved accuracy or offering optional modules for data analysis.

The following subsections present individually the most relevant solutions found as a result of this survey.

4.1.1 Zebra Motionworks

Zebra Technologies offers an RTLS solution called MotionWorks. Zebra's solution stands out because it offers a wide range of technologies including Wi-Fi, UWB, GPS, BLE, and both passive and active RFID, providing different accuracies/costs covering many location scenarios. It also allows running its *Savannah* location engine either on-premises or on the cloud, and offers two different ways of using the API (HTTP REST or XML pub/sub). It uses proprietary tags with IP67 rating (water and dust protection) and several years of battery life.

4.1.2 AiRISTA Flow

AiRISTA Flow is an experienced vendor in the RTLS market that offers customizable RTLS solutions based on multiple technologies and multiple ranging methods for enhanced location accuracy. It uses Wi-Fi, BLE, RFID, and IR together with RSSI and TDoA and its location engine can be deployed both on-premises or in the cloud. It has many different types of tags including tags that can integrate multiple technologies and sensors (motion, vibration, temperature, and humidity) and rugged design.

4.1.3 Quuppa

Quuppa is a company founded by former employees of the Nokia Research Center that offers a particular solution based on BLE and AoA. The vendor promises sub-meter accuracy down to 10 cm with low latency and high update rate allowing, for example, collision avoidance between people and vehicles in industrial environments, or following a hockey puck in real-time.

Quuppa uses a proprietary protocol based on BLE but they claim that any BLE device can be used as a tag with just minor software/firmware modifications. They provide off-the-shelf tags

Work Developed

made by Quuppa or its partners, firmware libraries for custom-designed tags and libraries for Android and iOS. The standard Quuppa tag is very versatile as it is small, lightweight and IP67 certified (water and dust protection) and its battery can last a few years. It also has accelerometer and thermometer sensors. Quuppa system is also able to receive and expose additional sensor data from the tags.

The system can be run either locally or in the cloud and provides an open API for integration.

4.1.4 Cisco

Cisco is one of the world's largest suppliers of networking hardware, which is by itself a reason to be considered a relevant player in the RTLS market. It leverages its wireless access points to offer additional location capabilities to clients. Its solution can track Wi-Fi and BLE devices based on RSSI and AoA (with compatible APs) providing different levels of accuracy. Its Hyperlocation feature based on AoA promises accuracy within 1 to 3 meters. It offers two different solutions, one for on-premises and the other for the cloud, along with an API for external access to the location information. It relies on partners to provide compatible tags.

4.1.5 Aruba Networks (HPE)

Aruba Networks is a subsidiary of Hewlett Packard Enterprise and is one of the largest vendors of RTLS solutions. Much like Cisco, Aruba uses its Wi-Fi/BLE APs as RTLS anchors in order to optimize customers' investment in hardware. Its solution can only be deployed to the cloud and uses BLE beacons for location alongside Wi-Fi for communicating with the engine. The system can be used as asset tracking with proprietary tags (third-party tag support is planned) or it can be used with BLE beacons, smartphones and applications developed using Aruba's SDK to offer additional features such as turn-by-turn navigation and push notifications. Its licensing costs are based on the coverage area.

4.1.6 Mist Systems

Mist Systems is a company that was recently acquired by Juniper Networks, a multinational corporation that supplies networking products. Much like Cisco or Aruba, Mist Systems provides a wireless infrastructure that includes location services. Its APs include a 16 element directional BLE antenna array that, combined with machine learning, provide a location accuracy of 1 to 3 meters. The vendor provides a mobile SDK that allows smartphones to receive signals from the access points and forward the information to the engine to obtain real-time location. Its engine is limited to the cloud and offers an open API for integration.

4.1.7 Ubisense

Ubisense is an experienced vendor in the RTLS market, especially in the manufacturing industry. It offers a UWB solution that uses both TDoA and AoA methods for calculating locations, providing even higher accuracy and precision than other UWB solutions. It is capable of centimeter accuracy in 3 dimensions. It offers plenty of different choices of tags, including different sizes, combinations of battery lifetime, Ingress Protection, and robustness. Its engine can be run either on-premises or in the cloud.

4.1.8 AirFinder

AirFinder is a product division of Link Labs, a computer network technology company that specializes in IoT. The vendor offers another solution based on BLE but in this case, the anchors work as beacons and the tags work as readers. The proprietary tags are able to calculate their own position based on received signal strength which is then communicated via a special access point to the server either on-premises or the cloud. Tags communicate with these APs using BLE which in turn can communicate with the server using several different options namely Ethernet, Wi-Fi, LoRa or LTE. The vendor claims that it can offer different types of accuracy up to 1-2 meters by changing the density of reference points (anchors).

The tags use accelerometers to detect motion and adjust their update location rate accordingly. Since tags are mostly scanning and transmit much less than traditional BLE beacon tags, they are more power efficient, resulting in better battery life. This solution is also much more scalable because instead of having a single anchor receiving signals from an increasing number of tags it has a single tag receiving signals from a fairly constant number of anchors.

The vendor also provides a 'super tag' that makes the bridge between outdoor and indoor location allowing continuous tracking while moving assets between facilities. These tags rely on GPS, WiFi, and LTE to calculate positions outdoors and then use either LoRa or LTE to communicate with the server.

4.1.9 LitumIoT

LitumIoT is an experienced vendor in the RTLS market with an impressive client portfolio within the manufacturing industry. It offers an UWB solution with promised sub-meter accuracy using TDoA, ToF, and trilateration. Besides the normal asset and employee tracking, it provides additional useful features based on location such as forklift collision avoidance, or employee headcount in case of accidents. The vendor offers several different tags, including badges, with IP protection, tamper evident technology, rechargeable batteries that last up to a year, and motion sensors. Its engine is limited to on-premises.

