Università degli studi di Modena e Reggio Emilia

Dipartimento di Scienze e Metodi dell'ingegneria

Dottorato di Ricerca in

INGEGNERIA DELL'INNOVAZIONE INDUSTRIALE

An IoT Software Architecture for User-Friendly Fault Diagnosis and Identification

---

Architettura Software IoT per la Diagnosi e Identificazione dei Guasti a Misura d'Uomo

Relatore:

Prof. Cesare Fantuzzi

Candidata:

Annalisa Bertoli

Direttore della scuola:

Prof. Franco Zambonelli

Esame Finale anno 2025 - Ciclo XXXVII

« *To my family*»

Sommario

Negli ultimi anni, la complessità dei sistemi produttivi è aumentata significativamente a causa dei progressi nelle tecnologie derivanti dall'Industria 4.0, in particolare attraverso l'Internet of Things (IoT) e i big data. Questa evoluzione ha facilitato l'accesso senza precedenti a enormi quantità di dati, ma ha anche introdotto sfide nella raccolta dei dati e nella loro applicazione pratica per gli operatori che interagiscono con questi sistemi.

Questa tesi presenta un'architettura IoT progettata per ambienti industriali reali, con l'obiettivo di dimostrare come i dati possano essere utilizzati efficacemente per monitorare le operazioni e i processi produttivi in tempo reale. L'approccio proposto migliora la capacità di rilevare e gestire i guasti, fornendo agli operatori le informazioni necessarie per prendere decisioni informate. Integrando sensori intelligenti e analisi avanzate, è possibile ottenere una visibilità dettagliata sullo stato del sistema, consentendo interventi di manutenzione tempestivi e preparando il terreno per future implementazioni di manutenzione predittiva.

La ricerca include un'analisi di due casi di studio distinti, mostrando la versatilità dell'architettura in diverse applicazioni industriali. Illustra come l'utilizzo efficace dei dati possa ottimizzare l'efficienza operativa e ridurre i tempi di inattività, contribuendo così a una migliore gestione del sistema. Inoltre, questo approccio consente agli operatori umani di comprendere meglio i loro ambienti e di prendere decisioni autonome basate su informazioni in tempo reale.

Abstract

In recent years, the complexity of production systems has increased significantly due to advancements in technologies stemming from Industry 4.0, particularly through the Internet of Things (IoT) and big data. This evolution has facilitated unprecedented access to vast amounts of data but has also introduced challenges in data collection and its practical application for operators interacting with these systems.

This thesis presents an IoT architecture tailored for real industrial environments, aimed at demonstrating how data can be effectively utilized to monitor operations and production processes in real-time. The proposed approach enhances the ability to detect and manage failures, providing operators with the necessary information to make informed decisions. By integrating smart sensors and advanced analytics, detailed visibility into system status can be achieved, enabling timely maintenance interventions and paving the way for future predictive maintenance implementations.

The research includes an analysis of two distinct case studies, showcasing the versatility of the architecture across different industrial applications. It illustrates how effective data utilization can optimize operational efficiency and minimize downtime, ultimately contributing to improved system management. Furthermore, this approach empowers human operators to understand their environments better and make autonomous decisions based on real-time insights.

Contents

1 Introduction **1**

2 Industry 4.0 and Industry 5.0 **3**

2.1 Background 3

2.2 Industry 4.0 4

2.3 Industry 5.0 7

2.4 Conclusion 8

3 Internet of Things **11**

3.1 Characteristic 11

3.2 IoT Technologies 11

3.2.1 RFID 12

3.2.2 Bar Code and QR Code 12

3.2.3 Wi-Fi 12

3.2.4 ZigBee 13

3.2.5 Bluetooth 13

3.3 IoT Application Architecture 14

3.4 IoT to support human operators 16

3.5 Conclusion 17

4 Maintenance and monitoring systems **19**

4.1 Maintenance 19

4.1.1 Maintenance strategy 19

4.1.1.1 Corrective Maintenance 19

4.1.1.2 Preventive Maintenance 20

4.1.1.3 Proactive Maintenance 22

4.1.2 Industry 4.0 and maintenance: new opportunities 22

4.2 Monitoring system and Fault diagnosis 22

4.3 Original contribution to fault identification 23

4.4 Conclusion 25

5 Proposed IoT architecture **27**
5.1 Software and communication protocols used . . . 27
5.1.1 MQTT . . . 27
5.1.2 REST API . . . 29
5.1.3 Python . . . 31
5.1.4 Node.js . . . 33
5.2 IoT node architecture . . . 34
5.3 IoT Server architecture . . . 36
5.3.1 HTTPS server . . . 37
5.3.2 Database . . . 38
5.3.3 Web application . . . 40
5.3.4 Notification server . . . 42
5.4 Conclusion . . . 43

6 First use case: AGV application **45**
6.1 System background . . . 45
6.2 System setup . . . 47
6.3 IoT node application . . . 48
6.4 Server application . . . 50
6.4.1 Login page and Registry page . . . 50
6.4.2 Data page . . . 51
6.4.3 Algorithm page . . . 52
6.4.4 Alert page . . . 53
6.4.5 Notification page . . . 54
6.4.6 Status page . . . 55
6.5 Fault analysis . . . 56
6.5.1 Number of the reflectors . . . 57
6.5.2 Number of times the AGV is in “warning” . . . 60
6.5.3 Difference between the set speed and the real speed . . . 62
6.6 Conclusion . . . 64

7 Second use case: MyWelder **65**
7.1 System background . . . 65
7.2 System setup . . . 67
7.3 IoT node application . . . 69
7.4 Server application . . . 73

