Universidade de Aveiro
2022

SUSANA CATARINA MOREIRA DIAS

Digital Twins em IIoT

Digital Twins in IIoT

Universidade de Aveiro
2022

SUSANA CATARINA MOREIRA DIAS

Digital Twins em IIoT

Digital Twins in IIoT

“*The greatest challenge to any thinker is stating the problem in a way that will allow a solution*”

— Bertrand Russell

Universidade de Aveiro
2022

SUSANA CATARINA
MOREIRA DIAS

Digital Twins em IIoT

Digital Twins in IIoT

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor Diogo Gomes, Professor auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Doutor João Paulo Barraca, Professor auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri / the jury

presidente / president	Professor Doutor Arnaldo Silva Rodrigues de Oliveira Professor Auxiliar da Universidade de Aveiro
vogais / examiners committee	Doutor Diogo José Domingues Regateiro Engenheiro de Software na Atlar Innovation
	Professor Doutor Diogo Nuno Pereira Gomes Professor Auxiliar da Universidade de Aveiro

agradecimentos / acknowledgements

Agradeço aos meus pais e irmãs pelo imenso apoio ao longo do meu percurso e por nunca duvidarem de mim. Agradeço também à Ana Santiago e ao Tomé, pelo apoio, paciência e incentivo. Um agradecimento especial aos professores Diogo Gomes e João Paulo Barraca, por toda a orientação e disponibilidade que permitiram a realização e apresentação deste trabalho.

Palavras Chave

Industria 4.0, Digital Twin, Processamento de Dados, Armazenamento de Dados, IIoT

Resumo

A indústria está em constante evolução e, como resultado dessa evolução, o número de produtos fabricados aumentou exponencialmente, levando a uma grande competitividade entre as companhias. Esta competitividade, gerou a necessidade de tornar o processo de fabrico mais eficiente e sustentável, para tal é necessário capturar os dados provenientes das máquinas e estudá-los, de forma a identificar possíveis melhorias nos produtos e no processo de fabrico. Estes requisitos podem ser alcançados através da implementação de um Digital Twin, uma vez que este permite espelhar o dispositivo físico e através disso, permite que os engenheiros da fábrica usem e analisem os recursos, permitindo que entendam, monitorem, calculem e testem comportamentos de dispositivos físicos num sistema virtual, de forma mais eficiente e económica para a indústria. Esta dissertação pretende apresentar uma infraestrutura que suporte uma implementação rápida e eficiente de uma solução Digital Twin, com foco no processo de aquisição e armazenamento de dados. Tendo como objetivo principal criar um sistema eficiente capaz de suportar um Digital Twin, com elevada capacidade de processamento e armazenamento de dados.

Keywords Industry 4.0, Digital Twin, Data processing, Data Storage, IIoT

Abstract

Industry is constantly evolving and, as a result of this evolution, the number of products manufactured has increased exponentially, leading to a great competitiveness between companies. This competitiveness, created the need to make the manufacturing process more efficient and sustainable, for this it is necessary to capture the data from the machines and study them, in order to identify possible improvements in products and the manufacturing process. These requirements can be achieved through the implementation of a Digital Twin, since it allows mirroring the physical device and through that, allows the factory engineers to use and analyse the resources, allowing them to understand, monitor, calculate and test the behaviours of the physical devices in a virtual system, in a more efficient and economic way for the industry. This dissertation aims to present an infrastructure that supports a fast and efficient implementation of a Digital Twin solution, focusing on the data acquisition and storage process. The main goal is to create an efficient system capable of supporting a Digital Twin, with high processing capacity and data storage.

Contents

List of Figures

List of Tables

Listings

Glossary

AI	Artificial Intelligence
DL	Deep Learning
DT	Digital Twin
ECNs	Edge Computing Nodes
IaaS	Infrastructure as a Service
IT	Information Technology
IoT	Internet of Things
IIoT	Industrial Internet of Things
ML	Machine Learning
M2M	Machine to Machine
OT	Operational Technology
PaaS	Platform as a Service
RAMI4.0	Reference Architectural Model Industrie 4.0
SaaS	Software as a Service

CHAPTER 1

Introduction

"Learn as if you will live forever, live like you will die tomorrow."
Mahatma Gandhi

Industry 4.0 has enabled new scenarios in the Industrial World, where humans, machines, and products cooperate with each other in real-time, allowing decentralised decision making. One of the main advantages of this industry is the need to meet consumer demands, allowing a faster, less expensive, and more flexible response.

Industry is, therefore, increasingly-betting on Internet of Things (IoT) solutions, solutions that are based on the interconnection of multiple devices that allow them to receive and exchange a set of precise information in real time information between them, and that can also be integrated with wireless networks and sensors to monitor shop floors and process events in real-time. [1]

Each device has a large amount of past and present information about the physical properties of the object. One of the main benefits of these devices is the capability of using this information to understand management processes, leading to an increase in production and Preventive Maintenance on the factory floor [2] and, consequently, a reduction in the time and resources required to rectify faults. From these points of view, it is possible to understand that we are facing an industrial revolution with the digitisation of factories, which will bring a great improvement in efficiency and energy management. By digitising all machinery and equipment, it will be possible to have a wider vision of automation and efficiency, where Artificial Intelligence (AI) can help in the optimisation of processes and procedures. For this to be possible, a Cyber-Physical Platform should be developed on which devices expose their properties and services so that through Information Technology (IT) we can benefit from them and use their knowledge to optimise production.

One of the means to achieve this is through a Digital Twin, a revolutionary concept of *Industry 4.0*, which allows the connection between the physical and virtual worlds, unifying them and allowing the abstraction of real-world devices with all their capabilities and aspects, including their digital representation. By this, achieving a closed loop between production and design. Therefore, this technology is an efficient way to connect the virtual and real worlds, with the ability to intelligently manage system optimisation as it keeps both worlds in sync and can be applied in industrial and consumer-centric IoT scenarios.

Digital Twin, then emerged in the industry and with it brought a multitude of advantages, such as, reduced maintenance time, quick prototyping of products, but most importantly, reduced number of errors, uncertainties and inefficiency of the systems [3]. Proposing to solve several of the problems of industries, Digital Twin (DT) came to evolve in the industry, such as technologies such as Artificial Intelligence, Cloud Computing, among others.

1.1 Motivation

The industry is in constant evolution and, as a result, there is an exponential and continuous increase in products, leading to greater competition between companies. Despite this competitiveness, however, the product manufacturing process is limited by both economic factors and the risk of conducting production control tests, including checking the expected quality of the final product and verifying all the necessary requirements for placing it on the market. These constraints have led to the need to create systems that enable more efficient and sustainable manufacturing of products.

The focus of this dissertation arises from this need, to create and integrate a Digital Twin solution from scratch within an existing Industrial Internet of Things (IIoT) platform that allows factory engineers to quickly integrate Digital Twins into the various devices and/or machines in the factory that produce large and diverse amounts of data where they can expose their properties and services. This solution makes it easier for engineers to use and analyse these resources by allowing them to monitor, understand, calculate, and simulate the behaviour of a physical system in a more efficient and cost-effective way for industry.

1.2 Work organisation

The first step in developing this solution was the study of the dataset, which consisted of understanding the data provided by the devices, its format and what it means. Secondarily, the necessary requirements were examined in order to implement a functional system.

The third step in the development of this system was the identification of the necessary components for the creation of a Digital Twin Platform, starting with the evaluation of several databases. This research was due to the high production of different types of data, seeking to find the most suitable and efficient database for the storage of the various types of data. Next, the technology used to implement the Digital Twin software standard was chosen.

1.3 Document Organisation

This document is organised as follows:

- Chapter 2 - **State of the Art** presents the state of the art of current solutions relevant to this work;
- Chapter 3 - **Proposed Solution** describes the proposed solution and the steps taken to achieve it;
- Chapter 4 - **Implementation** describes the implementation of the solution and the designed architecture;
- Chapter 5 - **Results** express the results obtained and the analysis of the main characteristics;
- Chapter 6 - **Conclusion and Future Work** exposes the conclusions and future work that is intended to be pursued.

CHAPTER 2

State of the art

"Learn as if you will live forever, live like you will die tomorrow."
Mahatma Gandhi

2.1 Industrial Internet of Things: a new concept on Industry

The Internet of Things is a computing concept based on the interconnection of physical objects(e.g.,vehicles, sensors..etc) enabling the exchange of information between them. This process can be done through the use of software, networks and sensor units. In other words, IoT is an extension of the internet that makes it possible for those same everyday physical objects, with computing and communication capabilities, to connect to the Internet. This connection enables the remote control of the physical objects and makes it possible for them to be used as service providers. These new capabilities came to great changes in the industrial world by unifying electronic and electromechanical industrial automation equipment.

The use of this technology in industry has created a new concept, shown in the figure below 2.1, *Industry 4.0*, where industrial machines and devices are connected and interact via the internet, making it possible to make more flexible, efficient and less costly decisions for the company.

This technology is less costly once it's based on cloud computing and it won't need too many people to manage and maintain the system, so the number of employees will decrease once the machines will do the work done by humans.

It's more flexible since the machines can be easily programmed, which will allow an easier way to create new products and change some existing ones. Therefore, the industry sector will be more efficient since the industry uses intelligent devices the inactivity time will be zero, because the machines don't need to rest as humans, and consequently, there will be an increase in manufacturing.

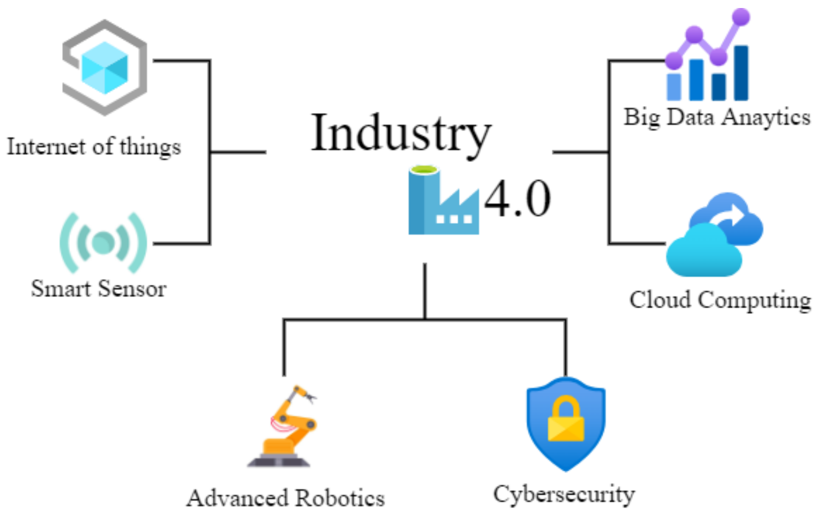

Figure 2.1: Industry 4.0

With adherence to Cyber Physical Systems, industries will become smarter, allowing the infrastructure to establish contact with suppliers and customers, thus having a more synchronised demand, providing more benefit to the company.

2.1.1 Internet of Things

In recent years it has become impossible to hear about technology and not hear about the Internet of Things, and even without using the term IoT it is easy to find something in the daily lives that represents it. An example of a “Thing” on the Internet of Things that is easily find in the daily lives is the Smartwatch, which is probably the most used device today. This one allows the counting of steps, distances, heart rate, sleep quality and many others. Through its sensors it collects data and can send it to a mobile phone or even to specific applications, allowing to keep a record of these.

According to the International Telecommunications Union, Internet of Things is “a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies”. [4]

It is incorrect to think that IoT is a single technology, since it is formed by a set of software and hardware technologies that cooperate with each other and result in IoT. [5]

Since IoT is made up of several components, there are several accepted architectures, however the architecture explored in this study is the three-tier architecture, which is demonstrated in figure 2.2. This architecture is the most basic of the existing architectures, however it is the most convenient and simple to implement [6]. It consists of the following layers:

- **Perception Layer**: consisting of sensors and devices that collect data, and make the connection between the digital world and the real world by converting analog signals into digital ones. There are several types of sensors, and these aggregate data such as temperature, pressure, distance, location, etc.

- **Network Layer**: Responsible for connecting the Perception Layer with the Application Layer. The data is transferred to the gateways through a specific communication protocol, the gateways (modems/routers) forward the data to the backend system. This routing is done in a similar way to the routing of sensors to gateways, however the protocols used are Wi-Fi, Ethernet or cellular.
- **Application Layer**:It is at this layer that the communication with the user takes place, providing specific services. An example is a smart home application, when users click an app button to turn on and set the heating temperature. This layer is responsible for providing the user with application-specific features. There are many more applications for IoT, be it smart homes, smart cities and smart health.

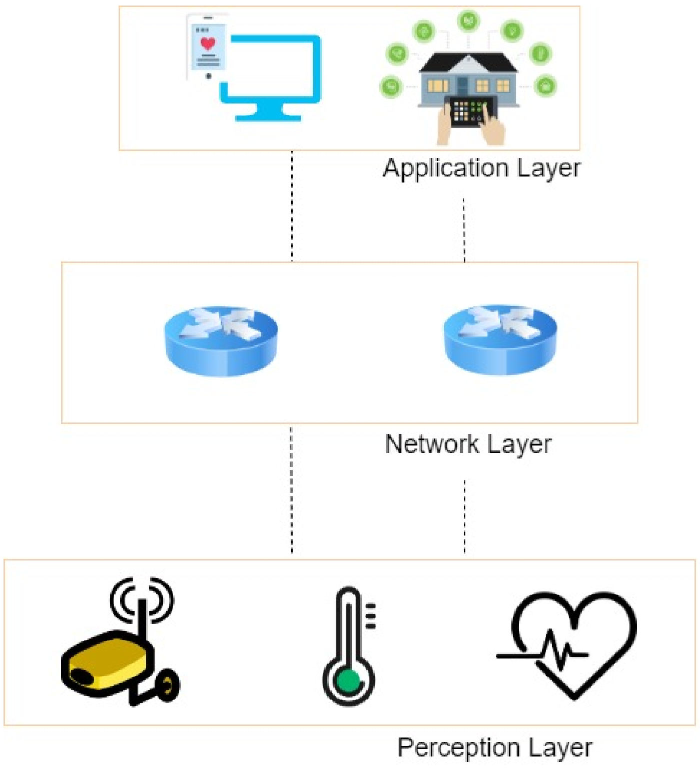

Figure 2.2: 3-Layer IoT Architecture

2.1.2 Industrial Internet of Things

The Industrial Internet of Things and Internet of Things are based on the same principle. However, IIoT is a subset of the IoT, as shown in the figure 2.3, that applies networking devices and their connection to the internet to share data in industrial processes. It addresses the areas of industrial and Machine to Machine (M2M) communication technologies using various tools, including the Internet of Things, Cloud Computing, Big Data, Artificial Intelligence and, more recently, Digital Twins. In the industrial world, people are seeing an increase demand for these types of solutions that aim to Operational Technology (OT) with the domain of IT. The principal motive for this growth was due to manufacturers seeking to increase productivity and efficiency through smart and remote management.