Work Developed

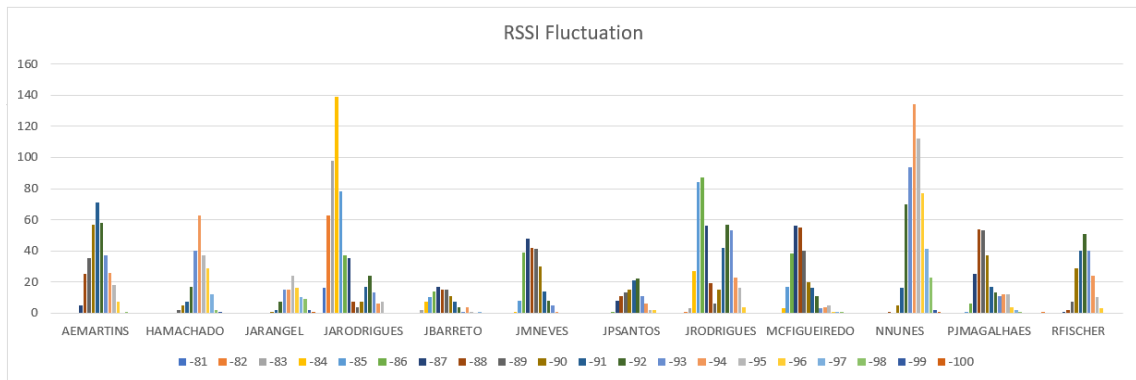


Figure 4.1: Example of RSSI fluctuation

4.2 Developed RTLS Solution

The second part of the work consisted in developing a proof of concept RTLS solution to be tested at Critical Manufacturing's facilities. The solution uses BLE with RSSI and is composed of three different applications: a Radar application; a Server application and a Manager application.

The solution architecture is described in section 3.2.2 and figure 3.1.

4.2.1 Radar

The Radar application is meant to be run on the company's laptop computers (among employees who volunteer) effectively turning them into anchors of the RTLS. The application was developed using .NET framework and WinForms to be run under Windows 10 and consists of a lightweight application that scans for nearby BLE tags and sends the respective RSSI to the server (which is responsible for computing distances and positions).

Radars communicate with the server using web sockets. On establishing a connection with the server, the server sends to the radar the list of tags to be tracked (whitelist). Each radar is actively scanning for BLE Advertisement Packets and on each packet received it parses the content looking for possible tag identifiers, tries matching any identifier found against the tags' whitelist, and if a match is found, sends the tag identifier together with the respective RSSI to the server.

The tag's identifier can be either the device's MAC address or one of the many identifiers that can exist in the payload including the ones that are part of iBeacon or Eddystone protocols. This flexibility greatly increases the number of devices that can be tracked including iOS or Android devices that use MAC randomization.

To minimize RSSI inconsistency, a sampling interval is used, i.e., instead of communicating to the server an RSSI value for each packet received, the radar aggregates the RSSI values within an interval and communicates the average value. This means that the maximum update rate of a tag is limited by the sampling interval value.

Figure 4.1 illustrates how RSSI values fluctuate for a static tag and several different radars.

Work Developed

The application was developed to be the least intrusive for the user and at the same time robust. By default, the application runs on Windows startup and minimizes to the system tray. It also allows disabling the scanner if the laptop is not connected to the network through ethernet cable (to ensure that the laptop is at its usual, known position and doesn't provide erroneous values to the server), and allows getting automatic updates. Radars try to reconnect with the server every time the connection is lost.

The application has two additional features. The first feature is the *Scan devices* that allows listing all nearby BLE devices, which is useful when looking for the identifier of a tag to add to the system. The other feature is *Calibration* which allows scanning for a specific tag and obtain the average RSSI.

4.2.2 Server

The Server application works as the engine of the RTLS and is developed using .NET Core 2.2, allowing it to run under Windows, Linux, or macOS operating systems. Its main purpose is to compute tags' locations.

The server uses two data collections to store all system information, one for radars and the other for tags that are stored persistently as JSON files. Both collections use *ConcurrentDictionary* where the keys are the radar and tag unique identifiers and the values are the respective *Radar* and *Tag* objects. Each radar object contains information regarding its ID, its position coordinates and specific correction value for calculating distances based on RSSI. The tag object contains some persistent information, such as its ID and Tx value, but the most part is temporary such as its current position and the RSSI information sent by each radar. Each tag contains a *ConcurrentDictionary* for storing the location data sent by each radar, where the key is the respective radar object and the value is an object that contains the received RSSI values with timestamps associated. When the server receives RSSI information from a given radar, for a given tag, it updates the location data in the respective tag object, for the respective radar. If the respective entry in the tag's dictionary for the given radar doesn't exist, it is created.

Periodically, the server calculates positions for all tags based only on recent information from radars. For every tag in the database, the server selects the respective location information whose timestamps fit the update interval and computes distances based on the RSSIs. If the tag has location information from at least 3 radars, the radar positions and respective calculated distances are used with the multilateration algorithm for calculating the tag position. If the tag doesn't have enough information to calculate the position 3 consecutive times, it is marked as inactive until a position is calculated again.