7.4.1 Alerts page 73
7.4.2 Notification page 75
7.4.3 Historical data page 75
7.4.3.1 Cell history page 75
7.4.3.2 Program history page 77
7.4.3.3 Operations history page 77
7.4.3.4 OEE chart page 78
7.4.4 Live data dashboard page 78
7.5 Fault analysis 79
7.6 Conclusion 83

8 Conclusions **85**

Bibliography **87**

List of Figures

Figure 1 Methodology schema. 24

Figure 2 MQTT broker functioning schema. 28
Figure 3 REST API functioning schema. 30
Figure 4 Generic system architecture for the IoT node. 34
Figure 5 IoT server architecture. 37
Figure 6 Website map of the HMI application. 40

Figure 7 System schema for the IoT node. 48
Figure 8 IoT node system architecture for the AGVs use case. 48
Figure 9 Login Page. 50
Figure 10 Add Graph Modal Dialogue. 51
Figure 11 Historical dashboard Page. 51
Figure 12 Algorithm Page. 52
Figure 13 New Algorithm Modal Dialogue. 52
Figure 14 New Alert Modal Dialogue. 54
Figure 15 New Notification Modal Dialogue. 54
Figure 16 Status Page. 55
Figure 17 Reflector analysis results. 58
Figure 18 Reflector analysis results with industrial plant layout. 59
Figure 19 Reflector analysis with the destination nodes. 60
Figure 20 Mail alarm screenshot. 61
Figure 21 Warning state analysis with the map layout. 62
Figure 22 Speed analysis with the map layout. 63

Figure 23 System photo. 67
Figure 24 Software architecture for the IoT node in the welding application. . . 69
Figure 25 Software architecture implemented in the specific robotic cell. 70
Figure 26 Website map of the HMI application. 73
Figure 27 Alarms page of the web application. 74
Figure 28 Alarms Info button structure. 74

Figure 29 Notification page of the web application. 75
Figure 30 Cell status historical dashboard page of the web application. 76
Figure 31 Program historical dashboard page of the web application. 77
Figure 32 Operations historical dashboard page of the web application. 77
Figure 33 OEE dashboard page of the web application. 78
Figure 34 Live dashboard page of the web application. 79
Figure 35 Command prompt of the execution of simulation step-by-step with singularities real-time detection. 82

List of Tables

Table 1 List of variables acquired. 49

1. Introduction

The rapid technological development of recent decades has profoundly transformed the way we live, work, and communicate. Innovations in fields such as artificial intelligence, robotics, digital communications have created unprecedented opportunities, improving productivity, quality of life, and access to information. However, while these technologies offer outstanding solutions, they also present significant challenges.

Humanity, in fact, has not always been able to keep up with such rapid and complex evolution. A common issue related to the new technological developmen is the fear related to job automation, in particular the idea that operators will one day be replaced by robots [1].

In addition to these challenges, there is an increasing demand for advanced technical skills in the workforce [2], [3]. Professions that once required basic knowledge are now evolving into highly specialized roles that demand expertise in areas such as data science, machine learning, cybersecurity, robotics and in general software development.

The rapid growth of technological innovation has created a gap between the skills needed in today's job market and the abilities of many workers. This skills gap has led to a growing need for continuous education and training in order to acquire the high-level skills required for complex jobs [4].

A key technology developed in the past years is the collection and use of data, thanks to the Internet of Things. It enables the connection of everyday objects to the internet, allowing for continuous data collection, analysis, and real-time decision-making. This has opened up new possibilities in various fields, including healthcare, transportation, and urban planning, where data-driven insights can improve efficiency and outcomes.

Many operators in various industries find themselves struggling to keep up with the rapid growth of technology. Operators may not be familiar with the complex systems, software, or tools involved, making it difficult for them to adapt quickly. As new technologies are integrated into the workplace, these operators often lack the necessary technical skills and knowledge to effectively use them [5].

This skills gap creates a barrier to fully leveraging the potential of new technologies, ultimately limiting their benefits. While these technologies are designed to improve efficiency and productivity, the lack of proper training and expertise can hinder their successful implementation. Without adequate support and training, many operators feel overwhelmed and unsure of how to navigate these innovations, leading to reduced effectiveness and increased frustration.

This thesis will propose an Internet of Things architecture capable of being implemented in real industrial environments and explain how this architecture can support operators in system analysis, management, and reacting when faults occur. The main goal of this work is to provide operators with a tool that empowers them to work independently and make the most of the new opportunities brought by technological innovation, particularly in the field of data collection.

To validate the proposed Internet of Things architecture, two use cases are considered. In the first case, the architecture is implemented with Automatic Guided Vehicles, and in the second use case, it is applied in a collaborative robotic cell. In both use cases, some faults are identified and prevented from occurring.

The structure of the thesis is as follows: in Section 2, Industry 4.0 and Industry 5.0 are analyzed. Then, in Section 3, the Internet of Things technology is presented, while Section 4 describes fault analysis and monitoring system technologies. The proposed software architecture is presented in Section 5, while the first use case is described in Section 6 and the second in Section 7. The conclusions are presented in Section 8.

2. Industry 4.0 and Industry 5.0

In this section, some basic concepts regarding Industry 4.0 and Industry 5.0 are introduced, in order to help the reader understand the environment where this research takes place and the objective of the original contribution presented in this work.

2.1 Background

When the development of new technologies transforms productive and social structures, a new industrial revolution is underway. During the last centuries, manufacturing systems have experienced main changes in their domain. The main differences between the different industrial revolutions are here reported [6]:

1. First Industrial Revolution

   The mechanization through water and steam power created the first manufacturing process mostly in the textile industry; there was a transition from farming and feudal society to a new manufacturing process [7]. The first industrial revolution improved the quality of life of the operators.