So its possible to comprehend that IIoT aims to connect industrial devices such as machines and control systems with information systems. As an example, ABB, a Swiss-Swedish multinational corporation, that works in the production of Robots, uses connected, low-cost sensors in order to monitor the maintenance of the robots, in this way this industry can repair parts before they break. ABB has developed a compact sensor that is attached to the frame of low voltage induction motors, where no writing is needed, thus the company gets information about motor health via smartphones or internet by a secure server.

Despite this increase, the Industrial Internet of Things still faces some major challenges, once IIoT applications need to connect a great number of devices to the internet at low cost, with limited hardware capabilities and energy resources. This needs, make low latency, low cost, energy efficiency and security desirable features. [7]

A large amount of data is collected by a variety of different machines and can then cover a variety of sectors, such as the healthcare sector, where it is possible to prevent people from dying due to hospital errors and analyse patient data for recommendations and medical reviews [8], and in the manufacturing sector, where data collected by various machines is used to create analytical solutions that lead to optimal industrial operations by monitoring the temperature and pressure of machines, monitoring the electrical network, which enables prevention of failures, and so on. [7].

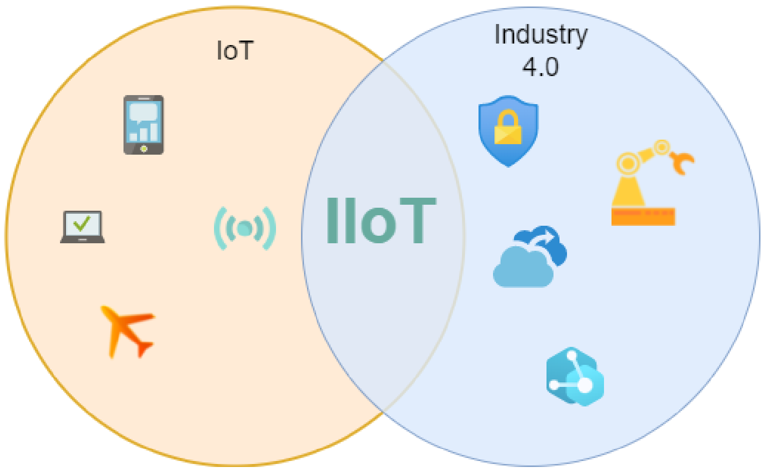

Figure 2.3: Venn Diagram IoT, IIoT and Industry 4.0

2.1.3 Manufacturing Execution System Solution

Manufacturing execution systems have emerged from the need of manufacturing companies to meet market requirements in terms of reactivity, quality, respect of standards, cost reduction and targets, allowing to deliver information that enables the optimisation of production activities from the product order to its finalisation. Through the use of current and accurate data, an MES guides, initiates, responds and reports on shop floor activities as they occur, and through this data enables decision makers to understand current shop floor settings and optimise the production process . In other words, its main objective is to ensure the efficient execution of manufacturing operations and improve production output.

Manufacturing Enterprise Solutions Association (MESA) International, is a community that aims to improve management operations in manufacturing through the implementation and application of IT and best management practices. In order to meet the needs of different manufacturing environments this association has created a MESA-11 Model, where the eleven main functions of an MES have been identified, these contribute to achieving timely performance meeting customer order schedules. Table 2.1, identifies these functions.

Functions	**Description**
Operations/Detailed Scheduling	Scheduling of activities for optimal plan performance based on finite resource capacities
Resource Allocation and Status	Guides what people, tools, machines should do and identifies what they are doing or have just done
Dispatching Production Units	Managing the flow of materials to resources in order to produce the product
Document Control	Manage and distribute product, process and order information and collect work certification statements and conditions
Product Tracking and Genealogy	Monitoring the progress of units or batches of product in order to create a complete product history
Performance Analysis	Compares the value measured on the shop floor with objectives set by the company/customers
Labour Management	Guide the use of operations personnel during the shift, according to qualifications, work patterns and business needs
Maintenance Management	To plan and execute activities in order to keep equipment and other assets working towards a goal
Process Management	Directing workflow based on planned production activities
Quality Management	Track and analyse product and process characteristics against engineering ideals
Data Collection/Acquisition	Monitoring, collecting and organising data on the process, people and machines

Table 2.1: MESA-11 Model functions

Some of the main benefits achieved through the use of MES are lower labour costs, increased customer satisfaction and better agility and time to market. According to a MESA survey, these systems have enabled a 45% reduction in manufacturing cycle time [9].

2.1.4 Cloud Computing

Cloud Computing allows users from anywhere in the world, as long as they have an internet connection, the use of services according to their needs without having to worry about where the services are hosted. Nowadays, industries require a faster response to their operations, so Cloud Computing has come to save time by distributing the load of requests across multiple clouds in multiple locations at the same time. Cloud Computing provides storage solutions and services, and these services follow a pay-as-you-go model for consumers. The services offered by Cloud Computing can be divided into 3 categories, Infrastructure as a Service(IaaS), Platform as a Service(PaaS) and Software as a Service(SaaS). [10] In the figure 2.4, are shown the services offered by Cloud Computing hierarchically.

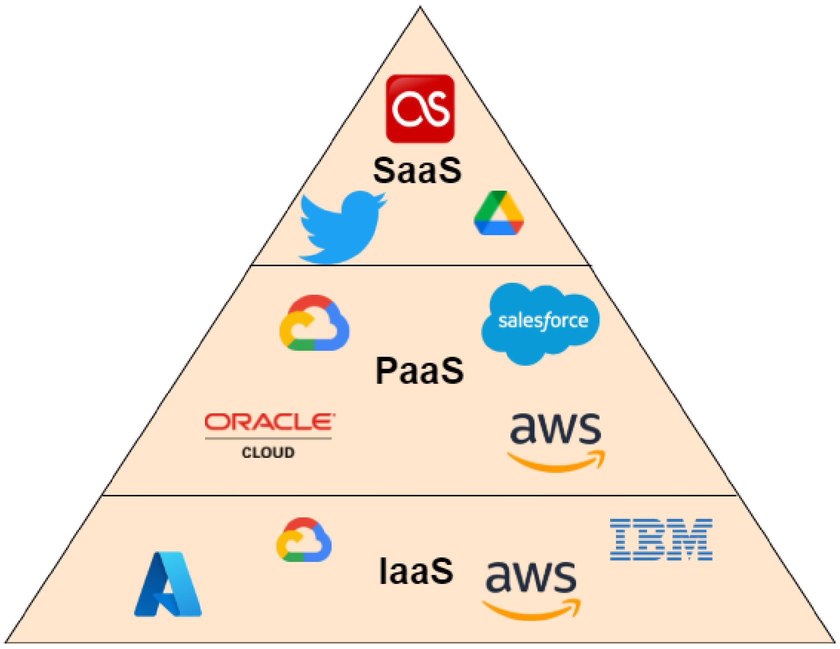

Figure 2.4: Cloud Computing Services Models

2.1.4.1 Types of Cloud Computing

Cloud computing provides storage solutions and services and these services follow a pay-as-you-go model for consumers. It can be divided into three types of service models, IaaS, PaaS and SaaS. [11], that describes the type of services that can be obtained from the Cloud. According to the chosen model, the service varies, this service is classified according to the level of the IT architecture in which it resides.

Infrastructure as a Service: The service provider offers resources such as storage servers, networking and data centres. In this case, it only has to supply the hardware components with the necessary capacity and consumers deploy and run their systems and applications. Despite there being many consumers with different requirements, the idea passed is that the hardware is what the consumer requests. Some providers of this service are Amazon

Web Service[1], IBM[2], Google[3] and Microsoft Azure[4]. [12]

The table 2.2 represent the advantages and disadvantages of IaaS.

Advantages	**Disadvantages**
High system scalability	Dependence on availability of infrastructure and networks
Redundant data storage	Unauthorised access to data due to possible misconfiguration
Physical separation between data usage and storage	Data location is not transparent in the cloud
Pay as you go	

Table 2.2: Advantages & Disadvantages of IaaS

Platform as a Service: This model is built on the IaaS model and prepared to support the application lifecycle (build, test, deploy, manage and update). In addition to providing storage servers, computing and data centres, it also provides software infrastructure such as operating systems, programming languages and development tools. [13] Allowing the user not to worry about the complexity related to software infrastructures. IaaS service providers like Amazon Web Service, Google and Microsoft Azure also provide the PaaS service, however, vendors like Oracle Cloud Platform[5] and Salesforce Platform[6] are major providers of this service.

According with the table 2.3, the advantages and disadvantages are explained.

Advantages	**Disadvantages**
Less administrative effort	Insufficient flexibility
Standardisation	Special requirements needed in case of access to proprietary applications/development environments
No maintenance required in running and configuring the platform/tools	Vendor lock-in due to lack of portability, interoperability and lack of standardised technologies
Pay as you go	

Table 2.3: Advantages & Disadvantages of PaaS

Software as a Service: In this type of model, the service provider offers a complete software solution to users as long as they have access to the internet, this access can be done through an API or through the browser. Applications can be various, from web based email

[1]`https://aws.amazon.com/`
[2]`https://www.ibm.com/`
[3]`https://www.google.pt/`
[4]`https://azure.microsoft.com/`
[5]`https://www.oracle.com/pt/cloud/`
[6]`https://www.salesforce.com/eu/`

to applications like Twitter[7] or Last.fm[8]. [12]. As the others models, also this has advantages and disadvantages, according to the table 2.4.

Advantages	**Disadvantages**
Fast deployment	Lack of portability
No maintenance required to run system functionality	Potentially longer response times
Application multi tenancy	Less integrability into the existing application environment
Pay as you go	Security vulnerabilities when using shared SaaS solutions

Table 2.4: Advantages & Disadvantages of SaaS

2.1.4.2 Types of Cloud Access

A company data must be carefully analysed before choosing the type of Cloud Computing to use in its infrastructure so that there are no implementation failures. In addition to the existence of three different categories in the services offered by Cloud Computing, we can also divide the type of cloud into three deployment models, which are differentiated by the form of access and physical location. [14]

Public Cloud: The provider provides the infrastructure and services and these are available to all users, dealing with its maintenance, management and security. There are paid and free public clouds. Amazon's EC2[9] is an example of a paid public cloud, while Google provides free storage, email and other tools for free. [15] The table 2.5 pretends to shown the advantages and disadvantages of this type of cloud .

Advantages	**Disadvantages**
Lower costs	Divided resources
Almost unlimited scalability	Less secure than the private cloud
No maintenance required	

Table 2.5: Advantages & Disadvantages of Public Cloud

Private Cloud: The private cloud is only available to the organisation, it can either create its own private cloud or hire a third-party provider to provide a private infrastructure. The management and supervision of the cloud is carried out entirely by the company, allowing full control of the system and greater security. [16] In this type of cloud, it is easier to scale resources and also meet specific requirements, since the hardware is dedicated to the organisation. According with the table 2.6, this type of cloud also has some advantages and disadvantages.

[7] `https://twitter.com/`
[8] `https://www.last.fm/`
[9] `https://aws.amazon.com/pt/ec2/`

Advantages	**Disadvantages**
High flexibility	Higher price on maintenance
More control and privacy	Higher price due to equipment acquisition
More scalability than the public cloud	

Table 2.6: Advantages & Disadvantages of Private Cloud

Hybrid Cloud: This type of cloud is the combination of the private cloud and the public cloud. Each remains distinct, however, they work together for a given organisation. For example, an organisation may decide to keep parts of its applications that it considers most sensitive in the private part of its cloud and parts of its less sensitive applications in the public part of the cloud. This is the most advantageous type of cloud, as it is flexible, as it allows you to take advantage of what each type of cloud (public and private) has to offer. Despite being the most advantageous type of cloud, as shown in 2.7 it still has some disadvantages.

Advantages	**Disadvantages**
More control	Difficulty in compatibility between infrastructures
Flexibility	Expensive
Cost-effectiveness	Network problem

Table 2.7: Advantages & Disadvantages of Hybrid Cloud

2.1.5 Digital Twins

Digital Twins technology has gained more and more fans, as it allows companies to develop, evaluate and improve their products and, consequently, reduce costs.

The idea of the DT came about in the 1960s at NASA, however it was only in the 1970s that it became more widely known when during the Apollo 13 mission the oxygen tanks exploded, destroying part of the main engine. It was then that the NASA team, committed to finding a solution in order to bring the team back to Earth, began to make simulations and analyse the conditions in which they found themselves in a mirrored model of Apollo 13. From this simulated environment, it was possible to model and test possible solutions, finding one. Engineers from Earth explained to the astronauts how to build the solution, which was based on an air purification system, using materials they had in the spacecraft. Despite not being an DT, as there is no connection to the asset or real time data exchange, this incident allowed the understanding of the added value that virtual and simulated models could bring. [17]

The idea behind the creation of the Digital Twin is the creation of a replica that is totally faithful to the physical object, so that it receives all the important data, and thus runs simulations, studies performance errors and generates possible improvements. In this way the engineers and other professionals can know exactly the end result of a project, its entire life cycle and even apply changes to the physical object.

The definition of Digital Twin most used and accepted by most people was given by *Glaessegen and Stargel* in 2012, which says that DT is a multiphysics, multi scalable and probabilistic simulation that uses the best available physical models, sensor updates and etc... that mirrors its twin's life [18].

2.1.6 Digital Twins and its Implementations

As already mentioned, a Digital Twin is the virtual representation of a system, which is indistinguishable from its physical part. Thus, it becomes possible to design, test, implement and use the virtual version of the system, which will enable the analysis of future solutions, understanding if they are viable or not, and understanding if and when a physical system will fail, all before it is produced. This is a very useful technology that can be applied in various fields, however, there are three main domains [19], that will be addressed in the following sub-sections 2.1.6.1, 2.1.6.2 and 2.1.6.3.

2.1.6.1 Digital Twins in Healthcare

It is possible to test various approaches to care delivery, train medical procedures according to the patient's anatomy and even prevent diseases with some time apart. With this, the DT will be able to provide insights that can lead to innovative practises that will correct error, minimise risks and also optimise patient care, decrease cost and increase performance by virtualising healthcare experiences. However, DT has been used more in this field, for preventive maintenance of medical devices and for their optimisation. [19]

An example of a Digital Twin implementation in Healthcare was developed by Siemens Healthineers to optimise Mater Private Hospitals in Dublin. The hospital was facing high patient demand, increasing clinical complexity and ageing infrastructure, making it impossible to provide efficient patient care, leading to the need to change the radiology department to overcome these needs. A computational model of this department and its operations was developed, resulting in a medical DT that allowed the optimization of a digital process by using workflow simulations and testing new scenarios. With the realistic animations provided by Digital Twin, and the reports produced, it was possible to predict the operational scenarios and evaluate alternatives in order to find the solution to the problem.From this solution the improvements were notorious, as there was a reduction in the waiting time, where patients started to wait 13 minutes less than usual for CT and 25 minutes for MRI, an increase in the use of the equipment with a capacity of 32% for MRI and 26% for CT, was also achieved. Consequently, there was a reduction in staff costs as an average of 50 minutes less per day was spent on MRI, representing a saving of €9,500 per year for the hospital. [20]

2.1.6.2 Digital Twins in Aviation

Although the DT technology, is used for various functions, in the aviation industry the Digital Twin is more used for predictive maintenance, seeking to avoid failures or know how to react to them in order to save lives.