The selected multilateration algorithm uses a minimum mean square error estimator to obtain the optimal position. If a position was already obtained for the tag to be calculated, its last position is used as the first estimator. Otherwise, a random position between all the radars used for the calculation is generated. It then computes all the Euclidean distances between the estimator and

Work Developed

```
1 Vector<double> calcRanges = DenseVector.Build.Dense(n);
2 for (int i = 0; i < nTries; i++)
3 {
4     double error = Double.MaxValue;
5
6     Vector<double> estimatorNew = DenseVector.OfVector(estimatorIn);
7     Vector<double> estimator = estimatorNew;
8
9     /* Remove random anchor */
10    var outlier = removeOutlier ? (nTries == anchorsIn.RowCount ? i : random.
        Next(n + 1)) : -1;
11    var anchors = removeOutlier ? anchorsIn.RemoveRow(outlier) : anchorsIn;
12    var realRanges = removeOutlier ? VectorRemoveIndex(rangesIn, outlier) :
        rangesIn;
13
14    Stopwatch stopwatch = new Stopwatch();
15    stopwatch.Start();
16
17    while (true)
18    {
19        Vector<double> totalDelta = DenseVector.Build.Dense(dim);
20        for (int j = 0; j < n; j++)
21        {
22            calcRanges[j] = Distance.Euclidean(anchors.Row(j), estimatorNew);
23            if (calcRanges[j] > 0)
24            {
25                var delta = (realRanges[j] - calcRanges[j]) / calcRanges[j];
26                totalDelta += delta * (estimatorNew - anchors.Row(j));
27            }
28            else
29            {
30                totalDelta += realRanges[j] * estimatorNew;
31            }
32        }
33
34        double errorNew = CalcError(realRanges, calcRanges);
35        if (error > tolerance && errorNew < error - tolerance && stopwatch.
            ElapsedMilliseconds < timeThreshold)
36        {
37            error = errorNew;
38            estimator = estimatorNew;
39            estimatorNew = estimatorNew + totalDelta / n;
40            result.Iterations[i]++;
41        }
42        else
43        {
44            result.Estimators.SetRow(i, estimator);
45            result.Errors[i] = error;
46            result.Outliers[i] = outlier;
47            break;
48        }
49    }
50 }
```

Listing 4.1: Multilateration algorithm

Work Developed

each of the radars and sums the square errors between the computed distances ('calcRange') in this step and the obtained distances from the RSSI ('realRange').

A new estimator is then obtained by calculating deltas for each position coordinate (x and y is this case) and summing them to the previous estimator. This iterative process is repeated until the sum of the square errors keeps decreasing or a timeout is reached. The deltas are calculated by multiplying, for each radar, the relative difference between 'realRange' and 'calcRange' with the difference between the respective coordinates of the estimator and the radar and summing the values of each radar and dividing them by the total number of radars (see expressions 4.1 and 4.2).

$$\delta_x = \left[\sum_{j=1}^n \frac{realRange_j - calcRanges_j}{calcRanges_j} \times (Estimator.x - Anchor_{j,x}) \right] / n \quad (4.1)$$

$$\delta_y = \left[\sum_{j=1}^n \frac{realRange_j - calcRange_j}{calcRange_j} \times (Estimator.y - Anchor_{j,y}) \right] / n \quad (4.2)$$

This algorithm has revealed to converge quickly with very good results. Because it can run really fast, an additional outlier removing step was introduced where the algorithm is run several times, and each turn a different radar is removed. The solution with the smallest error is compared with the solution where no radar was removed and if the difference is big enough, the radar is considered an outlier. This step can be used multiple times as long as enough radars remain to calculate the position, and a percentage of the initial number of radars is maintained.

Listing 4.1 shows the most relevant portion of the code of the multilateration algorithm used to compute tags positions.

A simple API was created for testing the integration with the MES using MQTT, a lightweight standard publish-subscribe-based messaging protocol. Every time the server calculates positions it publishes the positions of the updated tags in a JSON formatted message. Any application that subscribes to the specific topic will receive the message containing the updated positions. For testing purposes the Eclipse Mosquitto application was used as the MQTT broker.

4.2.3 Manager

The Manager application provides a graphical user interface for adding, editing and removing radars and tags to the system and provides a map for visualization of the tags' current positions. It was developed using .NET framework and WPF.

The location feature of the manager allows visualizing on a map the current position of each tag as well as the position and activity status of each radar. The radar status is indicated by different colors where red means the radar is turned off, green means it is on, and yellow means it is on but with the scanner turned off. A grey tag also indicates that its position hasn't been updated recently due to the lack of activity (instead of the regular color blue).

It provided a useful tool to evaluate results from the multilateration algorithm from the early stages of development and to compare results in the integration with the MES.