2. Second Industrial Revolution

   The discovery of the internal combustion engine that allowed the electrification of the industries allowed mass production in assembly lines; with that mass production became possible without product customization.

3. Third Industrial Revolution

   In the last century, thanks to information technology and the implementation of electronics automation was reached; in the manufacturing field this allowed flexible production in a variety production on flexible production lines with programmable machines; this did not allow flexibility in terms of quantity [8].

4. Fourth Industrial Revolution

   It is called Industry 4.0, it refers to the intelligent networking of machines and pro-

cesses in the industrial environment based on Cyber-Physical Systems (CPS) to achieve intelligent control [9].

5. Fifth Industrial Revolution.

   The Fifth Industrial Revolution represents a transformative shift in the paradigms of economic and technological development. It is characterized by a profound integration between human capabilities and machine intelligence, this revolution emphasizes sustainability, social well-being, and ethical considerations in production and consumption [10].

Industry 4.0 is triggered by the increased development of information and communications technologies (ICT) [8], while Industry 5.0 has the scope to overcome the main limitation of Industry 4.0. The Fifth Industrial Revolution is poised to reshape the future by balancing technological advancement with human values. By integrating AI, promoting sustainability, personalizing experiences, fostering collaboration, and encouraging social innovation, this revolution aims to create a more inclusive and sustainable world. It challenges us to rethink our relationship with technology and to consider the broader implications of progress, ultimately striving for a future where both humanity and the planet thrive.

2.2 Industry 4.0

In 2011, during the Hannover Fair, Henning Kagermann, Wolf-Dieter Lukas and Wolfgang Wahlster used for the first time the term Industry 4.0 (Industrie 4.0 in German language) and later the term was used in a project in the high-tech strategy of the german government. Later the term became globally adopted in the past decade [9]. Nine technologies are the building blocks of Industry 4.0. The scope is to transform production cells that are isolated into an optimized production system where those production cells are integrated and automated, to achieve a greater efficiency [11]. Those nine pillars can realize the Industry 4.0 vision; they are described as the enabling technologies of Industry 4.0 [12], [13]:

1. Big Data and Analytics

   Big data are described by four dimensions called the 4Vs: Volume, Variety, Velocity and Value. Volume refers to the quantity of data collected, Variety refers to the different kinds of data and sources, Velocity aims for fast acquisition and Value underlines the fact that a fast analysis of this data can support a real-time decision-making process.

Some authors consider the Veracity the fifth V, because the Veraciry refers to the accuracy of the data [14]. Big Data Analytics aims to analyse these heterogeneous datasets and find some information, such as trends, customer habits, the correlation between some metrics, or other information to improve the industry business and help in the strategy plan.

2. Autonomous Robots

   Robots are used to do the activities that humans cannot do easily. With Industry 4.0 the use of robots has increased in number and in different areas. The abilities expected in autonomous robots are to finish the given task, without forgetting the time given for the task, safety for humans and the environment in which the robots work, flexibility, versatility and collaboration with humans and other robots.

3. Simulation

   Simulation has the capability of creating more scenarios using a virtual model; everything can be part of a simulation, such as humans, machines, robots, and entire plants. Simulations can help during the decision-making process but also can reduce the issues during start-up phases and decrease the downtime. Even a strategic plan can be simulated, using real-time data.

4. System Integration: Horizontal and Vertical System Integration

   In [12] there are three dimensions in the integration process wanted by Industry 4.0: "horizontal integration across the entire value creation network, vertical integration and networked manufacturing systems, and end-to-end engineering across the entire product life cycle". Horizontal integration aims to integrate partners, while vertical integration has the objective to reach a flexible and reconfigurable system.

5. The Internet of Things

   The concept Internet of Things (IoT) refers to a network of interconnected devices that communicate via standard protocols. In this network, different kinds of elements can be connected, such as smart sensors, machines, physical objects, production processes and production lines. Data and software are important in this architecture. IoT systems have three main pillars in their application; the first focus is on process optimization; the second target is the optimization of resource consumption; the last scope is the creation of complex and autonomous systems [15].

6. Cyber security

   With Industry 4.0 there is an increase in connectivity, that requires an increase in protection with cyber security. CPS are systems where humans and physical objects are tightly integrated with computation, communication, and control systems. It is very important to guarantee not only data safety, but also environment and human safety in this kind of system where everything is connected.

7. The Cloud

   Data sharing between different sites and companies becomes crucial with Industry 4.0, and this process must be fast. The connection and the communication are possible thanks to cloud-based IT platforms. Devices, machines, an entire plant can be connected to the cloud and share information between all the elements in the system. Cloud Computing can be split in three models: Software as a Service, Platform as a Service and Infrastructure as a Service.

8. Additive Manufacturing

   Additive Manufacturing is one of the known technologies of Industry 4.0 [8]. Additive Manufacturing gives the possibility to produce a small batch of customized products following the Industry 4.0 vision of the client's need. It reduces the time to market and with this technology is possible to produce complex products with request. It is useful for prototypes and many materials can be used.

9. Augmented reality

   In [13], augmented reality is defined by Erboz "as the interactive technology that enables harmony between the virtual world and its users while the virtual world is being used as the part of the real surroundings". With this technology, the interaction between humans and machines is enhanced; in fact, augmented-reality-based systems can offer a variety of services that help the operators in their tasks. Workers can easily have access to information, know how to repair something, be helped in the maintenance task, and be helped during the decision-making process thanks to this tool.