An example is a study made by *Yang et al* [21], where a DT of an aircraft is described, in which an automatic image tracking method is used, this makes it possible to obtain knowledge about the crack tip deformation and growth behaviour of aluminium and steel cracks. This knowledge, allows Digital Twin to predict crack growth mechanisms during the whole life cycle of the aircraft, consequently allowing to reduce the cost and time required in maintenance and development.

A different example, is the work done by *Zakrajsek and Mall* [22]. In this work the DT mirrors a specific type of aircraft tyres when they land on the ground. The idea behind this study was that from this Digital Twin, a solution would be found to predict and avoid flat tyres when they touch the ground, as they can increase the environmental and logistical footprint of the aircraft. It uses parameters such as descent speed, landing and tyre conditions and considers the probability of failure according to these parameters.

2.1.6.3 Digital Twins in Manufacturing

Several works, explore the implementation of a Digital Twin in the manufacturing industry, as it mirrors the life cycle of a product, from the design phase to the finalisation phase, allowing to follow all the steps, predicting failures and looking for better production solutions.

An example is the work done by *Guo, Jiapeng, et al.* [23]. This work is done in the MBSE field, it is pointed out by the authors the use of DT models for factory design. This refers to the configuration of the equipment, the strategy of the logistic control unit and the whole integration with the production line equipment, allowing to generate a simulation of the factory system and enabling to make optimized decisions. Consequently, these decisions will increase production speed and reduce costs. As soon as the designers create the idea for the factory, the Digital Twin is created and grows and evolves as the factory design evolves. Even after the factory is ready, it is possible to easily reconfigure the modules if changes need to be made to the production line.

2.1.7 Digital Twin architecture

According with the image 2.5, the Digital Twin system architecture comprises two components, Hardware and Software components and data management middleware in between, responsible for connecting them. [24] To implement a DT it is crucial to understand them, to be aware of how they are formed, their functions and how they interact.

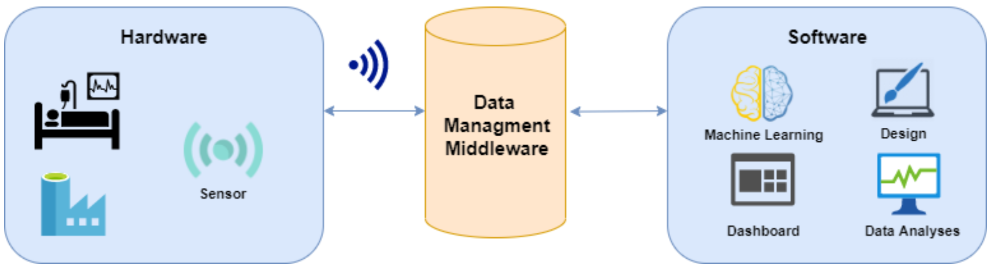

Figure 2.5: Components of Digital Twin system

2.1.7.1 Hardware components

The base technology that drives the DT are IoT sensors, these are responsible for initiating the exchange of information between the devices and their digital representation. Sensors measure and provide the information on which the system depends, such as temperature, pressure, humidity and other important characteristics, they are equipped with wireless interfaces for the acquisition of data that are consolidated through a gateway to the local network and then forwarded to the cloud.

There are several types of sensors that can be used in IIoT, among which the base sensors, smart sensors and intelligent sensors can be highlighted. Base sensors only collect and forward data to the network layer of the IIoT system, smart sensors, which collect and can do some minimal type of processing before being forwarded to the rest of the system and intelligent sensors, which allow increasing device resources as they are able to adapt to changes in conditions and are able to perform self-validation and tests, use more advanced computing technologies such as machine learning, becoming more useful in IIoT implementations.

2.1.7.2 Data Management middleware

It's a centralised repository, which accumulates the different data coming from the sensors and manages, treats and controls them. Ideally, it also handles tasks such as connectivity, allows the propagation of important information in order to keep consumers informed of changes, can filter message content and route updates only to necessary resources, handle data quality control and allow their visualisation.

Due to this component, messages can be distributed so that consumers can learn and evaluate the performance of certain actions [25]. It's in this component that the physical and digital twins connect, since the hardware component, after collecting the monitored data, stores them and where the software component will collect them.

2.1.7.3 Software components

What moves DT is the fact that it can analyse data and look for new valuable solutions for companies, in many cases this is achieved through the help of machine learning models. It is necessary to have several tools such as panels that allow real-time monitoring, simulation software and design tools for modeling. [26]

From these tools, it is possible, for example, to carry out a virtual verification and evaluation, as the Digital Twin provides a high-fidelity model of the product and its operating environment. Thus, a multidisciplinary and multi-scalable interaction is obtained between the product and its future operating environment. Since DT supports virtual prototyping and tests different versions of products in different application scenarios.

It is also possible to evaluate and optimise processes, since with the increase in manufacturing complexity it has become increasingly difficult to plan processes. Dynamic changes in the manufacturing process and the uncertain conditions of manufacturing resources have a major influence on product quality, real-time data acquisition and data mapping allow DT to be up-to-date with the actual manufacturing process and product conditions. equipment. The

Digital Twin collects measured data, deviation data, process planning data and uses these types of data for geometric, mechanical and material models, so the DT interacts with its manufacturing environment through interoperability interfaces, evaluating and optimising the combination of processes and parameters.

Virtual testing is still possible, as the Digital Twin can test operations under circumstances where failure of physical operation leads to loss and damage. So, DT works as a more realistic and accurate simulation, providing an identical simulated environment that can be explored and tested without negatively impacting physical systems.

2.1.8 Artificial Intelligence, Machine Learning and Deep Learning

As mentioned before, we are facing the fourth industrial revolution, oriented towards intelligent manufacturing. This means that in the future, the industry will be more intelligent, and the factories will be able to perform even more complex tasks independently. [27]

Intelligence shown by machines, are shown in the form of tasks performed, however, these tasks are only done due to the algorithms running in the background, that enable machines to perform tasks understanding the work environment and taking steps to maximise the possibility of successfully achieving objectives.

Artificial Intelligence refers to the intelligence demonstrated by machines and comprises several categories, being, Machine Learning (ML) one of it, since it is a field that tries to understand and build methods capable of learn, this is, methods that leverage data to improve performance of tasks. Nevertheless, there is also Deep Learning (DL), that arises from the need of overcome ML limitations, it's part of machine learning methods but with focus on artificial neural networks with representation learning methods with several levels of representation, these are obtained by composing modules that transform the representation at one level to a higher and more abstract level [28]. A diagram is shown in 2.6, that pretends to explain how DL and ML are organised. With the integration of Deep Learning algorithms, it's possible to upgrade the manufacturing process to another level, optimizing the environment by processing the information through its multi-layer architecture, thus this layers of features are learned from data using a general purpose learning procedure.

With the increase of data volume, generated by smart devices, sensors and machines of a factory, data is more easily available and the AI technologies, can create a value chain by gathering, processing, and learning from this data. [29]

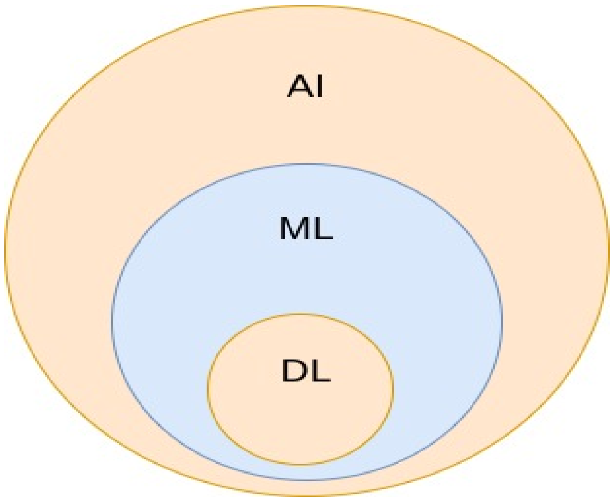

Figure 2.6: AI vs ML vs DL

2.1.9 Edge Computing

Before the emergence of Edge Computing, Cloud Computing had a centralised structure where computing and storage were deployed on a centralised server or distributed in the cloud. With the development of the IoT industry, Cloud Computing started to show some disadvantages as these devices require reduced latency and real-time response.

This development has led to an increase in data production which has caused bandwidth throttling which, despite the increase in data processing speed, has led to increased response times. Edge Computing emerged from the need to reduce the pressure on the network, reduce response time and make data processing more efficient. [30] Edge Computing extends the computing, network connection, and storage capacity from the cloud to the edge of the network, that is, data processing takes place at the physical location close to the data source rather than just in the cloud. By eliminating the distance and time required to transmit data to the Cloud, it became possible to obtain faster applications, with low latency, with real-time control and with control of the movement of critical data at a lower cost. It also avoids bandwidth limits, reduces transmission delays and limits service failures. This technology greatly benefits applications that depend on the response time, since it decreases, given that the processing is done at the edge of the network. [30]

Edge Computing allows for increased data security compared to Cloud Computing, because processing takes place close to the data source, making data theft and leakage more difficult.However, it still faces some security issues, as the edge nodes, servers and edge network can be attacked leading to the leak of private information.

It uses Edge Computing Nodes (ECNs), which can be smart devices, smart gateways, local clouds and intelligent systems, so that they serve as a bridge between the physical world and the digital world. They are responsible for controlling devices, acquiring data and monitoring the transaction of packages along the factory floor [31].They are mainly used to pre-process data, in real time at the edge, using rules established in advance, in this way they can discard unnecessary data, and forward only relevant data to the cloud, reducing the load on clouds. ECNs support plug and play of new devices and quick replacement in case of failure, their integration with IIoT makes the control stronger and allows to improve the scalability of the network. [32] They also provide temporary local storage, in case of connection problems with the Cloud the data can be temporarily stored on these nodes and later, when the connection is restored, the data is automatically synchronised with the Cloud.

2.1.10 Industrial Edge Computing and Industrial Cloud Computing

However, Edge Computing is only an advantage for the industry used together with Cloud Computing, since with efficient data flow between Cloud and Edge, the cost of data-driven product quality tracking and data mining could be reduced.

The Industrial Cloud is suitable for global, non real-time and long term big data analysis for preventive maintenance and support of important decisions, the data used here is collected by the ECNs. While Industrial Edge Computing is suitable for handling local, real-time and short-term big data analysis, crucial for making real-time decisions and controlling execution. [31] Industrial Edge Computing supports industrial cloud applications related to data mining and data pre-processing at millisecond speeds, leading to improved efficiency of data analysis, decreasing the communication delay between the edge and the cloud. It also executes strategies requested by the Cloud, including device, resource and connection management strategies.

The Industrial Cloud, on the other hand, implements optimised rules and models in the ECNs, based on big data analysis. In addition, applications in industrial clouds, including analysing device operations and optimising energy consumption, can be enhanced by real-time data collection by edge-computing devices. According to the reading, the fundamental requirement of Industrial Edge Computing is an ability to implement automatic control. With the support of ECNs, automatic control accuracy for machines, systems and processes can be improved with automatic fault detection, information processing and operation control techniques, while real-time constraints are guaranteed. [32] By monitoring the status of device operations and scheduling maintenance plans, it is possible to ensure that devices are always operating in good condition, reducing the time the system is down due to failures. [31]

In IIoT structures, a great challenge is the device compatibility, present due to the fact that there are several different protocols, to deal with this challenge, ECNs provide several access modes within their modular network interfaces. It is therefore possible to conclude that industrial Edge Computing has improved three areas in the industry: preventive maintenance of devices, quality control and optimization of the process of control and monitoring of product quality and optimization. The figure 2.7 pretends to represent the difference in the structure of Industrial Cloud Computing and Industrial Edge Computing.

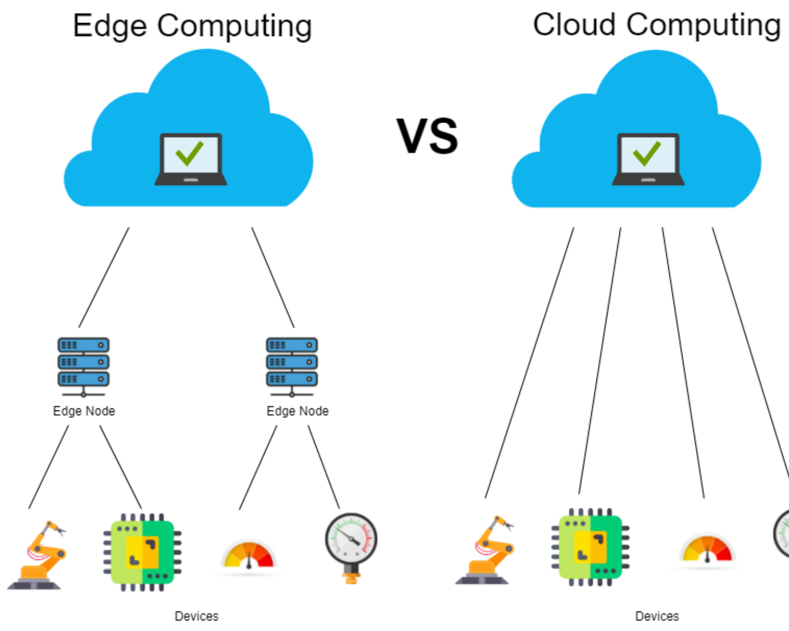

Figure 2.7: Industrial Edge Computing VS Industrial Cloud Computing

2.1.11 Reference Architectures for Edge Computing

A reference architecture is a document that presents recommended structures and integration of service/product technologies, incorporating industry-accepted best practises and suggesting specific optimal technologies to create a solution. [33] In this way, it facilitates collaboration and communication between project managers, software developers, designers and IT managers during project implementation. Below, two reference architectures for Edge Computing will be analysed.

2.1.11.1 Industrial Internet Consortium Reference Architecture

This reference architecture, represented in figure 2.8, is a template, with the objective to help developers and software architects build their industrial IoT systems based on industry recommended approaches. It also helps establish a common architectural approach across the IoT ecosystem.