Work Developed

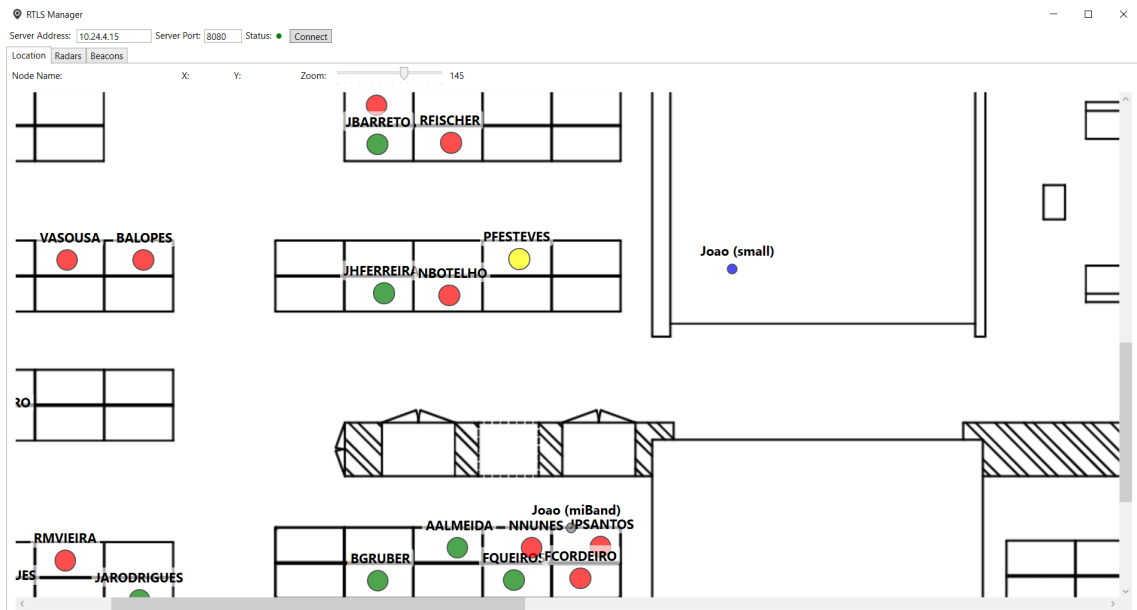


Figure 4.2: Manager's location view

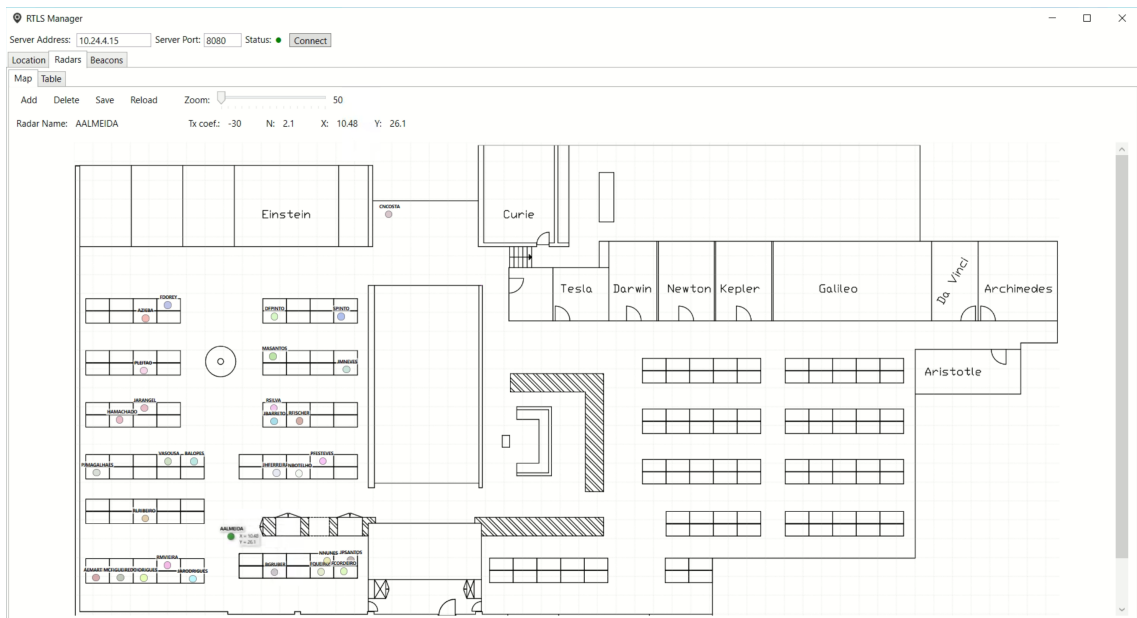
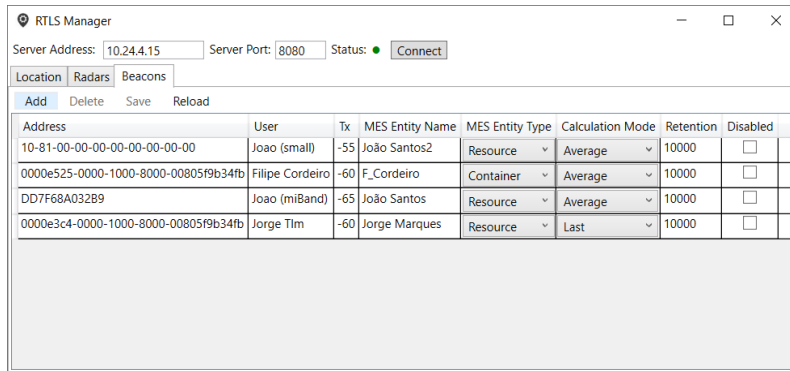


Figure 4.3: Manager's radar view

Work Developed



Address	User	Tx	MES Entity Name	MES Entity Type	Calculation Mode	Retention	Disabled
10-81-00-00-00-00-00-00-00	Joao (small)	-55	João Santos2	Resource	Average	10000	<input type="checkbox"/>
0000e525-0000-1000-8000-00805f9b34fb	Filipe Cordeiro	-60	F_Cordeiro	Container	Average	10000	<input type="checkbox"/>
DD7F68A032B9	Joao (miBand)	-65	João Santos	Resource	Average	10000	<input type="checkbox"/>
0000e3c4-0000-1000-8000-00805f9b34fb	Jorge Tlm	-60	Jorge Marques	Resource	Last	10000	<input type="checkbox"/>

Figure 4.4: Manager’s beacon view

Figure 4.2 shows an example of the Manager’s location GUI.