The rapid advent of Industry 4.0 technologies in industrial applications affects the operator's role in the work system. The term Operator 4.0 defines the operator that is assisted by automated systems that can provide a sustainable relief of both physical and mental stress

[16]. Operator 4.0 concept highlights the idea of an operator that interacts with machines in a trusting and interaction based relationship; the scope is to empower the operator with the new acquired skills and increase the opportunities of Industry 4.0 [17]. The scope is to enable the operator to improve its capabilities and skills in managing manufacturing complexity [18].

2.3 Industry 5.0

Ten years after introducing Industry 4.0, Industry 5.0 was announced by the European Commission, and the scope is to respond to emerging societal challenges [19]. Industry 5.0 aims to overcome the limitations of Industry 4.0 by including the new society and operator 5.0 [18]. The idea of Industry 5.0 is to promote a paradigm where the system is flexible and resilient thanks to the use of flexible and adaptable technologies [20]. The focus is not only on the technological progress point of view as in Industry 4.0, but it is considered with a special attention to human centrality [21]. Industry 5.0 leverages on those aspects:

- Integration of Artificial Intelligence

  The key concept of the Fifth Industrial Revolution is the advanced application of Artificial Intelligence (AI). Unlike previous revolutions, which primarily focused on automation for efficiency, this phase leverages AI to enhance not only productivity but also quality of life. By enabling machines to perform complex tasks, organizations can redirect human intelligence towards strategic decision-making, fostering innovation and creativity. For this reason, a key activity is to increase the knowledge about human-technology interaction, with the goal to fill the gap between production and societal needs [22]

- Sustainability

  A defining feature of the Fifth Industrial Revolution is its strong emphasis on environmental sustainability. As global challenges like climate change and resource depletion intensify, businesses are increasingly held accountable for their ecological impact [23]. This revolution encourages the adoption of sustainable practices, aiming to create a circular economy where waste is minimized and resources are reused.

- Personalization and Humanization

  In contrast to the mass production models of the past, the Fifth Industrial Revolution

advocates for personalized and human-centered products and services. Through advanced data analytics and AI, businesses can tailor offerings to meet individual needs, enhancing user satisfaction and fostering a deeper connection between consumers and brands. This shift towards personalization underscores the importance of considering human well-being in all aspects of production

- Human-Machine Collaboration

  The Fifth Industrial Revolution promotes a synergistic relationship between humans and machines [24]. This collaboration enables humans to focus on tasks that require critical thinking, empathy, and creativity, while machines handle repetitive, dangerous, or mundane tasks. Such a model not only enhances productivity but also allows for a more fulfilling work experience for individuals.

- Social Innovation

  Beyond technological advancements, the Fifth Industrial Revolution prioritizes social innovation as a means to address pressing global challenges [23]. Issues such as inequality, access to education, and healthcare are integral to this new paradigm. By fostering innovative solutions that promote social equity, this revolution seeks to ensure that progress benefits all members of society [25].

In this scenario there is a co-existence between Industry 4.0 and Industry 5.0. The main difference between the last two Industrial revolutions is that Industry 4.0 is considered technology-driven, while Industry 5.0 is considered value-driven [9]. In this prospective, the operator becomes not just a technology-assisted operator as in the 4.0 operator vision, but a Resilient Operator 5.0 who focuses both on human resilience, including health and safety, and on system resilience, which involves the exchange of control between the operator and the machine to ensure the continuous functioning of the system [26]. Industry 5.0 is not just a technological evolution, but a new vision that integrates artificial intelligence, advanced robotics, and other emerging technologies, with a strong focus on human well-being, sustainability, and ethics.

2.4 Conclusion

Industry 4.0 integrates smart technologies like big data, autonomous robots, IoT, and additive manufacturing to optimize production and automation, while Industry 5.0 aims to

overcome the limitations of Industry 4.0 by prioritizing human-machine collaboration, sustainability, personalization, and social innovation. It emphasizes human well-being, ethical production, and the integration of AI and advanced robotics.

Industry 5.0 is value-driven, focusing on human well-being, ethics, and sustainability, in contrast to Industry 4.0's technology-driven approach. The objective of this research is to present a fault methodology that can be easily implemented by non expert-operator thanks to the proposed industrial IoT architecture.

3. Internet of Things

As previously introduced, IoT is one of the nine pillars in the Industry 4.0 revolution. This sectionon aims to describe the key points of the IoT technology.

3.1 Characteristic

IoT refers to a network of physical objects that are embedded with software, hardware, sensors, etc.; those objects do not need to be the same devices, but it is possible to connect different typologies of 'things' [12]. IoT allows connectivity and enables the object to collect data [27] and connect physical and digital spheres, considering the distributed nature of modern production lines. Thanks to this architecture, a physical system receives a digital ID; it must be unique in the same network, to allow the device recognition in the network [28]. Building an IoT application normally requires the integration with different technologies, such as Wireless Sensor Networks, Radio Frequency Identification, and Machine-to-Machine (M2M) communication [29].

IoT can be used in Industrial applications or not; when it is used in an industrial environment, is normally called Industrial IoT (IIOT). The advantage of IIoT adoptions is the increase in efficiency and productivity through the possibility of having smart and remote management of the system [30]. A further advantage of IoT technologies is the ability to easily manage reconfigurations and interoperability through the software without giving up the tight constraints of many industrial systems [31].