This architecture is represented by three layers, edge, platform and enterprise, each of which has a specific function. [34]

- **Edge Tier**: This layer is responsible for collecting data through the ECNs over a proximity network. Depending on the use cases, architectural characteristics such as distribution breadth, location and proximity to the network may vary.
- **Platform Tier**: This layer is responsible for processing and forwarding control commands between the edge and enterprise tier. It consolidates processes and analyses data flow of the remaining layers, it is also responsible for providing management functions for devices. Through domain services, it is possible to manage active edge tier devices to query and analyse data.

- **Enterprise Tier**: This layer is responsible for implementing decision support systems, operations management, among others. It receives data streams from the lower layers (edge and platform) and generates control commands for them.

These layers are interconnected by different networks which are proximity, access and service networks.

- **Proximity network**: Responsible for interconnecting sensors, actuators and control systems, also known as edge nodes. These nodes are connected to a gateway that serves as a bridge to other networks.
- **Access network**: This network enables connectivity between data and control flows and edge and platform layers.
- **Service network**: This network, in addition to allowing connectivity between services of the same layer, also allows connectivity between the platform layer and the enterprise layer.

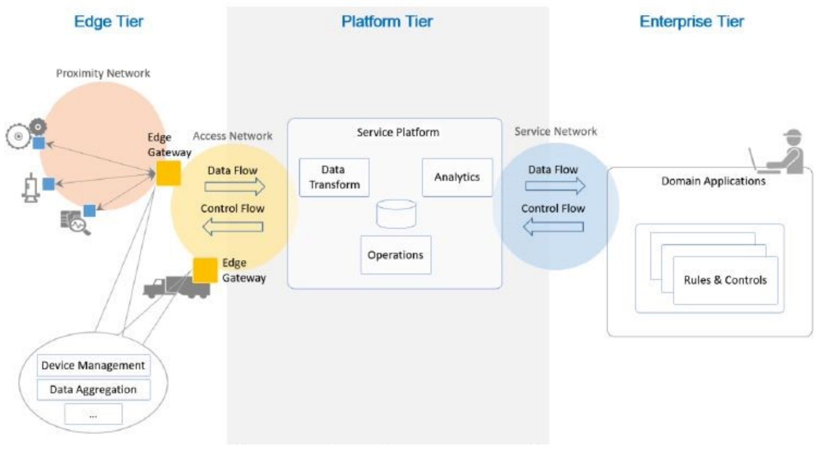

Figure 2.8: Industrial Internet Consortium Reference Architecture [34]

2.1.11.2 Internet Consortium Reference Architecture 2.0

This architecture has two perspectives, horizontal and vertical, according to the figure 2.9.

Horizontally it follows a layered model with open interfaces. It has four layers, Smart Services, Service Fabric, Connectivity and Computing Fabric and finally, Edge Computing Node. [35]

- **Smart Services**: This is based on a model-driven service framework. Coordination between service implementation and development is achieved through the Development service framework and Deployment and operation service framework. These frameworks allow for consistent development of software interfaces and automatic implementation and operations.
- **Service Fabric**: Consists of several types of functional services that collaborate with each other to implement specific service requirements. It defines the tasks, logical pro-

cesses, and parameter control of the processing phases, implementing a fast deployment of service policies and fast processing of various types of products.

- **Connectivity and Computing Fabric**: The Operation, Information and Communication Technology (OICT) infrastructure is responsible for the deployment of operations and coordination between the services of computing resources and the needs of the organisation.
- **Edge Computing Node**: In this layer, ECNs have real-time processing and response capability from the data they collect and are compatible with multiple heterogeneous connections.

Vertically it uses services such as management, lifecycle data services, and security services. In this way, it can provide intelligent services across all service processes throughout the service lifecycle.

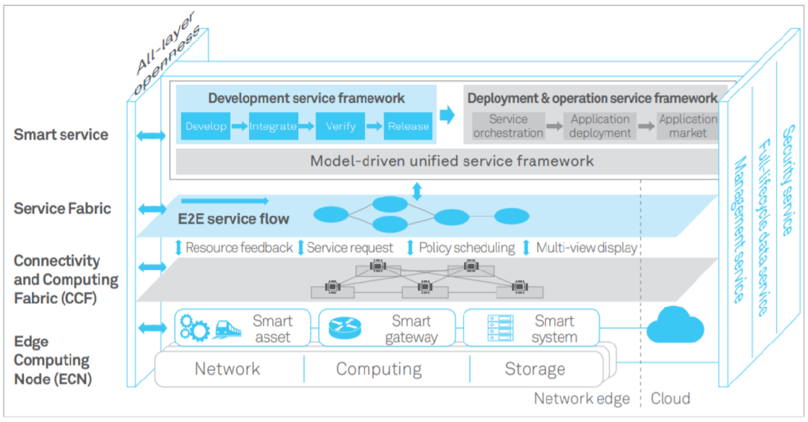

Figure 2.9: Internet Consortium Reference Architecture 2.0 [35]

2.1.11.3 Reference Architectural Model Industrie 4.0

Reference Architectural Model Industrie 4.0 (RAMI4.0) is a three-dimensional model that shows the most important aspects of Industry 4.0 and ensures that the participants involved share the same perspective and develop a common understanding.

It allows tasks and procedures to be divided into manageable parts and rules to be defined for implementing Industry 4.0 applications. It also helps to identify and position existing standards, allowing to identify gaps when applying these standards. The dimensions, as represented in the figure 2.10, are Life Cycle & Value Stream, Layers and Hierarchy Levels. [36]

The Life Cycle & Value Stream axis represents the lifetime of a product and the added value of a process, according to Standard IEC 62890. This is divided into two main parts, Type and Instance. Type describes the product from its conception and extends the development phase to the maintenance/use phases of the concept. [37] It is characterised by the design, development and testing applications through to the first sample and prototype of the product. Meanwhile, Instance describes the concrete product according to the idea formed in the Type state. After production, instances are sold and installed in systems. Improvements that are

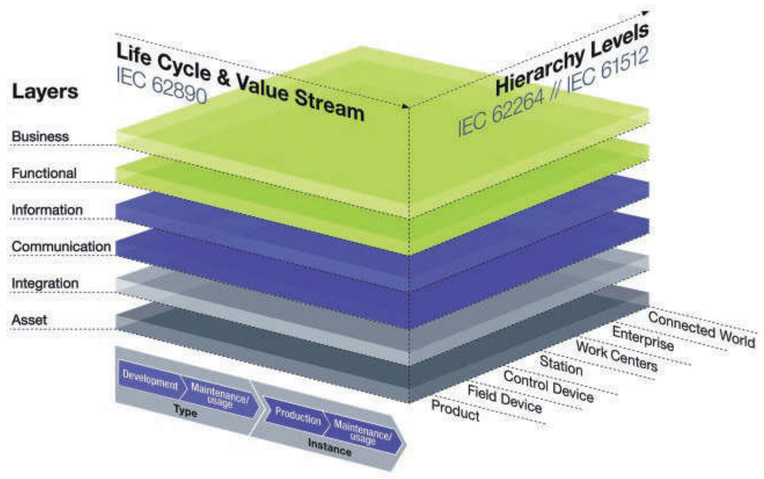

Figure 2.10: RAMI4.0 [36]

reported to the manufacturers of a product from the sales phase can lead to the change of type documents. New types can be used to produce new instances. And, like the instances, the type is also subject to usage and updates.

The "Hierarchy Levels" axis is based on the IEC 62264 and IEC 61512 Standards (responsible for the integration of enterprise IT and control systems) together with the reference architecture model for factories. Describes the functional classification of various circumstances within Industry 4.0. In order to consider uniformly and cover all possible sectors of an industrial process up to factory automation, the levels "Enterprise", "Work Unit", "Station" and "Control Device" were added, however, in industry the important thing is not only the plant and machines used in the production of the product, the product is also important. Hence "Product" was added to the last level of the hierarchy. [38]

This "Product" can be seen as part of the solution that aims to influence the production process. In Industry 4.0, both the control device and the considerations within a machine/system are decisive, and the "Field Device" level representing the functional level of an artificial intelligence device, such as an intelligent sensor, was then added to the hierarchy. A "Connected World" level was also added, as Industry 4.0 also describes collaboration between a set of factories and external engineering firms, not just between levels within the factory.

Finally, the "Layers" axis aims to describe the architecture in terms of system properties and structure with specific functions and data in the form of layers, these layers are, Asset, Integration, Communication, Information, Functional and Business. Interactions between layers can only happen between adjacent layers or within the layer itself. However, interactions can be passed through. [38]

The table bellow, table 2.8, represents the each layer and its description.

Layer	Description
Asset	• Represents reality (physical components) • Humans are part of this layer • Humans connect to the virtual world through the Integration Layer
Integration	• Provides asset information • Assets generate events • Where asset related properties and functions are stored • Contains IT related elements (RFID readers, sensors, HMI)
Communication	• Handles the communication needed to implement processes • Describes what data is used and where it is used • Describes the distribution of data • Based on the ISO-OSI layered communication model, it allows: o get a better view of standards o describe the architecture accurately
Information	• Describes the information model for the functions • Gathers important asset data and attributes • Formally describes rules • Execution of event-related rules
Functional	• Rules are generated • Describes an asset's functions according to its role in the industry • Remote Access • Horizontal integration
Business	• Represents business processes and legal conditions • Describes the business view • Models the rules that the system has to follow • Orchestrates services in the Functional layer

Table 2.8: RAMI4.0 Layers Description

2.2 Data Persistence and Data Storage

Over time more and more applications for databases emerged, consequently the demand for features such as simplicity, scalability, flexibility and performance increased, however, one feature stood out more than the others, the scalability. [39]

In the IIoT scenario, the need for both flexibility and scalability was perceived more. That's because, with the increase of IoT devices, the amount and diversity of data to be stored grow. Hence, IoT applications require high availability, more flexibility to scale up easily and a more flexible database schema to greet the changes.

The database solutions chosen for this analysis were those represented in the subchapters 2.2.1 and 2.2.2, according to the ranking between the most popular relational and non-relational databases. [40]

2.2.1 Relational Databases

As the name indicates, relational databases are databases that store and provide access to data that is related to another piece of data. This data is organized in tables, therefore it's more intuitive to represent and access data.

The main advantages of this type of database are that they comply with the ACID model, which stands for atomicity, consistency, isolation and durability. These four properties must be accomplished so that data validity can be guaranteed, the principle of this model is that if a change fails, the whole transaction will fail and the database will not be modified, remaining in the state it was in before the attempted transaction. Furthermore, another advantage is the use of primary and foreign keys, which allows data accuracy to be maintained, as it ensures that there is no duplicated data. Last but not least, allows simplicity due to the existence of a wide variety of tools developed to help in the initiation and interaction with relational databases.

Despite having many advantages, relational databases also have disadvantages. These databases are easily scalable, however this only happens if they are located on a single server, if the resources of the server where the database is located run out, it becomes necessary to increase the number of computational nodes in order to distribute the load over them. It is at this stage that problems start to arise, as the complexity of this type of database increases, which depends on the amount of scalability. Another disadvantage is the lack of flexibility, since it's mandatory to define columns and the data types for those columns, which makes it more complex to change the structure of the data. Another major disadvantage is the fact that with the growth of the number of tables and the amount of data in them increases query execution time, consequently decreasing the performance of the database. [41]

A program called Relational Database Management System (RDBMS) allows a person to create, update and administer a relational database, the majority of these programs use the SQL language to access and manipulate the database. SQL databases have many techniques to access and search data quickly and efficiently as indexing, sub-databases and query optimization.

Among the relational databases, we will discuss the three more important and the table 2.9 is used to demonstrate the main advantages and disadvantages of each.

1. **Oracle:**[10] This is the most used relational database, the query language that use is SQL language. It also uses conditional entry updates, composite keys, Unicode characters and full-text search. The integrity model used by Oracle is ACID, and it also offers some integrity features as revision control, isolation and referential integrity. However, Oracle Database has multiple types of constraints on tables, and when we are trying to insert new records at the database we have to be careful not to violate them.
2. **PostgreSQL:**[11] PostgreSQL uses and extends the SQL language and as Oracle, it uses the model integrity ACID.It is an extensible database, because it's possible to build custom functions and write code from different programming languages without recompiling the database. Beyond extensible, it's also scalable once it can manage a big amount of data and users it can support.
3. **InfluxDB:**[12] It's a relational database optimized for time series data, this is the right choice to handle metrics and events that are time-stamped. Influxdb is ready to store large amount of data and perform realtime analysis from it. It also allows you to create data management policies for a given time interval, for example, to delete data that already has a certain time. One interesting feature of this data base is that it compacts old data to save and uses a similar syntax to SQL.

Database	**Disadvantages**	**Advantages**
Oracle	Licenses are expensive Extensive SQL knowledge	High compatibility Robust security and privacy features
PostgreSQL	Comparatively low reading speed	Highly expandable Largely compliant with SQL standard
InfluxDB	Database for log	Fast Storage Time-series data retrieval

Table 2.9: Relational database comparison

2.2.2 Non Relational Databases

Non-relational databases, also known as NoSQL (Not Only SQL), emerged from the need to respond to the limitations presented by relational databases. Therefore, the principles applied in relational databases are not applied to these databases. At this type of databases, data is stored without the need to define a scheme, they don't use only tables to store data it also supports documents, graphs and associative arrays. Non relational databases were designed with the cloud in mind, making them great solutions for horizontal scaling. Thus,

[10] `https://www.oracle.com/pt/database/`
[11] `https://www.postgresql.org/`
[12] InfluxDB `https://www.influxdata.com/`

horizontal scalability is one main advantage of NoSQL databases. This is very important for IIoT scenarios as it deals with a great amount of data, and may need to increase the number of servers. Non-relational databases use the model BASE that's a model based on data availability, this acronym stands for basically available, soft state and eventual consistency. Non-relational databases are divided into four types, each type is defined according to how data is stored.

1. **Column Oriented:** Data are stored in columns and each key, row key, is associated with one or more columns. This way data can be stored more efficiently and save space if a value for some column is nonexistent, data isn't written into database. However, names of the columns don't have to match in each row. Column Oriented databases have the benefit of being flexible, so queries are fast. Due to being flexible, this database is good at handling "big data" and unstructured data. Nevertheless, compared with relation databases, this type of database is slower when handling with transactions. This happens because instead of using rows, they use columns to group the similar attributes together, which means that transactions have to be performed across multiple files. Examples of such databases are Apache HBase and MariaDB.
2. **Key-Value:** Data are stored as pairs of key-value. The key is a string and the value is the data to be stored, this data can be an object, a document or a usual data type. This type of database is considered the fastest of databases, once they are very simple. The simplicity of key-value databases became an advantage, because everything is stored as a unique key and value that is either the data itself, or its location, consequently reading and writing will be faster. However, this simplicity limits the type of use cases for which it can be used. More complex data requirements may not be supported. Examples of such databases are Redis and DynamoDB.
3. **Document Oriented:** Data are stored as collections of documents and each individual document can have different columns. Documents structures don't need to be similar, so every document can contain a different set of data, making it possible to achieve flexibility. In the type of databases, the data is stored in pairs, similar to key/value pairs and they are highly scalable. The most used formats to store data are JSON, BSON and XML, and due to documents being JSON-like, make reading and urdestanding much easier. A very interesting fact is that some document oriented database systems, allow the implementation of a schema validator, being able to take advantage of this type of database but with a defined data model. Plus, the databases are self-healing which means high availability. Examples of such databases are MongoDB and CosmosDB.
4. **Graph Oriented:** This is the most specialized of the non-relational database types, where data is stored in graph form. Relations are presented in the form of a graph where nodes act as the objects and edges as the relationship between objects, this makes the searches fast. Also, has the advantage of being flexible, as nodes and edges can be added easily. However, the more relationships you have the lower the performance, as they are not efficient in handling queries that span the entire database. Examples of such databases are Neo4j and Titan.