The radar editor also provides a map for easily adjusting a radar’s position by simply dragging the radar’s respective thumb.

Figures 4.3 and 4.4 respectively show examples of the Manager’s radar editor and tag (beacon) editor.

4.3 MES Integration

The last part of the work developed was to integrate Real-time location systems with the Critical Manufacturing MES. This consisted in developing a Connect IoT Task that could interpret messages from multiple different RTLSs and that allowed expanding to new systems in the future. The purpose was to be able to use the MES 3D model engine FabLive to locate people and objects.

4.3.1 Connect IoT Task

The ConnectIoT module is developed using Node.js and tasks are written in Typescript. Since the Connect IoT module runs separately from the MES it requires external calls to it, using, in this case, specially designed libraries called LBOs.

The developed task has the responsibility to receive data from an RTLS and update information in the MES accordingly. The task accepts two different means of inputs: either a single string that requires parsing a specific message format and can contain multiple tag information, or individually each of the several values required to update a single tag. For the purposes of testing with the developed RTLS solution, a JSON format message with information for multiple tags was used.

Figure 4.5 shows a workflow using the RTLS where the several inputs can be seen. For the specific test case, the controller is receiving messages from an MQTT driver.

The MES has the ability to store every change to its entities in its database, which in the case of position information can be used as tracking. To avoid filling up the MES database quickly, an option to not save changes can be used, by only triggering location changes in the 3D models

Work Developed

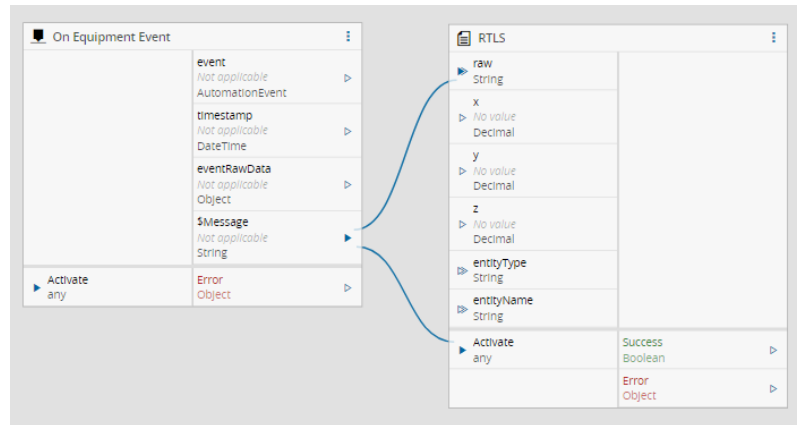


Figure 4.5: Workflow using the RTLS Task

The screenshot shows the RTLS Settings dialog box. It is divided into three sections: General, Settings, and Transformations. The General section includes fields for Name (RTLS), Description, and Color. The Settings section includes fields for Entity type (Resource), Raw format (JSON_X_Y_TYPE_NAME), Tracking (checked), Tracking Interval (1000), and Location Interval (500). The Transformations section includes fields for Scale X (0.003), Scale Y (-0.003), Scale Z (0.003), Offset X (0), Offset Y (-31.4), and Offset Z (0). The dialog has Cancel and OK buttons at the bottom right.

Figure 4.6: RTLS Task Settings

Work Developed

of fabLive. The task allows disabling the tracking functionality altogether, or setting a minimum waiting interval between consecutive saves, between which, the changes are reflected only in the fabLive. An interval was also added for restricting updates in fabLive.

Some types of entities of the MES, namely *Resources*, *Materials* and *Containers*, already have location coordinates and these are the ones that can be updated by the task and reflect those changes in fabLive. It is required, for each type of entity to explicitly set the option to trigger location changes in real-time in fabLive.

All the task parameters that can be set are modified in its settings menu, namely the expected message format in the raw input, the option to track changes, the intervals for restricting updates and all the transformations required to convert from the RTLS location coordinates to the MES coordinates. Figure 4.6 shows the settings menu for the developed RTLS Task.

4.3.2 FabLive

Fablive is a powerful 3D model visualizer included in the Critical Manufacturing MES that allows users to see in an easy, graphical, and attractive way, and in real-time, what is happening on the factory floor. It has a complete editor that also allows importing 3D objects from other sources. FabLive allows binding any object in the model to any entity of the MES and reflect the changes made to that entity. This isn't restricted to location, as other information can trigger changes in color, for example, or simply be displayed in text.

This part of the work also involved developing a complete 3D model of Critical Manufacturing headquarters with accurate measurements to mimic the RTLS test scenario.

Figure 4.7 shows the referred model developed for the tests with the RTLS integration working successfully.

4.4 Results

Regarding the RTLS market solution survey, most of the information obtained was from research on the internet and only a small part was obtained from directly contacting the vendors. Most of them did not reply to the inquiries, and the ones that did, did not fully disclose prices. Also, none was able to provide a test solution. Nonetheless, information for a large number of commercial solutions was gathered that allowed a detailed analysis of several different existing options.

The developed RTLS solution showed the limitations of localization using received signal strength. Because the RSSI is very inconsistent, especially at higher distances, this results in errors while estimating distances, and consequently in lower accuracy and precision. The use of sampling in radars allowed reducing some of the RSSI fluctuation but also limited the system responsiveness.