3.2 IoT Technologies

Suresh in [32] describes five wireless communication standards that are used globally to implement an IoT system with some notes regarding the IoT scenario, even if there are other standards used. Those technologies are:

1. RFID.

2. Bar Code and QR Code.
3. Wi-Fi.
4. ZigBee.
5. Bluethooth.

3.2.1 RFID

With RFID technology, an RFID tag is attached to every single object, that can be automatically identified. RFID tags are read and in this way, they are monitored and tracked, using radio waves [33]. This technology has the advantage of being cheap because only RFID tags, RFID readers backend signal processing and IT infrastructure are required. The RFID readers query the RFID tags that have a small microchip that stores data and processes information. There are two types of tags: passive and active. The passive tags can have energy from the reader's radio waves and they can only answer them with the answer requested. The active tags have an embedded power source; for them, it's possible to operate at a farther distance from the reader than the passive ones. This technology finds its main application in manufacturing enterprise operations [28].

3.2.2 Bar Code and QR Code

A barcode is a symbol placed on every object that can be read by a barcode scanner. This technology is used worldwide because it is simple to implement. A QR code is a 2D matrix representation of a barcode; the QR code encodes information on a printable support, in the grid of black and white square dots [34]. The phone camera is able to read the QR code and obtain information regarding the specific object. The main difference between RFID and a barcode is that a barcode is a sticker that can be attached to any product, whereas an RFID is a chip that stores information about that item. The barcode technology is less expensive than the RFID.

3.2.3 Wi-Fi

Wi-Fi allows you to connect to the internet and communicate wirelessly using radio waves providing high-speed internet access over short to medium distances. It can support high-speed data transfer, making it suitable for activities like streaming video, gaming, and down-

loading large files. Modern Wi-Fi routers can support many devices connected simultaneously. Wi-Fi can be power-demanding, and it is not the best solution for battery-powered IoT devices like sensors or wearable devices. For these applications, ZigBee, Bluetooth other low-power protocols may be more suitable.

3.2.4 ZigBee

ZigBee is based on IEEE 802.15.4 – 2003 WPAN standard [35]. It is considered a Low Power Consumption standard because Zigbee devices are designed to be energy-efficient. It is used to develop different applications, and some example are home energy monitors or wireless light switches. Zigbee aims to be simpler and less expensive that Bluethooth or Wi-Fi. Zigbee supports mesh networking, which means devices can communicate with each other directly or relay data through other devices, improving coverage and reliability. It works in a short range, depending on the environment and device settings. Zigbee chips are embedded with radios and with microcontrollers and they communicates using radio bands. Zigbee is an open standard protocol managed by the Connectivity Standards Alliance[1].

3.2.5 Bluetooth

Bluetooth[2] allows devices to exchange data over short distances and this technology is based on radio waves, and for this reason, it may not be suitable for longer distances or large networks. It was designed for short-range communication [36]. It uses encryption and authentication mechanisms to ensure secure communication between devices. Bluetooth is energy-efficient, especially in low-power modes like Bluetooth Low Energy (BLE), making it ideal for battery-operated devices. A possible limitation depending on the specific system could be the data transfer speed, because Bluetooth may not be the best solution for high-bandwidth applications compared to Wi-Fi.

One application developed by me and other researchers is described in [37]. In this work, BLE technology is used to implement an IoT architecture with the aim of collecting small amounts of data from different sources over a long period. The system includes many BLE devices, each with its own battery, and a single gateway, which is the only object powered by electricity. The gateway has both BLE communication protocol and a Wi-Fi module. To

[1] https://csa-iot.org
[2] https://www.bluetooth.com

extend the battery life of individual devices and reduce their consumption, a sleep procedure is implemented that involves all the BLE devices.

The BLE devices can be embedded with sensors to monitor a relevant variable in a specific environment. The goal is to assist the operator in monitoring the system, particularly in large structures, such as healthcare facilities or industrial plants. To preserve the battery life of the devices, a cycle time is used with two phases: the first is the active phase, and the second is the deep sleep phase. During the active phase, the devices turn on and transmit their sensor data to the central node. At the end of the active phase, they enter standby mode to reduce battery consumption. In the network, they send both the sensor data and their battery status.

The gateway can then communicate using the Wi-Fi communication protocol. It can send the collected data to a cloud server, allowing the data to be stored, accessible to the user from anywhere, and enabling the user to manage the system.

The proposed architecture aims to achieve fault tolerance, ensuring that the data can still be sent through the network, reaching the available BLE nodes. This architecture can be implemented to monitor objects that do not require large amounts of data to be sent, thus supporting the operator in system maintenance.

3.3 IoT Application Architecture

Developing an Internet of Things (IoT) application necessitates the integration of various technologies. Central to this architecture is data and software, which play a key role in ensuring functionality and effectiveness [12].

One of the significant advantages of IoT is its capacity to connect smaller devices to the internet, facilitating real-time communication between physical objects and the digital realm. This connectivity streamlines the management of complex production lines within distributed systems. Each computerized system generates data that can be collected and analyzed, providing valuable insights for decision-making.

Furthermore, many systems automatically produce alerts that assist in monitoring their operational health. Currently, there is no universally accepted architecture for IoT applications [38], which can affect both development and widespread adoption [39].

At the beginning of the research activity, the three-layer architecture was introduced [29],

[40]. The three-layer architecture includes:

- Perception layer: this layer connects the phisical word with the real one and it consists of sensor networks responsible for data collection.
- Transmission layer: this layer facilitates the transmission of data from sensors to subsequent levels.
- Application layer: this uppermost layer provides the user interface for interaction with the system.

Soumyalatha in [41] proposes a four-layer architecture that considers:

- Perception layer.
- Gateway and network layer: it is the layer responsible for transferring the information from the sensors to the next layer.
- Management service layer: it is an interface between the two layers.
- Application layer.