Among the non-relational databases, the three most used ones were chosen in order to understand how they work, and furthermore, to understand their advantages and disadvantages, using the table 2.10.

1. **Redis:**[13] Is a key-value database, when Redis run data are loaded into memory, so all the operations are running in memory and periodically Redis saves data to the hard drive.
2. **Cassandra:**[14] Is a column orieted database. This database has a component, partitioner, that defines how data must be distributed across the nodes. Cassandra architecture is a masterless ring, meaning that all nodes play the same role and making it easly to scale.
3. **MongoDB:**[15] Is a document-oriented database. MongoDB supports real time data processing and has is own language that implements GRUD operatios (create, read, update, delete).

Database	**Disadvantages**	**Advantages**
Redis	No queries	Open Source Wide range of data types
Mongodb	Hardware Requirements Non-indexed querys	Horizontal Scalability
Cassandra	No join or subquery support	Linear scalability Fault-tolerant

Table 2.10: Non-relational database comparison

2.3 Related Work

The main purpose of a Digital Twin is to provide an exact copy of a real-life device, in order to produce new product testing, failure prevention and so on. Several approaches have been made, in the sense of implementing a DT in several industries, one of them is a study made by *Qamsane, Yassine, et al.* [42], in which a methodology to develop and implement a Digital Twin for Manufacturing Systems is proposed. This solution fails in the sense that it only mention that it is important to have a low latency of data capture, however, it does not present any solution for the storage and processing of these, therefore does not propose a solution to one of the major needs of implementing a Digital Twin, that is data capture latency.

Another solution propose is described in [43] by *Talkhestani, Behrang Ashtari, et al.*, in this solution is intended the integration of a Digital Twin in an intelligent automation system, for this purpose are proposed architectures for a Digital Twin and for an intelligent Digital Twin. In this solution three requirements are focused, synchronization with the real device, active data acquisition and ability for simulation. In addition to the requirements,

[13]`https://redis.io/`
[14]`https://cassandra.apache.org/`
[15]`https://www.mongodb.com/`.

the importance of the existence of a database to store and process the data captured in real time is also emphasised. However, this solution fails because no tests are carried out to decide which is the best database to use, nor its type, which varies according to the type of data collected.

The work described in *IoTwins* [44] presents a solution for a distributed implementation of Digital Twins. It aims to democratise Industry 4.0 for companies that do not have the resources to invest in these tools, therefore it targets small and medium enterprises in the automotive components and infrastructure production sector in order to lower the barriers for the use of DT and increase productivity, security and resilience. This solution relies on an MQTT broker(Rabbitmq) to provide the data to the databases that will then provide the data to the dashboard. It also relies on the use of two databases, Influxdb and Postgresdb. Influxd is used to store data locally, while Postgres is used to store time-series data generated by the sensors. In addition, the broker, Rabbitmq, is used as the communication protocol to manage the flow of data to the rest of the architecture.

However, this solution fails as it uses two different databases for different purposes because different requests have to be made, sending the telemetry through different paths, leading to a loss of data integrity and greater consumption of resources.

The solution presented by *Lacueva-Perez, Francisco Jose, et al.* [45], represents the implementation of a Digital Twin in the model industry, with the intention of maintaining the quality and consistency of the parts in injection molding is a challenge. In this solution, the real-time monitoring system is fed through a REST API, and then the data produced by sensors are stored in Inflxdb, which is deployed in a container. However, this solution also implements a second database, MongoDB, responsible for storing the workflows that train and support predictive models, registering the data workflow received in Influxdb.

Through a broker, Mosquitto, the results of this predictive maintenance are communicated to the workers. As explained in the implementation of this work, the accuracy of the prediction models varies according to the quality of the data provided to the system. To improve this quality, only one database must be used to maintain the data's integrity. Furthermore, this solution fails as it does not rely on data processing, which will lead to a loss of resources by storing irrelevant data in the database, increasing the amount of data in the database and consequently increasing the complexity.

CHAPTER 3

Proposed Solution

“Learn as if you will live forever, live like you will die tomorrow.”
Mahatma Gandhi

As described in chapter 1, more and more industries are embracing Industry 4.0 by trying to create new products or make the production process more efficient, meeting customer expectations. However, customer orders can vary sharply over time, and industries must respond as quickly as possible. With the implementation of Digital Twin, these requests can be answered quickly, as it allows the idealisation of a product and its test using a cyber-physical platform.

In chapter 2, various emerging technologies, such as Cloud Computing, Edge Computing and the Digital Twin, were discussed that somehow optimise production time. These technologies, associated with others, have enabled the creation of a cyber-physical platform. Although there are already several IIoT platforms, few are those that integrate DT technologies. This dissertation came up to deal with IIoT scenarios to find an efficient solution, where possible, to integrate a Digital Twin in an Industrial Internet of Things platform.

Traditional industries use sensors in production equipment and infrastructure, such as piping. These sensors capture the data that allows the industry to improve efficiency and productivity, allow better communication between machines and provide operators with data that enable a clearer view of how the equipment works. Thus, industries can evaluate their energy consumption, water and other resources. However, they cannot evaluate with precision if the products idealised by the customers will be possible to produce, nor will the costs. This would require manufacturing the product necessary to assess whether it is viable, leading to a loss of time, resources and capital. Considering these problems, this chapter will present a solution to deal with them. For this, architecture and its requirements will be proposed, and a typical architecture will be presented as a model for comparison. However, it will also be necessary to consider the data in the solution. Since this is a solution for IIoT, the most common data are pressure, temperature and humidity. In section 3.1, the data types used and how they fit into the solution will be discussed, with the help of the 3.1 image.

In the section 3.1, the data types used and how they fit into the solution will be discussed, with the help of the 3.1 image.

3.1 Data

As Clive Humby said, "Data is the new oil", and it is possible to assume that accurate data powers entire industries and has tremendous value.

When a company can understand its data and flow, decision-making becomes more agile, reliable and assertive. An IIoT platform has this logic as its foundation. It is essential to understand the type of data the industry is dealing with and, through it, make decisions that lead the company to achieve meaningful results. The types of data are divided into two categories, Qualitative and Quantitative. The image 3.1, shows path taken for the chosen data.

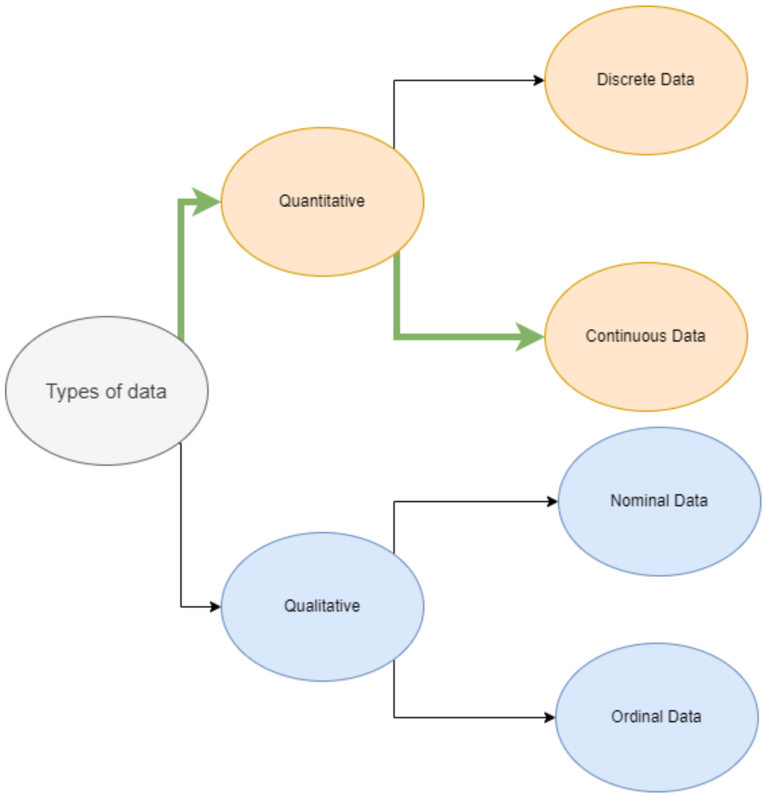

Figure 3.1: Selected data

Qualitative data are data that cannot be measured or counted in numbers but instead in categories, are represented in the form of audio, images, videos, etc. and can be further subdivided into two types of data.

The first type is nominal data, used to label something without assigning any numerical value, such as gender. It can take various labels like a female or male, among others. The second type is ordinal data, categorised on a specific range and cannot be used for any arithmetic operations despite having a numerical value. Instead, it is used for observation, categorising the degree of satisfaction of a customer and even the state of a disease. This data is relevant in industrial evolution as it allows us to understand the vision of the final consumer about the product and, according to the data collected, understand what is missing in the product and how to improve it.

However, despite their importance for industrial evolution, the relevant data for this study fall into the quantitative category, as they are expressed in numerical values used by researchers for mathematical calculations and statistical analysis in order to make real-life decisions based on mathematical derivations. Similar to qualitative data, this can also be divided into two types. The first type is discrete data, countable values that do not allow division. That is, they cannot be fractional or decimal values. Some examples of These data types are the days of the week and the total number of people in a room. The second type of data is continuous data, and this is the type used for this study as it can be represented in the form of fractional or decimal numbers, i.e. it represents information that can be divided and measured on a scale and can take any value within a finite or infinite limit of values, as mentioned earlier, the most relevant data in an industrial scenario are, among others, humidity, temperature and pressure, which will be the data analysed and fit this type of data.

3.1.1 Data in Digital Twin

As described in chapter2, the Digital Twin is a replica of a physical object that receives all the crucial data, performs simulations, studies performance errors and generates possible improvements. For this, it is necessary that the data received by it be 'live'. That is, there cannot be a significant difference between the state of the physical device and its virtual representation at a given moment. One final point to be borne in mind is that for data to be considered life, it needs to be updated to reflect changes in the state of the measured subject. If a photograph is taken, it cannot be considered a ‘live’ event, as it does not retain parity between the real and virtual world. Thus a deviation between the two is possible at any point.

However, in a real scenario, there is always a delay in the transmission of data from its production to the capture and process to the digital copy. This delay is known as capture latency. With the decrease of this latency, more accuracy and fewer errors are found based on old and outdated information since a user can make the same decisions and actions using the DT information as directly accessing it at the source. To get closer to a real scenario, in this study, several tests were made with a capture latency of 10ms, 20ms, 50ms and 100ms and also for reference, and tests were made with no latency, that is, in a perfect scenario, all with the same amount of data.

As shown in figure 3.2, represented below, if the capture latency is too high, the virtual object will not be an authentic representation of the physical asset since they are not in the same state, making the Digital Twin useless since the data is not accurate and does not allow a correct analysis of it.

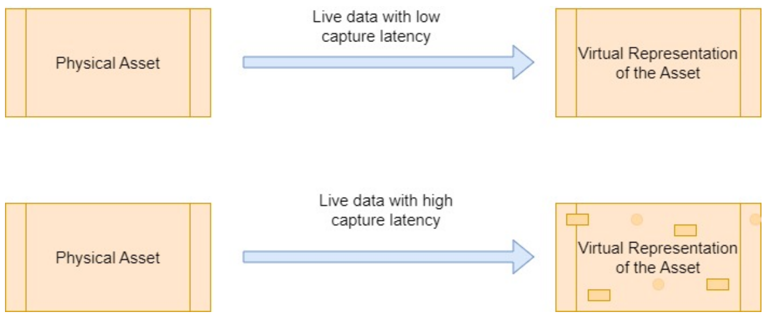
Physical Asset
Live data with low capture latency
Virtual Representation of the Asset
Physical Asset
Live data with high capture latency
Virtual Representation of the Asset

Figure 3.2: Low capture latency vs. High capture latency

3.2 Requirements

As discussed, this work aims to create and integrate a Digital Twin solution into an existing IIoT platform that can handle a large amount of data as efficiently and quickly as possible. To develop this work, it is necessary to take some requirements into account in order to make this platform more efficient. In order to achieve this, a set of dummy data was provided. Although it is not data provided directly from machines of a factory, the values such as temperature, pressure and humidity provided allow us to proceed with this study since they are actual data and of the same type of data provided by industrial machines. The table represented in 3.1, pretend to show the system requirements.

Since not all the data obtained is relevant to the study, a selection of the necessary data was made, discarding the remaining data such as the battery, link quality, and voltage, one of the requirements of this solution is "Data Processing". The second requirement is "Live Data", as already explained before, the data must be 'live'.

Data storage must be done efficiently, as a large amount of data is produced by the devices in the factory every minute, making this the third requirement needed to achieve the goal, named "Storage Capacity".

The last requirement, "Vertical Scalability", comes from the need for the system to handle a massive amount of data efficiently. That is, the system must be scalable since, in an industry, more assets can always be added, and the volume of data can increase significantly.

Req#	**Requirement**
1	Live Data
2	Data Processing
3	Storage Capacity
4	Vertical Scalability

Table 3.1: System Requirements

3.3 Proposed Solution

Considering the requirements previously discussed and the technologies presented in chapter 2, an evaluation of technologies was made to verify their efficiency.

Starting from the base, the first step for the evaluation is related to data processing. It was necessary to have a platform capable of allowing the processing of a large amount of data. For this, there must always be the availability of computational and network resources, aiming for a solution that uses technologies such as Cloud Computing, Containerization and Container Orchestration to eliminate the limitations once offered by traditional deployments. As mentioned in chapter 2, Cloud Computing allows scaling and configuring infrastructures more quickly and cheaply, and Containerization and Orchestration solutions allow a higher fault tolerance.

The SCoTv2 [46] allows for meeting the requirements related to data processing and vertical scalability, being that this is the infrastructure used for the proposed solution.

Regarding the data persistence analysis, some databases were analysed, and some operations were performed to analyse their efficiency. However, only time series databases were considered since we are working with time series data. For this study, four-time series databases were analysed, to which two of them, Apache Cassandra[1] and InfluxDB[2] proved inefficient since the insertion time needed to be lowered, not satisfying the second requirement, leading to a very high capture latency, making it impossible to obtain an accurate digital twin. The databases that proved to be a good solution, fulfilling the requirement, were TimescaleDB[3] and MongoDB[4]. However, sometimes, TimescaleDB showed better performance and was thus chosen for our final solution.