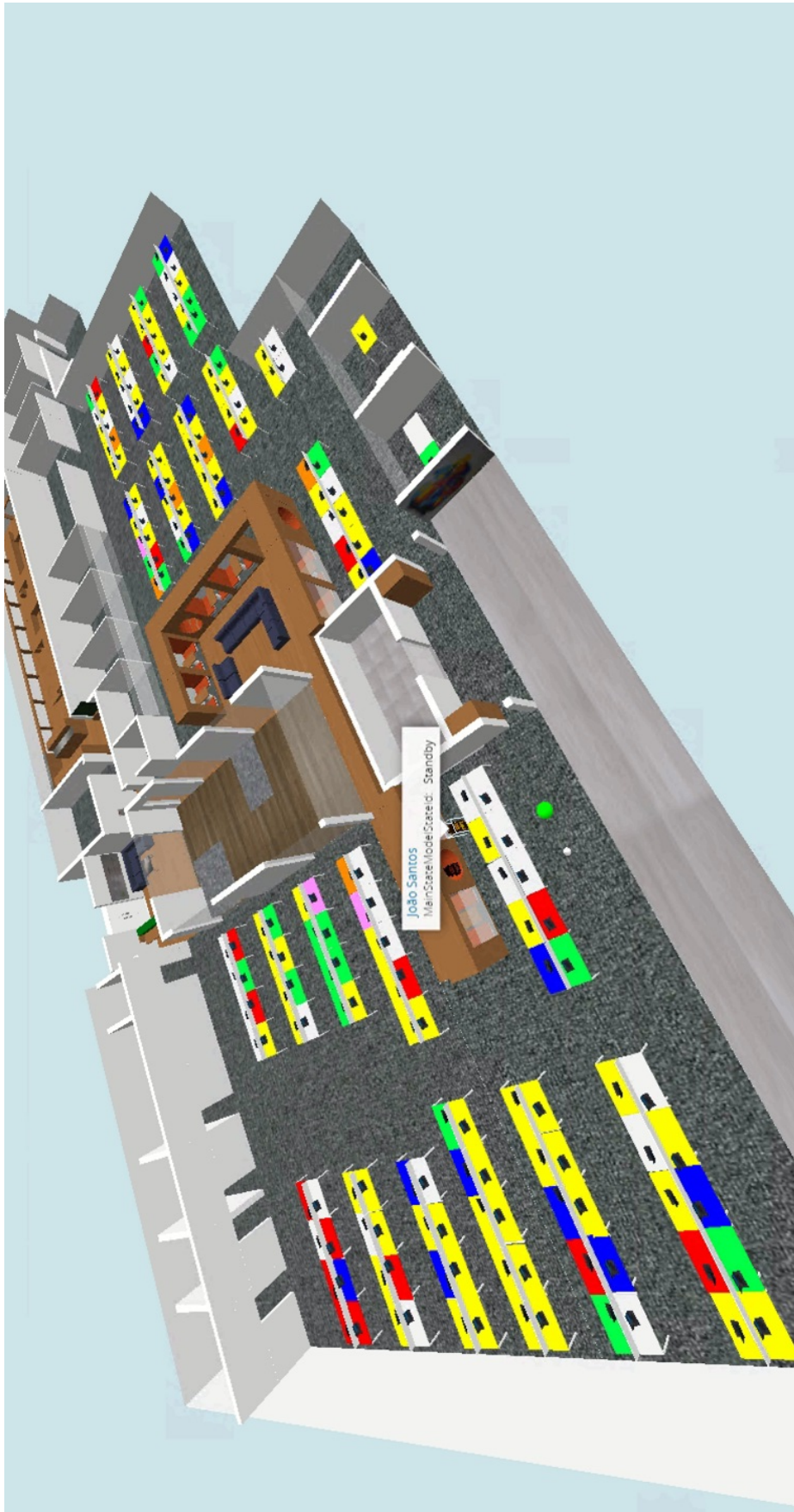


Figure 4.7: FabLive - 3D model of Critical Manufacturing Headquarters

Work Developed

The measured accuracy was on par with what is expected for this type of solution at around 5 m. This was obtained by recording several calculated positions for a static tag in 3 distinct positions. Using an outlier detection method allowed a small improvement in accuracy.

The fact that the solution used several different hardware, including different laptops with different Bluetooth adapters, also influenced the results and required special calibration.

The results of the MES integration are exactly what was expected. The task was able to translate information from the RTLS to the MES and objects could be followed using fabLive. The update intervals could be changed in the task settings and the changes were recognized in fabLive.

Chapter 5

Conclusions and Future Work

5.1 Objectives' achievement

A comprehensive analysis of the current RTLS market solutions was performed, and many of the most relevant ones were presented in detail in this document. Unfortunately not many prices were obtained to make a clear comparison between costs of different solutions and/or technologies due to the lack of responses from inquiries to vendors. Neither was possible to acquire trial versions of any of the systems because of the costs involved and also the lack of responses. To get around this issue, an own RTLS solution was developed to allow the second objective to be accomplished, which significantly altered the initially selected approach.

The developed RTLS is based on Bluetooth Low Energy and uses laptops as anchors and different types of beacons (including both standard BLE beacons as well as ones using the iBeacon and Eddystone protocols) and smartphones as tags, and was able to obtain reasonable accurate positions (considering the technology).

The developed RTLS Task that allows integrating an RTLS system into the MES had the expected results, allowing updating virtual objects in a 3D model based on real-world objects' positions, effectively enhancing the digital twin of the Critical Manufacturing MES. Since no additional sensor information was available, no development was made in this area.

5.2 Future Work

In view of both objectives of the proposal, it seems that RTLS integration has more room to continue development. The developed task is very flexible as it allows adding different message formats but it still involves work and some knowledge of its code. Extending compatibility to existing systems in the market would be the most practical route in terms of future work.