Comparing this with the beginning architecture, the Transmission layer is split into two different layers.

Mouha in [40] reports an architecture of five-layer, which are:

- Perception layer.
- Application layer.
- Processing layer: it processes the information received from the network layer in order to have information available and make decisions.
- Transport layer: it transmits data from the perception layer to the processing layer and over networks.
- Business layer: it visualizes information and statistics from the application layer.

The common point of those architectures is that they consider the core elements to enable effective data management and user engagement.

3.4 IoT to support human operators

The advent of IoT has transformed numerous sectors, offering new opportunities to improve operational efficiency, optimize resources, and enhance decision-making processes. One significant area where IoT has shown remarkable potential is in supporting operators across various industries. Operators, in this context, refer to individuals or systems that manage, monitor, and control operations in sectors such as manufacturing, transportation, logistics, and utilities.

IoT-enabled remote monitoring has transformed the way operators manage operations. Where it is implemented, operators no longer need to be physically present at the site to monitor equipment or systems. For instance, IoT sensors and devices can continuously collect data on critical parameters of interest. This data is transmitted to central monitoring systems, which operators can access remotely. Such capabilities not only enhance safety by minimizing the need for on-site personnel but also provide operators with the flexibility to respond quickly to emergencies from any location.

IoT technologies have enhanced operational efficiency by providing operators with real-time data from connected devices [42]. According to many studies, real-time monitoring through IoT systems enables operators to track equipment performance, detect malfunctions, and predict failures before they occur. In manufacturing, IoT sensors on machines help track parameters like temperature, vibration, pressure, etc., which operators can use to optimize maintenance schedules and prevent unplanned downtime [43]. Furthermore, IoT-driven automation and smart devices can reduce human intervention in routine tasks, allowing operators to focus on more strategic decisions. The integration of Artificial Intelligence and Machine Learning with IoT can help operators predict operational trends and make informed decisions [44].

Another applications of IoT to support operators is in predictive maintenance [45]. By gathering data from connected equipment and using analytics tools, IoT systems can provide operators with early warnings about potential equipment failures or irregularities. This enables operators to take corrective action before issues escalate, reducing costly downtime and increasing the lifespan of machinery. Moreover, IoT systems can provide operators with decision support tools that leverage big data and analytics. These systems aggregate vast amounts of data from different sources (e.g., sensors, historical performance data) and

present it in actionable insights. As a result, operators can make faster, more accurate decisions, thereby improving overall system performance.

There is much research focused on IoT applications to support operators. Rosati in [46] uses a IoT system integrated with Building Information Modeling to monitor air quality parameters in buildings. In [47], an IIoT monitoring solution is presented to support advanced predictive maintenance applications, and it is tested in a real industrial scenario. Ruppert, in [48], proposed a review of IIoT infrastructure to improve the Operator 4.0 concept; both the architecture and IIoT infrastructure technologies were considered in the study. In [49], it is reported that there are IoT applications in industrial environments, even if they are not the primary focus of researchers.

All these studies are very interesting, but there is a lack of focus in the literature on non-expert operators in new technologies in industrial scenarios. This thesis aims to propose a solution in this direction.

3.5 Conclusion

IoT refers to the network of physical objects or "Things" equipped with sensors, software, and other technologies; the main goal is to be connected with other "Things" and exchange data with other devices and systems over the internet or other communication networks. The section describes some wireless technologies used in the IoT architectures; the use of a specific wireless network protocol instead of another depends on the specific problem domain. There is no universal architecture for all IoT applications, and this fact can slow down IoT growth.

4. Maintenance and monitoring systems

This section describes the maintenance strategy and the traditional approach fault diagnosis; the last part shows the methodology implemented in this work to diagnose the faults in industrial system, and that is applied in the proposed use cases.

4.1 Maintenance

Maintenance activities, particularly in industrial environments, want to keep something in proper condition, with the scope of reducing failures, decreasing downtime, maximising system availability, minimising breakdown, and maximising system reliability; sometimes maintenance is a multi-object function of the previous ones [50].

4.1.1 Maintenance strategy

Maintenance is an expensive activity from the industrial point of view, in fact maintenance costs are between 15% and 60% of the total operating costs of production [51], [52]. For this reason, the maintenance strategy implemented needs to be fitted in the industrial field where it is applied. Moreover, in the same industrial plant or in the same production line the maintenance strategy can be different for two different machines depending on their requirements. Now the maintenance strategies are described in order to understand the problem's difficulty.

4.1.1.1 Corrective Maintenance

Corrective Maintenance (CM) is performed when a breakdown appears. It is the less expensive strategy because the full life of the specific component is used, but it is not a good practice when the failure consequence is severe [53]. For this reason, CM is the best one when there are cheap components [54] that are easy to replace and do not require a long system storm, or when the costs for spare parts are very demanding and the downtime maintenance costs are very high [51]. CM can be split into:

1. Emergency Maintenance

   Emergency maintenance is the maintenance that occurs when the systems need immediate maintenance actions in order to not stop production. When it occurs, it requires priority over all the other scheduled actions. It is the most expensive maintenance due to its nature; when it appears it stops the production; for this reason, the companies must reduce the emergency maintenance actions [55]. Many stops can be avoided thanks to scheduled maintenance actions.

2. Deferred Maintenance

   The term deferred maintenance includes all the maintenance actions that are not performed immediately due to some limits such as unavailability, budget, lack of time, etc, even if the failure is evident. For this reason, these maintenance actions are scheduled in the future [56]. It is important to underline that some saving costs can be reached immediately but it can increase the deterioration of the assets and the future maintenance and repairs costs.