Several aspects were considered, such as data acquisition, processing and persistence, to build an architecture for this platform and satisfy some of the proposed requirements. Initially, it is necessary to have a data source responsible for the generation and provision of the data used, which will be forwarded to the input ports of each piece of equipment. Once the data is at the gateway of each piece of equipment, techniques will be used to enable its access and collection. For this data collection, it is necessary to consider that the data coming from the data sources is confidential information and must be forwarded to the structure securely.

It is mandatory to have a structure that receives the data from the source, processes it, stores it, and persists it. In this structure, a broker will be part of the data acquisition because it will serve as a bridge to receive and forward the data to where they are needed. Finally, there will also be a database responsible for data persistence. Figure 3.3 represents the proposed structure, where the arrows represent the data flow, being the data in real-time, and the squares represent the blocks of data processing, acquisition and storage, without referring to the selected technologies. This figure represents an overview of what an ideal structure should look like.

After this processing, as soon that data is ready, it is possible to send it through the gateway to the Digital Twin, where it will be used for various purposes, as mentioned in

[1]`https://cassandra.apache.org`

[2]`https://www.influxdata.com/`

[3]`https://www.timescale.com/`

[4]`https://www.mongodb.com/`

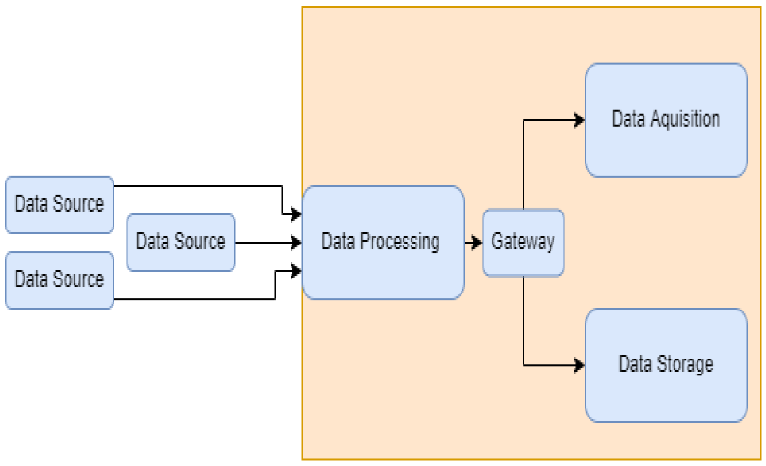

Figure 3.3: Proposed Solution

chapter 2.

CHAPTER 4

Implementation and Results

"Data by itself is useless. Data is only useful if you apply it. "

Todd Park

Considering the previously proposed solution, its implementation is described in this chapter. Thus, the chapter is organised as follows: section 4.1 describes the architecture chosen for the implementation of the solution, section 4.2 describes the implementation of the Digital Twin in an IIoT environment, section 4.3 describes the data storage selection and implementation, section 4.4 presents the results obtained and their explanation and finally, section 4.5 describes the evaluation of requirements.

4.1 Architecture

In order to achieve a more sustainable world, the digitization of the factory floor is a great ally, as it will allow us to have a global view of automation and efficiency where AI can help to use automated processes and procedures.

For this, it is necessary to create a cyber-physical platform. A solution for this platform is represented in figure 4.1.

This architecture was built taking into account three fundamental aspects: data acquisition, processing and storage. There are different types of protocols, which allow the architecture to communicate with the devices. The most used protocols in IoT platforms are AMQP, MQTT, HTTP and CoAP which represent data acquisition.

Effective data processing is possible to achieve due to Eclipse Hono[1]. It allows the connection of many IoT devices, which can use different protocols, to the rest of the architecture blocks. It receives telemetry data from the devices and exposes them through an API. It also has integrated security, as it allows simple authentication mechanisms, X.509 certificates and uses the transport layer security (TLS) when connecting to devices.

[1]`https://www.eclipse.org/hono/`

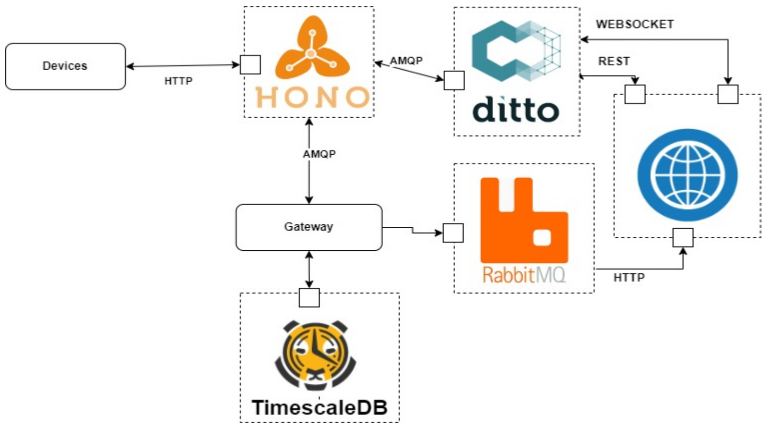

Figure 4.1: Proposed architecture

Since we are dealing with time-series data, the storage must be done taking into account its type. There are several solutions for this data type storage, such as Apache Cassandra or TimescaleDB. The database chosen for our solution was TimescaleDB, a relational database and an extension of PostgreSQL. It was an advantage since it inherited its reliability and tools. Data is sent to the database through a gateway.

The Digital Twin chosen was Eclipse Ditto[2]. It provides a framework that allows working and managing the state of the DT, creating a bridge between real-world IoT devices and their digital twin. Another service included is RabbitMQ[3], a message broker whose function is to expose data collected from devices through AMQP.

Finally, we have the last block, where the tools that will use the data collected from the devices and the Digital Twin are found. These tools range from visualisation tools such as Grafana[4], technologies for Machine Learning such as Tensor Flow[5] and among many others.

4.2 Digital Twin in IIoT

The first step was deciding which Digital Twin to use in our solution. The chosen one was Eclipse Ditto, which provides APIs for Web/mobile applications and other services. In addition, it allows it to work and manage the status of the digital twins and provide additional services to the device. In the case of Eclipse Ditto, it has two communication channels (twin and live channel).

Regarding the function of Ditto in Twin View, shown in Figure 4.2, when a device is connected, it indicates its prior status and saves it. However, there is no historical data, so only the current status is saved. When the current state changes, the previous one is lost, and the new one is saved, alerting APIs of the changes

[2]`https://www.eclipse.org/ditto/`
[3]`https://www.rabbitmq.com/`
[4]`https://grafana.com/`
[5]`https://www.tensorflow.org/`

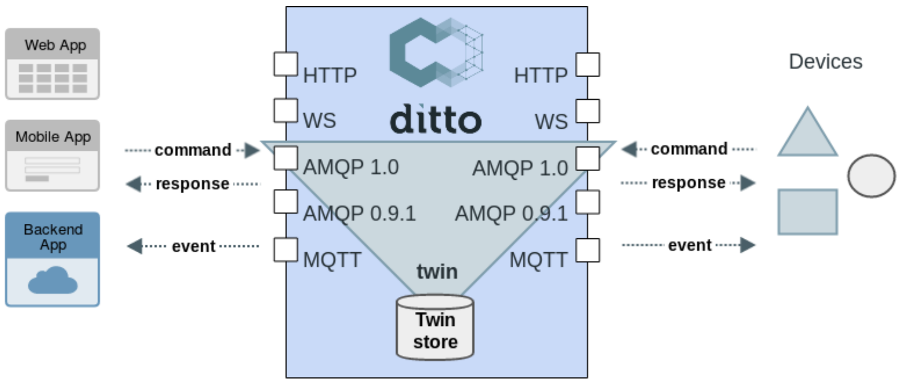

Figure 4.2: Twin Channel perspective

Then there is the Live View, shown in Figure 4.3, where Ditto works as a router, as it does the routing and enforces access control if an application wants to know a sensor's live temperature/humidity value, Ditto routes that message to the device.

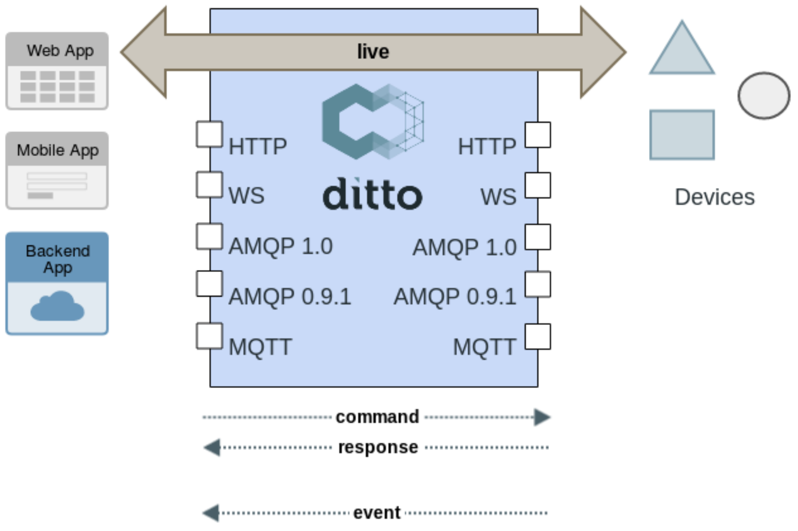

Figure 4.3: Live Channel perspective

4.2.1 Digital Twin implementation

The first step to implement the DT was to create a connection between Eclipse Hono and Eclipse Ditto, and Eclipse Hono and the gateway to allow connectivity of the devices to the database and the Digital Twin and to allow data processing.

As mentioned in the previous chapter, the data most commonly provided by machines and used for studies are temperature, humidity and pressure. When creating the Digital Twin of a device, the first step is to create a policy, and this is needed because it allows developers to configure access control for entities. In other words, it is responsible for deciding if a person has permission to read/write a given resource. So, in this policy, Eclipse Hono can read and write from messages and Things.

The next step is the creation of the Digital Twin, and it is necessary to provide a data model where mandatory fields are defined, like the policy ID and the features we want the twin to have. In code 4.1, there is a representation of the function used to create the Digital Twin.

Listing 4.1: Digital Twin creation example

```
class EclipseDitto:
  def create_digitalTwin(self, device):
        url = 'http://{}:30870/api/2/things/org.acme:{}'.format(self.addr, device)
        headers = {'Content-Type': 'application/json'}
        json_data = {
            'policyId': 'org.acme:my-policy',
            'attributes': {
            'location': 'Portugal',
            },
            'features': {
            'temperature': {
                'properties': {
                    'value': None,
                },
            },
            'humidity': {
                'properties': {
                    'value': None,
                },
            },
            'pressure': {
                'properties': {
                    'value': None,
                },
            },
            },
        }
```

Once the DT is created and all the connections are made, the system is ready to be executed.

4.3 Data Storage Selection

With factory floor digitization, data production is very high, so it was necessary to find the best database in terms of performance to store all the data produced by them. To this end, dummy devices that connect to Eclipse Hono tenants were created and fed their values(temperature,pressure and humidity) to the databases.

Ten tests were made, for each database, for 500, 1000, 2000, 3000 and 6000 lines of data and the files were respectively 73.3kB, 153.4kB, 334.2kB, 497.6kB, and 981.8kB. However, for the tests performed, there was no time between insertions; as in a real scenario, there is the time between insertions. Five more tests were performed for each database, with the same conditions as the previous tests but with variations in insertion times of 10ms, 20ms, 50ms and 100ms.

As we saw earlier, the chosen database would have to be a data-series database, so we chose the best four databases, MongoDB, TimescaleDB, Apache Cassandra and InfluxDB. We tested our solution with each one of them. For all of them, a node was deployed in a Kubernetes cluster.

4.3.1 Data Storage Implementation

In terms of implementation of each of the databases used in this study, the images below identified have 4.2, 4.3, 4.4 and 4.5, representing the code made for Influxdb, MongoDB, Apache Cassandra and Timescaledb, respectively. Here is represented the method that this study has the most value, the method of saving data to the database. As we are dealing with a large volume of data, its efficiency is vital to coping with the speed at which devices generate data.

Regarding the difficulty of implementation, the database whose level of difficulty is higher is Timescaledb because it implements a type of table, Hypertable, that other databases do not implement. This type of table has unique features that allow them to improve performance and handle time-series data more efficiently and faster. In this database, to define the Hypertable, it is also necessary to define a table, where the type of data that will be part of it is defined.

To implement this database, the first step required was to deploy a node of this database in the Kubernetes cluster. Once it is deployed and running, we connect it to the database. This connection is made using the Psycopg2 library, an adapter between the python language and the database. A basic authentication was used, where we provided the username and password. After forming the connection string, consisting of the IP and associated port, username, password and database name, the connect function of the Psycopg2 library was used to finalize the connection. The next step was to create the database table with the desired fields, indicating its type and other restrictions (char size, foreign keys, etc.). Once the table is created, we can finally create the Hypertable. Once the desired table and the associated Hypertable are created, the store function is defined, and the necessary steps to store the data are performed.

A Prepared Statement is created. This is not mandatory. It is defined since these statements are queries parsed by the database to be used later. Since they are already prepared, it is only needed to send the data when it is necessary to make a query to the database. In this way, it is optional for the database re-parse the query every time it is performed. Finally, the record with the data we want to save is created according to the prepared statement previously created, and the query is executed through the execute method. These sentences have the benefit of reducing the use of the CPU.

Listing 4.2: TimeScaleDB implementation

```
import psycopg2
class TimeScaleDB:
   def __init__(self, host='10.105.88.216', port='5432', user='admin', pwd=PWD,
       dbName='tsdb'):
       CONNECTION ="postgres://{}:{}@{}:{}/{}".format(user, pwd, host, port,
           dbName)
       self.conn = psycopg2.connect(CONNECTION)
       self.cursor = self.conn.cursor()
       self.conn.commit()
       query_create_sensordata_table = """CREATE TABLE sensor_data (
                             time TIMESTAMPTZ NOT NULL,
                             temperature VARCHAR(50),s
                             humidity VARCHAR(50),
                             pressure VARCHAR(50),
                             FOREIGN KEY (sensor_id) REFERENCES sensors (id)
                             );"""

       query_create_sensordata_hypertable = "SELECT
           create_hypertable('sensor_data', 'time');"

       cur = self.conn.cursor() cur.execute(query_create_sensordata_table)
           cur.execute(query_create_sensordata_hypertable)
       #commit changes to the database to make changes persistent
       self.conn.commit()
       cur.close()

   def store(self, queue, tenant, device, timestamp, data):
       if data:
              data_json = json.loads(data)
              # timescale DB query execution
              insert_query = "INSERT INTO sensor_data (db_insert_time, queue,
                  tenant, device, timestamp, temperature, humidity, pressure)
                  VALUES (%s, %s, %s, %s, %s, %s, %s, %s)"
              record_to_insert = ("now()", queue, tenant, device, timestamp,
                data_json['temperature'], data_json['humidity'],
                data_json['pressure'])
            self.cursor.execute(insert_query, record_to_insert)
            self.conn.commit()
    else:
        print(data, "No data")

def close(self):
    print("Close connection TimescaleDB")
    self.cursor.close()
```

The second most challenging database to implement was Influxdb. In this database, you insert points, where each point represents a data record.