Regarding the survey, considering that technology is evolving at a galloping pace, there are always new solutions appearing and new approaches to discover, and taking into account that an

Conclusions and Future Work

ever-increasing demand for RTLS is expected, continuous research will be needed to be kept up to date. BLE 5.1, for example, is one of the technologies to keep an eye on.

The proof of concept developed may also be interesting to explore from an academic point of view. Testing different approaches with different types of methodologies, or algorithms to improve accuracy (artificial intelligence) are some of the aspects that can be worked on.

References

- [BCLN05] Gaetano Borriello, Matthew Chalmers, Anthony LaMarca, and Paddy Nixon. Delivering real-world ubiquitous location systems. *Communications of the ACM*, 48(3):36–41, 2005.
- [CGK⁺18] G. Cwikla, C. Grabowik, K. Kalinowski, I. Paprocka, and W. Banas. The initial considerations and tests on the use of real time locating system in manufacturing processes improvement. *IOP Conference Series: Materials Science and Engineering*, 400, 2018.
- [CLCL16] Larissa P.M. Cruz, Luiz Landau, Gerson Gomes Cunha, and Maria Celia S. Lopes. Identification of assets and statement of wireless technologies to support real time location systems: case rnest – abreu e lima refinery. *International Journal of Service and Computing Oriented Manufacturing*, pages 103–114, June 2016.
- [DSCD15] Davide Dardari Senior, Pau Closas, and Petar M. Djurić. Indoor tracking: Theory, methods, and technologies. *IEEE Transactions on Vehicular Technology*, pages 1263–1278, February 2015.
- [fS14] International Organization for Standardization. ISO/IEC 24730-1:2014: Information technology – Real-time locating systems (RTLS) – Part 1: Application programming interface (API), 2014.
- [Gaf08] Brian Gaffney. Considerations and challenges in real time locating systems design. *DecaWave white paper, Dublin*, 2008.
- [HRFCFLE17] D. L. Hernandez-Rojas, T. M. Fernandez-Carames, P. Fraga-Lamas, and C. J. Escudero. Design and practical evaluation of a family of lightweight protocols for heterogeneous sensing through BLE beacons in IoT telemetry applications. *Sensors (Basel)*, 18(1), 2017.
- [KBB12] Maged Kamel Boulos and Geoff Berry. Real-time locating systems (RTLS) in healthcare: A condensed primer. *International journal of health geographics*, 11:25, June 2012.
- [KPO15] Dmitrii Kirov, Roberto Passerone, and A.A. Ozhiganov. A methodology for design space exploration of real-time location systems. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 15(4):551–567, June 2015.
- [KY10] Hakan Koyuncu and Shuang-Hua Yang. A survey of indoor positioning and object locating systems. *International Journal of Computer Science and Network Security (IJCSNS)*, 10, January 2010.

REFERENCES

- [LDBL07] Hui Liu, Houshang Darabi, Pat Banerjee, and Jing Liu. Survey of wireless indoor positioning techniques and systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37:1067 – 1080, December 2007.
- [LLC⁺09] WJ Lee, W Liu, Peter HJ Chong, Bertrand LW Tay, and WY Leong. Design of applications on ultra-wideband real-time locating system. In *Advanced Intelligent Mechatronics, 2009. AIM 2009. IEEE/ASME International Conference on*, pages 1359–1364. IEEE, 2009.
- [Man] Critical Manufacturing. Critical manufacturing mes. Available at <http://www.criticalmanufacturing.com/en/critical-manufacturing-mes/overview>.
- [MHN⁺18] Ardiansyah Musa, Hyojeong Han, Gde Dharma Nugraha, Deokjai Choi, Seungho Seo, and Juseok Kim. A design of indoor rtls by use of the uwb-wsn based two reference points. In *2018 2nd International Conference on Applied Electromagnetic Technology (AEMT)*, pages 29–33. IEEE, 2018.
- [RLJ⁺15] Mohamed Er Rida, Fuqiang Liu, Yassine Jadi, Amgad Ali Abdullah Algawhari, and Ahmed Askourih. Indoor location position based on bluetooth signal strength. In *Information Science and Control Engineering (ICISCE), 2015 2nd International Conference on*, pages 769–773. IEEE, 2015.
- [Rou17] Margaret Rouse. What is manufacturing execution system (mes)? Available at <https://searcherp.techtarget.com/definition/manufacturing-execution-system-MES>, November 2017.
- [RPE05] Miguel Rodriguez, Juan P Pece, and Carlos J Escudero. In-building location using bluetooth. In *International Workshop on Wireless Ad-hoc Networks*, 2005.
- [TGLP08] Ricardo Tesoriero, José A Gallud, Manuel Lozano, and Victor M Ruiz Penichet. Using active and passive rfid technology to support indoor location-aware systems. *IEEE Transactions on Consumer Electronics*, 54(2):578–583, 2008.
- [ZYC⁺16] Daqiang Zhang, Laurence Tianruo Yang, Min Chen, Shengjie Zhao, Minyi Guo, and Yin Zhang. Real-time locating systems using active rfid for internet of things. *IEEE Systems Journal*, 10(3):1226–1235, 2016.