4.1.1.2 Preventive Maintenance

Preventive Maintenance (PM) aims to prevent system failures by proactively addressing potential issues before they occur, using the mean-time-to-failure (MTTF) parameter as a key indicator [51]. By employing PM, the likelihood of system breakdowns is significantly reduced, as maintenance activities are scheduled to replace or repair components before they fail. Moreover, regular maintenance prolongs the expected life of the assets [57].

However, the main challenge associated with this approach is its potential lack of cost-effectiveness, as maintenance is performed even when the components are still functioning properly, which may lead to unnecessary expenditures [54].

It is possible to define different techniques in the PM field, and some of them are:

1. Time-Based Maintenance

   Time-Based Maintenance (TMB) is a traditional maintenance technique that schedules determined maintenance actions over time based on failure time analyses [58]. The basic assumption of TBM is that the failure of the assets is predictable.

2. Failure Finding Maintenance

The goal of Failure Finding Maintenance (FFM) is to detect hidden failures. Normally the routine inspections do not find these faults, and the equipment seems working until something occurs [59]. FFM are normally implemented with security and protective functions, such as pressure safety valves, and trip transmitters [55]; that kind of equipment is not used until other faults occur that require their intervention. The maintenance is normally performed based on fixed time intervals.

3. Risk Based Maintenance

   Risk Based Maintenance (RBM) is developed to assess the failure risks of all system components. The high-risk components are then taken into account, as the goal is to achieve tolerable risk criteria. To accomplish this, a higher maintenance frequency is required [60].

4. Condition Based Maintenance

   Condition Based Maintenance (CBM) prefers maintenance actions based on the data collected thanks to a condition monitoring process. The target is to monitor the equipment life in its operating condition, to maximize the asset life duration. In this scenario, when a specific threshold of a specific variable is exceeded, the maintenance is performed. The measures are done based on specific monitoring parameters, and some examples are vibration, temperature, and noise levels [58].

5. Predictive maintenance

   Predictive maintenance (PdM) is the most complex maintenance strategy to implement because it predicts failures with the scope to optimize the maintenance efforts [61]. PdM, estimates the remaining useful life (RUL) of the system [45] and maximize the component utilization, based on the entire system status. PdM can be considered an extension of CBM. It is the most expensive maintenance technique because it requires data, analysis and time.

It is important to underline that there is not a clear distinction between the presented techniques. In the same environment, more techniques can be implemented, or for example, the RBM can be implemented by monitoring some parameters, and the result is a technique that is a mix between the RBM and the CMS.

4.1.1.3 Proactive Maintenance

Where potential failures are identified early, it becomes possible to implement a preventive maintenance strategy. The goal is to reduce unplanned downtime and increase the useful life of assets by minimizing asset deterioration. This type of maintenance is applied not only before a failure occurs but also at the beginning of an asset's life, to preserve the health of its components [54].

4.1.2 Industry 4.0 and maintenance: new opportunities

Today, maintenance practices can take advantage of the technological advancements driving the Industry 4.0 revolution [62]; the interconnection between Industry 4.0 and PdM is underlined for the fact that PdM is also called Maintenance 4.0 [63]. To successfully implement a PdM strategy, it is crucial to gather data from the system that can accurately predict the health status of the equipment. The advent of Industry 4.0, coupled with the widespread use of the IoT [45] and Machine Learning, has enabled the practical application of PdM.

IoT plays a key role in connecting various devices through the internet, providing useful real-time data for system monitoring [64]. The ability to collect vast amounts of data opens up numerous opportunities for analysis, including the critical area of maintenance analysis, allowing for more informed decision-making and proactive maintenance strategies. Moreover, PdM leverages advanced Machine Learning techniques to prevent premature and costly equipment breakdowns, ensuring that timely repairs are made before a failure occurs [65].

4.2 Monitoring system and Fault diagnosis

To be able to monitor the asset's health status, a monitoring system is required.

The first way to implement a monitoring system is to do manual machine monitoring. The inspections can be visual inspections or instrument inspections [50]. In visual inspections, there is a periodic physical inspection, and maintenance leverages on inspectors' expertise. With instrument inspections, the analysis is done using both the inspector's expertise and instrument results. This process is implemented mainly by companies that only implement time-based maintenance. Probably the use of Information Technology (IT) systems is minimal, typically limited to scheduling and recording inspection feedback through spreadsheets

and notes. Sensors and advanced control systems are not employed, and assets are regularly inspected manually by checking the machines. Without integration with IT systems, this strategy is not very flexible.

The other possibility is to implement machine health monitoring with advanced IT integration. This second approach monitors the machine's health status over time, and data are continuously tracked. This data is collected and processed by software solutions to alert technicians in case of anomalies and assist management in planning maintenance activities. These software solutions enable companies to manage maintenance activities effectively, aiming to prevent faults.

According to [45], to support PdM is important to design a proper IoT infrastructure. Thanks to Industry 4.0 a large amount of data is now available that can be useful, but at the same time it is important to underline that acquiring, storing, maintaining and analysing the data has a high cost, and it increases with the amount of data; the monitoring system costs should not overcome the system benefits.

Implementing a fault diagnosis system requires a predictive maintenance architecture, which is described in [51]. The main functionalities are:

- Data acquisition.
- Data Analysis.
- State Detection.
- Healt Assessment and Prognosis.
- Maintenance Actions and Alerts.