The first step of the implementation flow of this database was similar to the first step of the Timescaledb implementation, and it was to deploy a node of this database in the Kubernetes cluster. To connect to this database, it was necessary to create an object, InfluxDBClient, which receives as arguments the URL, username and password (to proceed to authentication) Moreover, the name of the organisation, a default name, was chosen. This object has a method, write_api, which is used for configuration. This itself has a write method responsible for writing to the database.

Before using the write method, the points must be created, and each point must be similar to a row in a SQL database table. Each point has a measurement, a set of tags, a key and the field corresponding to that key. In order to make this correspondence, it was made a conversion of the data to that format generated the points to be able to insert them in the final database.

Listing 4.3: Influxdb implementation

```
import json, time
from influxdb_client import InfluxDBClient

class InfluxDB:
    def __init__(self, url='http://localhost:8086', user='admin', pwd=PWD,
        org='primary'):
        self.client = InfluxDBClient(url=url, username=user, password=pwd, org=org)
        self.write_api = self.client.write_api()

    def store(self, queue, tenant, device, timestamp, data):
        if is_json(data):
            points = json_to_points(queue, tenant, device, timestamp, data)
            try:
                self.write_api.write("primary", "primary", points)
                print('Inserido ', points)
            except Exception as e:
                logger.exception(e)
        else:
            logger.warning('Skip data; not json...')

    def close(self):
        print("Close connection InfluxDB")
        self.client.close()
```

The most accessible database to implement was Mongodb, and this is because it is not necessary to define the attributes or their type. Is only necessary to create a dictionary and add the fields we want to store.

The first step was the same as the first steps of the previous databases. A database node was deployed to the Kubernetes cluster.

To make the connection with the database, an object was created, MongoClient. It receives the username and password to proceed to authentication. Besides these arguments, it also receives the database name chosen by the user and the IP and port associated with that node to proceed to the connection.

As the database name is defined, creating a collection associated with the database is necessary. This works as a kind of table. Nevertheless, the documents saved in this collection may have different fields. Which does not happen, for example, in Timescaledb, the data and the types of data we intend to save must be defined when the table is created.

Finally, we save the data using the insert_one method. It is possible to use the insert_many method, which allows you to add documents in bulk. However, the insert_one method was chosen as the data is received sequentially, making it easier to use.

Listing 4.4: MongoDB implementation

```
from pymongo import MongoClient
class MongoDB:
    def __init__(self, host=HOST, port=PORT, dbName='testMongo'):
        client = MongoClient('mongodb://127.0.0.1:27017/my-database',
            username='my-user', password=PWD)
        db = client['my-database']
        self.collection = db['test-collection']

    def store(self, queue, device, tenant, value_final, timestamp):
        post={
            "type": queue,
            "device": device,
            "tenant": tenant,
            "data": json.loads(value_final),
            "timestamp": timestamp.isoformat()
        }
        self.collection.insert_one(post)
```

The second most accessible database to implement was Apache Cassandra. The implementation of this database is different since it is made a basic authentication using the PlainTextAuthProvider, which is done by providing the username and password. As well as the previous databases, a node of this database has been deployed on the Kubernetes cluster.

To interact with the database, it is necessary to create a Cluster, which is responsible for serving as a contact bridge. The connection to the database is only made through the connect method, which allows the connection to the nodes. Finally, we need to insert the values using the prepared statement and the values we want to insert as arguments in the execute function.

Listing 4.5: Apache Cassandra implementation

```python
from cassandra.cluster import Cluster
from cassandra.auth import PlainTextAuthProvider

class ApacheCassandra:
    def __init__(self, host=HOST, port=PORT, user='admin', pwd='password',
        keyspace='bitnami_kong'):
        # Basic authentication
        auth_provider = PlainTextAuthProvider(username=user, password=pwd)
        self.cluster = Cluster([str(host)], auth_provider=auth_provider)
        self.session = self.cluster.connect()

        self.insert_values = self.session.prepare('INSERT INTO values (queue,
            tenant, device, times, values) VALUES (?, ?, ?, ?, ?) IF NOT EXISTS')

    def store(self, queue, tenant, device, timestamp, data):
        data_json = json.loads(data)
        self.session.execute(self.insert_values, [queue, tenant, device, timestamp,
            data_json])

    def close(self):
        print('Close connection Apache Cassandra')
        self.cluster.shutdown()
```

Since we are dealing with IIoT scenarios, as described above, our main interest is to build a more efficient and flexible system in order to deal with industry demand and keep up with the industrial revolution.

4.4 Results and goals achieved

First we started with database evaluation and, as it was referred before, were made 10 tests for each one with 500, 1000, 2000, 3000 and 6000 lines, then the average time of insertion of each database was found, with this we can assume the average time of insertion per row. For these tests there was no time between insertions, this is, an ideal scenario has been assumed, where there is no capacity latency.

There are a few notions to be aware of before evaluating the results, (i) the insertion time can be conditioned by the machine resources (RAM, CPUs,...), (ii) insert time is not an absolute value and, finally, (iii) the performance of a database can also be influenced by its design and number of tables. The machine used to carry out this tests was a machine with 16GB RAM, 4 CPUS and 40GB Disk Space.

As shown in Figure 4.4, we can assume that the best database is TimescaleDB, with increasing data volume the average time per row tends to remain the same, this can be explained because it uses Hypertables, which allow the partition of data by time, allowing queries to be faster by excluding irrelevant data. In contrast InfluxDB's insertion time tends to increase with increasing data.

Despite InfluxDB being very used, during the insertion of each data line it starts to get slower with each insertion, this can be influenced by the language drivers (python) used for the database (influxdb_client), another explanation could be the fact of inserting data line by line instead of inserting it in batch, since it allows the decrease network overhead when writing data.

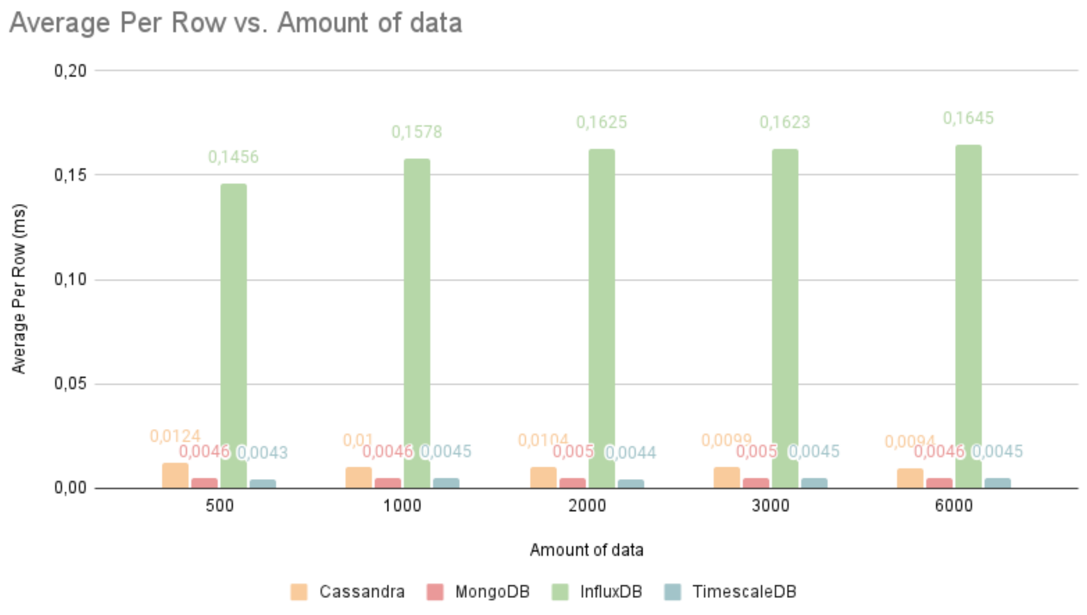

Figure 4.4: Insert time per Row vs Amount of Data without time without time between insertions

The following figures represent a capacity latency of 10ms, 20ms, 50ms and 100ms, respectively.

As we can see there are no significant differences in these results, so the capacity latency is not relevant for insertion time in a database. With these results, we can conclude that the Digital Twin created can thus be an almost perfect twin of the devices, because when the received data are being inserted into the database, the "thing" is being mapped, which causes that the data are also being sent to this twin, with latency practically equal to the time of insertion per row into the database. With this, we can conclude that the requirement "Live

Data" was achieved.