Appendix A

Commercial Solutions Survey

A.1 Comparison of RTLS Solutions

Table A.1: Comparison of RTLS Solutions

Solution	Engine	Technologies	UI	API	Tags	Notes
AirFinder	on-premises or cloud	BLE	Web Application	RESTful and data streaming modes	10 different types of tags; allows sensors to be added to tags (partners?); provisioning and updating can be performed directly on tags via smartphone	Location calculations are done in each tag and sent to the "engine" (just proximity or trilateration??)
AiRISTA Flow	on-premises or cloud	Wi-Fi, BLE, RFID	Web-browser (includes dashboard and reporting capabilities)	Open API	Air-T5: optional telemetry and sensor expansion, rugged design (water resistant) + other tags	-
Aruba Networks (HPE)	cloud-only (Meridian)	BLE	mobile and Web SDKs	REST + mobile SDK	3-4 years battery life (non-replaceable?)	APs deployed every 15m to ensure coverage. Future 3rd party BLE tags support
BlueUp	BlueBeacon Cloud	BLE (iBeacon, Eddystone, Quappa)	BlueBeacon Manager (iOS/Android); BlueBeacon Cloud (web)	HTTP REST	Multiple solutions (tags, badges/cards, usb, etc). Some tags include waterproof, accelerometer for battery savings, or sensors	BlueBeacon Locate: 3 alternative modules, with different levels of performance, accuracy and cost: AccuRTLS, ZoneRTLS, BeaconRTLS
Bluvision	cloud-only (Bluzone)	BLE	Web (AP management + tracking)	REST	several form factors, IP67, 3 to 9-year battery life	sub-two meter accuracy, does not require wired hardware (AC-powered BLE/WiFi gateways)
CenTrak	?	Multi-mode technology: BLE, Wi-Fi, IR, Low Frequency RF	?	?	?	-

Commercial Solutions Survey

Table A.1: Comparison of RTLS Solutions

Solution	Engine	Technologies	UI	API	Tags	Notes
Cisco	on-premises (Aironet), cloud (Meraki)	Wi-Fi, BLE		REST	can be integrated with motion, temperature, humidity and voltage sensors	-
Eliko	on-premises KIO RTLS Server	UWB	KIO RTLS Manager for easy setup and visualisation	Open API for seamless integration	Anchors supporting PoE or Wi-Fi	-
LitumIoT	on-premises	UWB	Asset Management software	?	KIWI/COCO TAGS: IP55/IP67 protection, tamper evident, embedded motion sensors, Battery life 0.5y/1y (rechargeable); BADGE TAG	-
Mist Systems	cloud-only	APs with 16 element directional BLE antenna array + machine learning in the cloud	web interface for engine; mobile SDK with open APIs	?	3rd party only?, verified compatibility with Bluvision, kontakt.io and Radius Networks	Can work both as listener (assets) and as transmitter (personnel) (simultaneous), Virtual Beacons (BLE beaconing functionality in the AP)
Pole Star	on each device*, cloud platform for management	BLE	mobile devices - NAO SDK; web - NAO Cloud	?	optional for asset tracking (3rd party?): using "BLE listeners" or crowd-sourced approach based on staff smartphones and NAO Track+; data is processed on NAO Cloud	* the location algorithm resides on the mobile device (NAO SDK);
Pozix	cloud or on premise (with some limitations)	UWB	Companion software (local or cloud): Real-time visualization, Deployment, Device and user management	MQTT (local or cloud streams) and open API	deep sleep mode (accelerometer movement); IP20/IP67; Battery life-time of +3 years	Maximum of 1000 tags; Total maximum of 1000 positions/s; Up to 8 anchors daisy chained PoE+; Additional modules for the companion software
Quuppa	"on-premises or cloud, Quuppa Positioning Engine (QPE)"	BLE	Java-based software: Quuppa Site Planner; Quuppa Data Player; Quuppa System Simulator	JSON/REST-push/pull API for the QPE	IP67, 3-axis accelerometer, thermometer, programmable button, LED	can integrate any BLE device (Quuppa Tag Firmware Libraries / emulation)

Commercial Solutions Survey

Table A.1: Comparison of RTLS Solutions

Solution	Engine	Technologies	UI	API	Tags	Notes
Redpoint	server hosted locally or in the cloud	proprietary UWB/BLE hybrid	Redpoint Sitewise (web) and Redpoint Site Manager (iOS)	REST + Pubsub + Tag API	Asset tag: ruggedized, water and dust-proof, 3-year battery life with an easy-to-replace 1/2AA cell. includes a call button, motion sensing capability	Class I - up to 10 anchors; Class II - up to 50 anchors; Class III - up to 1,000 anchors; Anchors range 30-50 m -
Siemens SIMATIC RTLS	?	UWB	?	?	?	-
Sinfosy Sintra	Cloud?	BLE	Android app to register and find assets	?	BLE iBeacons 1 Hz	Receivers are standard mobile phones (Android app available in the playstore, app sends beacon data to a server)
Tracktio uRTLS	Stores information, in a local server or the cloud	UWB, RFID	Web	REST + Pub-sub	optional: ruggedized, waterproof, antitamper, temperature, accelerometer, etc (also for sensors)	-
Ubisense	on-premises or remotely in a centralized data center	UWB	?	?	different combinations of battery lifetime, Ingress Protection, robustness and even location modalities (including GPS)	-
Zebra Technologies	on-premises, private cloud, AWS-hosted public cloud instance or managed service	Wi-Fi, UWB, GPS, BLE, passive and active RFID	?	Full REST API + XML event publisher	BLE: 5 different beacons available; WhereNet: IP67, 5y battery life; UWB: tag/badge, IP67, 7y battery life, blink rate from 0.01 to 200 Hz	BLE solution is proximity only, doesn't provide readers, just SDK; WhereNet solution uses ISO/IEC 24730-2 (2.4 GHz RF)