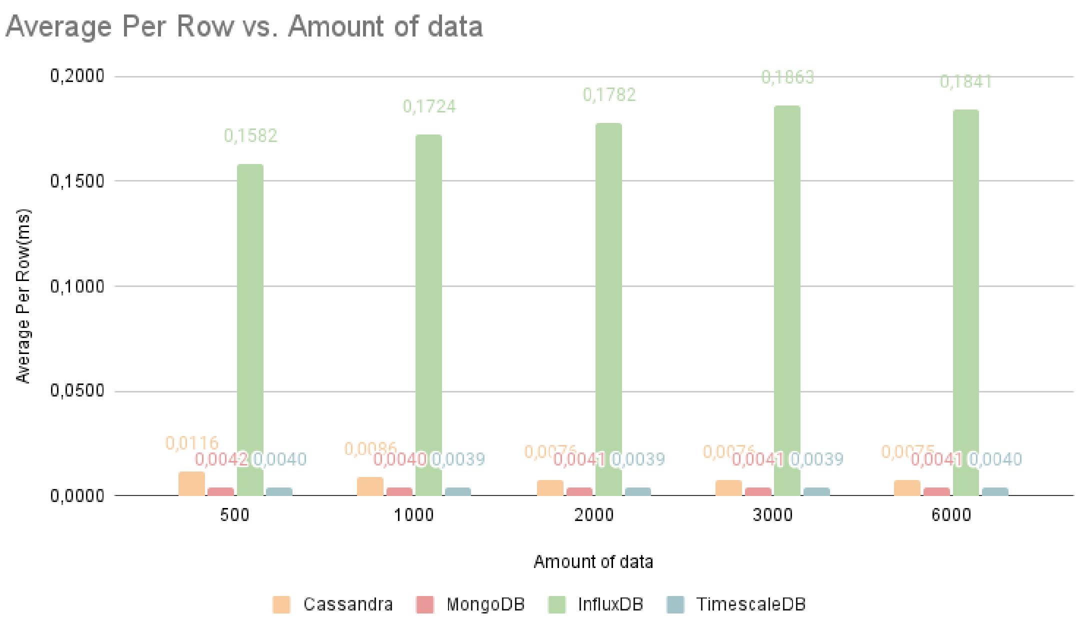

Figure 4.5: Insert time per Row vs Amount of Data with 10ms between insertions

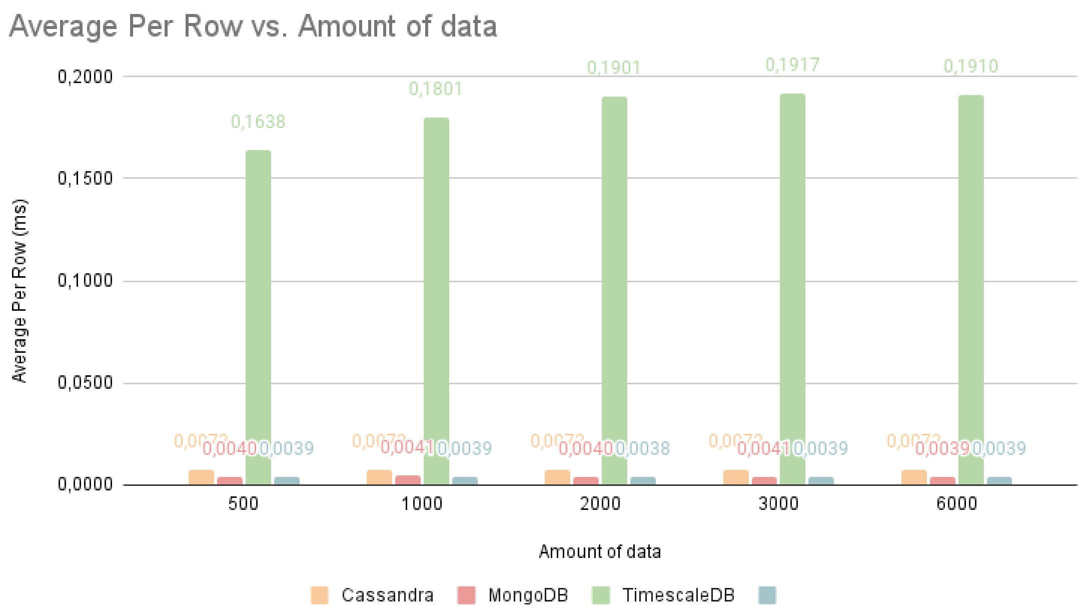

Figure 4.6: Insert time per Row vs Amount of Data with 20ms between insertion

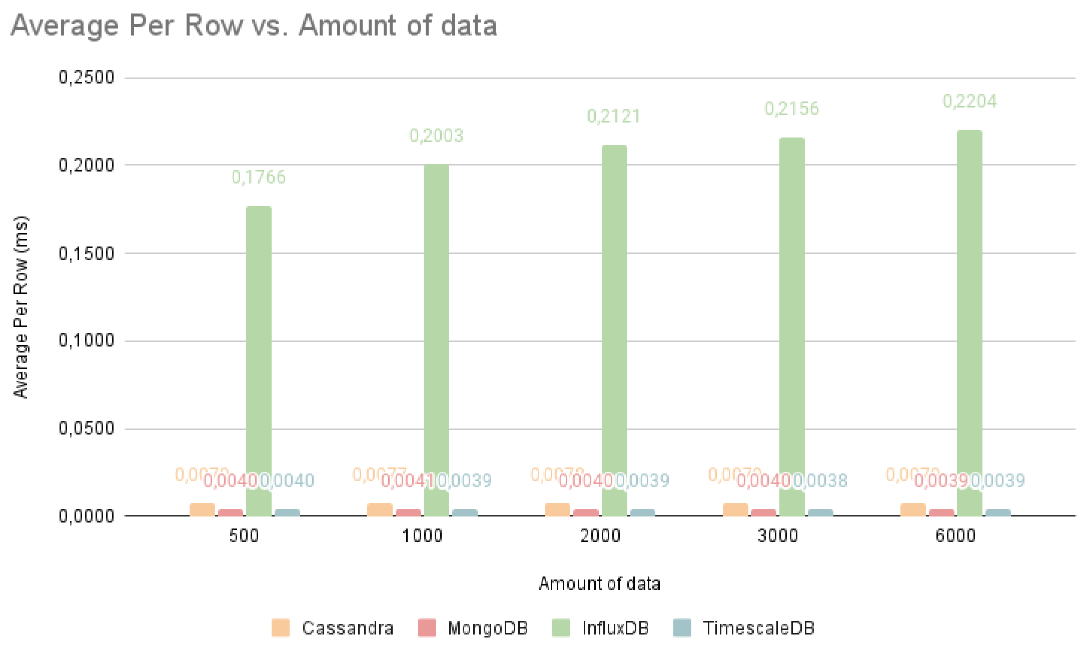

Figure 4.7: Insert time per Row vs Amount of Data with 50ms between insertion

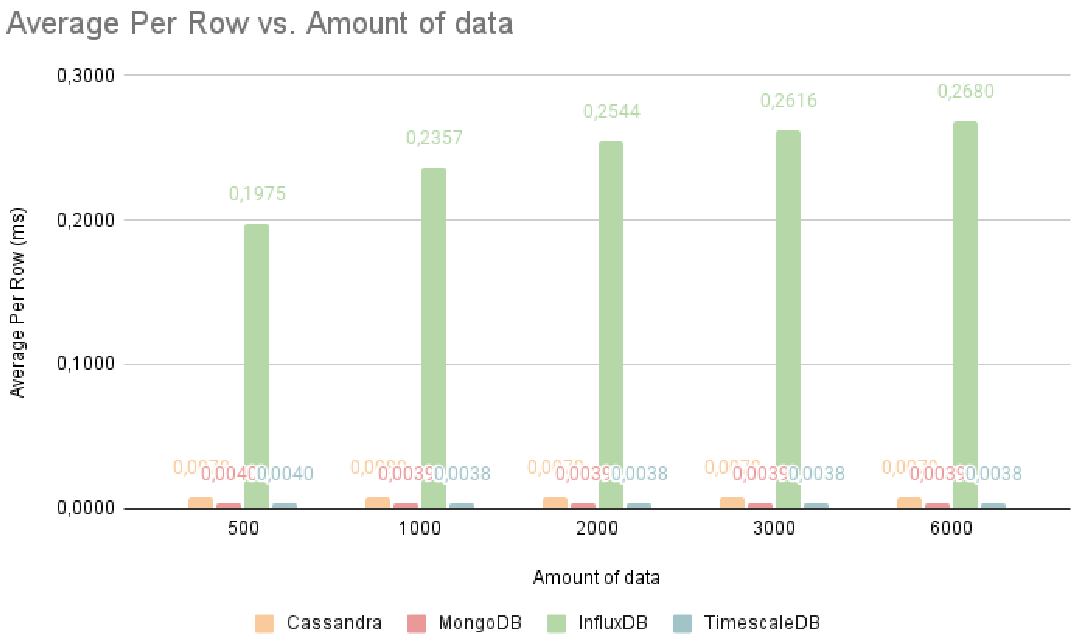

Figure 4.8: Insert time per Row vs Amount of Data with 100ms between insertion

Continuing with the database study, the last three requirements to be achieved are data processing, storage capacity and vertical scalability, thus the plataform must be efficient and fast so that it becomes possible to do multiple operations at the same time, have efficient storage, so that the huge amount of data created by IoT devices can be saved and ensure vertical scalability to support this increase of data.

Time processing remains the same as the data increases, which is an optimal result for systems that produce a large volume of data, with this is also possible to assume that the system is scalable, once the time processing is supportable, despite these guarantees, we may also guarantee that efficient storage capacity.

4.5 Requirements Evaluation

In order to understand if the solution was successful, some tests were carried out to understand if the characteristics and performance of the chosen technology were efficient and if it met the requirements proposed in subsection 3.2.

The most important requirement being related to "Live Data", a study was done to understand and ensure that a minimum capacity latency was achieved. As such efficient data processing and storage should also be ensured as the data produced by IoT devices is always increasing. Last but not least, vertical scalability must be ensured.

CHAPTER 5

Conclusion and Future Work

"When human judgment and big data intersect there are some funny things that happen."

Nate Silver

With the significant progress of the industry and consequently with mass production, the need was felt to match this advance also on the technological side, thus trying to achieve better results.

Although we are still in Industry 4.0, officially since 2013, there have been several technologies that have developed and stood out, one of which was Digital Twin. This dissertation had as main objective to propose and implement a solution of Digital Twins in IIoT, in order to make more efficient and fast, the production process by the industries.

To implement a Digital Twin it is necessary to build a platform that supports the requirements necessary for its implementation. For the construction of the architecture that supports the DT implementation, three fundamental aspects were taken into account, data acquisition, data processing and storage. Despite their different functions, these blocks, together with the DT, has enabled this platform to operate efficiently, and allowed to bring benefits to industries, workers and the environment.

Despite the main focus of this dissertation was the Digital Twin, it was also necessary to build an IIoT platform where it could be implemented.

5.1 Future Work

Despite the implemented solution working and reaching the proposed requirements, in the area of technology there is always room for improvement. During the elaboration of this dissertation, some improvements that are relevant for the development of the Digital Twin technology and allow the evolution of the architecture proposed in this work were identified.

For Future Work, it would be interesting and useful, to implement a visualization service in order to observe the virtual models of each physical object in operation.

One more interesting field of study would be the integration of Fault Prediction Systems into a Digital Twin. It leverages simulation models based on data-driven intelligence to provide insightful clues about how failure would occur and how it might progress, allowing manufacturers to make timely decisions on how to proceed or correct these possible faults in the physical object.

Referências

[1] S. El-Gendy, “Iot based ai and its implementations in industries,” *Proceedings of ICCES 2020 - 2020 15th International Conference on Computer Engineering and Systems*, Dec. 2020. DOI: 10.1109/ICCES51560.2020.9334627.

[2] I. C. Ng and S. Y. Wakenshaw, “The internet-of-things: Review and research directions,” *International Journal of Research in Marketing*, vol. 34, no. 1, pp. 3–21, 2017, ISSN: 0167-8116. DOI: 10.1016/j.ijresmar.2016.11.003.

[3] M. Singh, E. Fuenmayor, E. P. Hinchy, Y. Qiao, N. Murray, and D. Devine, “Digital twin: Origin to future,” *Applied System Innovation*, vol. 4, no. 2, p. 36, 2021.

[4] “Itu-t rec. y.2060 (06/2012) overview of the internet of things.” (), [Online]. Available: https://www.itu.int/rec/T-REC-Y.2060-201206-I. (accessed: 06.30.2022).

[5] K. K. Patel, S. M. Patel, *et al.*, “Internet of things-iot: Definition, characteristics, architecture, enabling technologies, application & future challenges,” *International journal of engineering science and computing*, vol. 6, no. 5, 2016.

[6] N. M. Kumar and P. K. Mallick, “The internet of things: Insights into the building blocks, component interactions, and architecture layers,” *Procedia computer science*, vol. 132, pp. 109–117, 2018.

[7] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, “Industrial internet of things: Challenges, opportunities, and directions,” *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 4724–4734, 2018. DOI: 10.1109/TII.2018.2852491.

[8] M. S. Hossain and G. Muhammad, “Cloud-assisted industrial internet of things (iiot) – enabled framework for health monitoring,” *Computer Networks*, vol. 101, pp. 192–202, 2016.

[9] B. Saenz de Ugarte, A. Artiba, and R. Pellerin, “Manufacturing execution system–a literature review,” *Production planning and control*, vol. 20, no. 6, pp. 525–539, 2009.

[10] X. Xu, “From cloud computing to cloud manufacturing,” *Robotics and computer-integrated manufacturing*, vol. 28, no. 1, pp. 75–86, 2012.

[11] R. Buyya, “Cloud computing: The next revolution in information technology,” pp. 2–3, 2010.

[12] H. Brian, T. Brunschwiler, H. Dill, *et al.*, “Cloud computing,” *Communications of the ACM*, vol. 51, no. 7, pp. 9–11, 2008.

[13] A. Huth and J. Cebula, “The basics of cloud computing,” *United States Computer*, pp. 1–4, 2011.

[14] V. Rajaraman, “Cloud computing,” *Resonance*, vol. 19, no. 3, pp. 242–258, 2014.

[15] Amazon. “O que são clouds públicas, privadas e híbridas?” (), [Online]. Available: https://azure.microsoft.com/pt-pt/overview/what-are-private-public-hybrid-clouds/#overview. (accessed: 02.23.2022).

[16] Amazon. “O que é uma cloud privada?” (), [Online]. Available: https://azure.microsoft.com/pt-pt/overview/what-are-private-public-hybrid-clouds/#private-cloud. (accessed: 02.23.2022).

[17] “Apollo 13: The first digital twin.” (), [Online]. Available: https://blogs.sw.siemens.com/simcenter/apollo-13-the-first-digital-twin/. (accessed: 10.29.2022).

[18] E. Glaessgen and D. Stargel, “The digital twin paradigm for future nasa and us air force vehicles,” 2012.

[19] B. R. Barricelli, E. Casiraghi, and D. Fogli, "A survey on digital twin: Definitions, characteristics, applications, and design implications," *IEEE access*, vol. 7, pp. 167 653–167 671, 2019.

[20] "Apollo 13: The first digital twin." (), [Online]. Available: `https://www.siemens-healthineers.com/news/mso-digital-twin-mater.html`. (accessed: 10.30.2022).

[21] J. Yang, W. Zhang, and Y. Liu, "Subcycle fatigue crack growth mechanism investigation for aluminum alloys and steel," in *54th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics, and materials conference*, 2013, p. 1499.

[22] A. J. Zakrajsek and S. Mall, "The development and use of a digital twin model for tire touchdown health monitoring," 2017.

[23] J. Guo, N. Zhao, L. Sun, and S. Zhang, "Modular based flexible digital twin for factory design," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 3, pp. 1189–1200, 2019.

[24] "Digital twins: Components, use cases, and implementation tips." (), [Online]. Available: `https://www.altexsoft.com/blog/digital-twins/`. (accessed: 10.30.2022).

[25] V. Kuts, G. E. Modoni, T. Otto, *et al.*, "Synchronizing physical factory and its digital twin through an iiot middleware: A case study.," *Proceedings of the Estonian Academy of Sciences*, vol. 68, no. 4, 2019.

[26] D. Knuth. "Digital twins: Components, use cases, and implementation tips." (), [Online]. Available: `https://www.altexsoft.com/blog/digital-twins/`. (accessed: 01.21.2022).

[27] H. Nguyen, K. Tran, X. Zeng, L. Koehl, P. Castagliola, and P. Bruniaux, "Industrial internet of things, big data, and artificial intelligence in the smart factory: A survey and perspective," in *ISSAT International Conference on Data Science in Business, Finance and Industry*, 2019, pp. 72–76.

[28] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[29] J. Wan, J. Liu, and L. Liao, "Guest editorial: Special issue on "advanced artificial intelligence for industrial internet of things"," *Journal of Internet Technology*, vol. 21, no. 5, pp. 1477–1478, 2020.

[30] W. Shi and S. Dustdar, "The promise of edge computing," *Computer*, vol. 49, no. 5, pp. 78–81, 2016.

[31] W. Dai, H. Nishi, V. Vyatkin, V. Huang, Y. Shi, and X. Guan, "Industrial edge computing: Enabling embedded intelligence," *IEEE Industrial Electronics Magazine*, vol. 13, no. 4, pp. 48–56, 2019.

[32] B. Chen, J. Wan, A. Celesti, D. Li, H. Abbas, and Q. Zhang, "Edge computing in iot-based manufacturing," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 103–109, 2018.

[33] I. Sittón-Candanedo, R. Alonso, S. Rodríguez, J. Garcia Coria, and F. De La Prieta, "Edge computing architectures in industry 4.0: A general survey and comparison," in 2020, pp. 121–131.

[34] I. I. Consortium. "The industrial internet of things volume g1: Reference architecture." (), [Online]. Available: `https://www.iiconsortium.org/pdf/IIRA-v1.9.pdf`. (accessed: 02.6.2022).

[35] E. C. Consortium. "Edge computing reference architecture 2.0." (), [Online]. Available: `http://en.ecconsortium.org/Uploads/file/20180328/1522232376480704.pdf`. (accessed: 02.6.2022).

[36] S. C. I. 4.0. "Rami 4.0 in national and international standardization." (), [Online]. Available: `https://www.sci40.com/english/rami4-0/`. (accessed: 02.20.2022).

[37] J. Frysak, C. Kaar, and C. Stary, "Benefits and pitfalls applying rami4. 0," in *2018 IEEE Industrial Cyber-Physical Systems (ICPS)*, IEEE, 2018, pp. 32–37.

[38] P. Adolphs, H. Bedenbender, D. Dirzus, *et al.*, "Reference architecture model industrie 4.0 (rami4. 0)," *ZVEI and VDI, Status report*, 2015.

[39] M. Ha and Y. Shichkina, "Translating a distributed relational database to a document database," *Data Science and Engineering*, vol. 7, no. 2, pp. 136–155, 2022.

[40] M. F. Andreas Bader Oliver Kopp, "Survey and comparison of open source time series databases," in *Datenbanksysteme für Business, Technologie und Web (BTW2017) – Workshopband*, B. Mitschang *et al.*, Eds., ser. Lecture Notes in Informatics (LNI), vol. P-266, Gesellschaft für Informatik e.V. (GI), 2017, pp. 249–268, ISBN: 978-3-88579-660-2.

[41] (), [Online]. Available: `https://www.mongodb.com/compare/relational-vs-non-relational-databases`. (accessed: 10.29.2022).

[42] Y. Qamsane, J. Moyne, M. Toothman, *et al.*, “A methodology to develop and implement digital twin solutions for manufacturing systems,” *IEEE Access*, vol. 9, pp. 44 247–44 265, 2021.

[43] B. A. Talkhestani, T. Jung, B. Lindemann, *et al.*, “An architecture of an intelligent digital twin in a cyber-physical production system,” *at-Automatisierungstechnik*, vol. 67, no. 9, pp. 762–782, 2019.

[44] A. Costantini, G. Di Modica, J. C. Ahouangonou, *et al.*, “Iotwins: Toward implementation of distributed digital twins in industry 4.0 settings,” 2022.

[45] F. J. Lacueva-Perez, S. Hermawati, P. Amoraga, R. Salillas-Martinez, R. D. H. Alonso, and G. Lawson, “Shion: Towards an interactive digital twin supporting shopfloor operations on real time,” *IEEE Internet Computing*, 2020.

[46] S. Manso, “Platform to support the development of iot solutions,” 2019.