4.3 Original contribution to fault identification

Now the proposed methodology to be able to diagnose failures, by monitoring some variables, is described. The methodology schema is available in Figure 1. To implement this methodology, a good knowledge of the system is required, otherwise, the possibility of forgetting some aspects increases, and the output of the analysis is not enough to understand the system's behaviour. The four steps are:

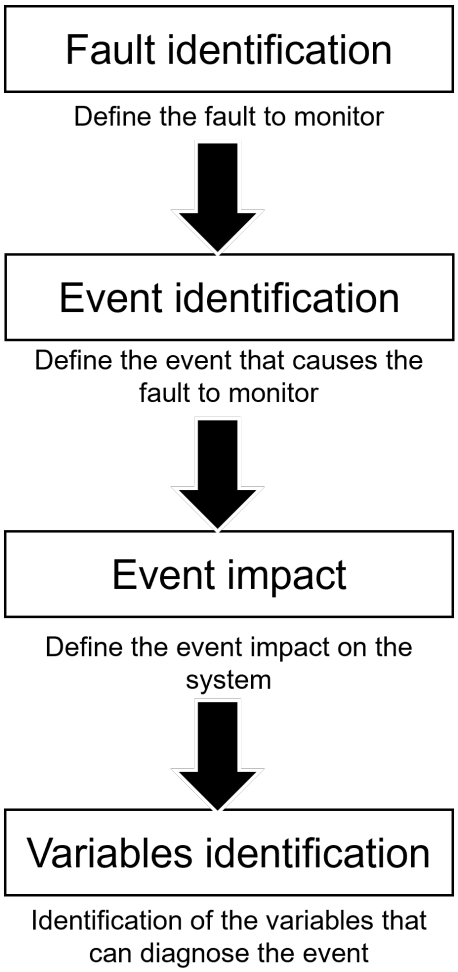


Figure 1: Methodology schema.

1. Fault identification

   The first step to implement this methodology requires the Fault identification. It is not possible to consider all the possible faults altogether in this kind of study, and for this reason, the methodology requires the identification of a specific fault.

2. Event identification

   Then it is possible that the fault is not caused only by a specific event, but more events can cause the same fault. For this reason, it is important to determine all the events that cause the fault.

3. Event impact

   For all the events it is necessary to understand the impact on the system. The fault impact on the system depends on the specific event that causes the system failure. Knowing which event occurred simplify to understand the corrective action to imple-

ment in order to restore the system.

4. Variables identification

   Last it is important to define a variable or a set of variables to monitor in order to monitor the event. In this way, it is possible to prevent failure or at least be able to respond promptly when a system failure occurs.

4.4 Conclusion

This section analyzes the importance of maintenance and system monitoring in preventing failures and downtime. The main maintenance techniques have been introduced to help the reader understand the impact of maintenance and its use on the system. Fault diagnostics remain a critical issue, and research aims to provide solutions to industrial problems. The proposed methodology tries to contribute in this direction. In the use cases in section 6 and section 7 the methodology will be applied to better understand its potential.

5. Proposed IoT architecture

In this section, the implemented architecture is described. Some of the aspects were published in [66]. Two different elements compose the system, that are the IoT node and the server. In this document, the term IoT node defines any industrial element that collects data interesting for the user that is generated in the system. In the same system, it is possible to have more IoT nodes that collect the data and send them to one server, and for this reason, different ID are given to different nodes. This architecture allows the user to have a unique access point to the IoT nodes' data. The following chapter introduces the used software and communication protocols and then describes the system software architecture.

5.1 Software and communication protocols used

5.1.1 MQTT

Message Queue Telemetry Transport[1] (MQTT) is an ISO[2] standard protocol (ISO/IEC PRF 20922). It is a lightweight messaging protocol designed for efficient communication between devices in low-bandwidth, high-latency, or unreliable networks. It is particularly well-suited for Internet of Things (IoT) applications where devices need to send and receive small amounts of data. The data transmieted can be in various forms such as binary data, text, XML, or JSON [67]. It is a versatile and efficient protocol that facilitates reliable communication in IoT and other applications where resources may be limited or networks may be unstable. Some key aspects describes in [68] are:

- Publish/Subscribe model: MQTT uses a publish/subscribe model, where devices (clients) can publish messages to a topic and the devices that subscribe to the topic receive messages from that topic. This decouples the message producers from message consumers, allowing for flexible communication.
- Lightweight and efficient: The protocol is designed to minimize overhead, making it

[1] https://mqtt.org
[2] https://www.iso.org/standard/69466.html

ideal for resource-constrained devices. It uses a small packet size, which reduces the amount of data transmitted.

- Retained messages: A message can be retained on the broker, so when a new client subscribes to a topic, it immediately receives the last retained message.
- Last will message : MQTT allows clients to specify a "last will" message that is sent by the broker if the client unexpectedly disconnects, providing a way to notify other clients of a device's status.
- Quality of Service levels: MQTT supports three levels of Quality of Service (QoS) for message delivery:
  - QoS 0: At most once (fire and forget).
  - QoS 1: At least once (ensures message delivery).
  - QoS 2: Exactly once (ensures that messages are delivered once and only once).

MQTT is commonly used in various applications, including smart home devices, for communication between sensors, lights, and hubs [69].It is used in Industrial IoT application to monitor and control equipment and processes [70]. In healthcare is used to monitor patient and transmit their data [71].

An element that must be present is the MQTT broker. An MQTT broker is a server that facilitates communication between MQTT clients by managing message distribution. It plays a central role in the publish/subscribe model of MQTT, allowing devices to communicate without being directly connected to each other [72]. The MQTT broker schema is represented in Figure 2.

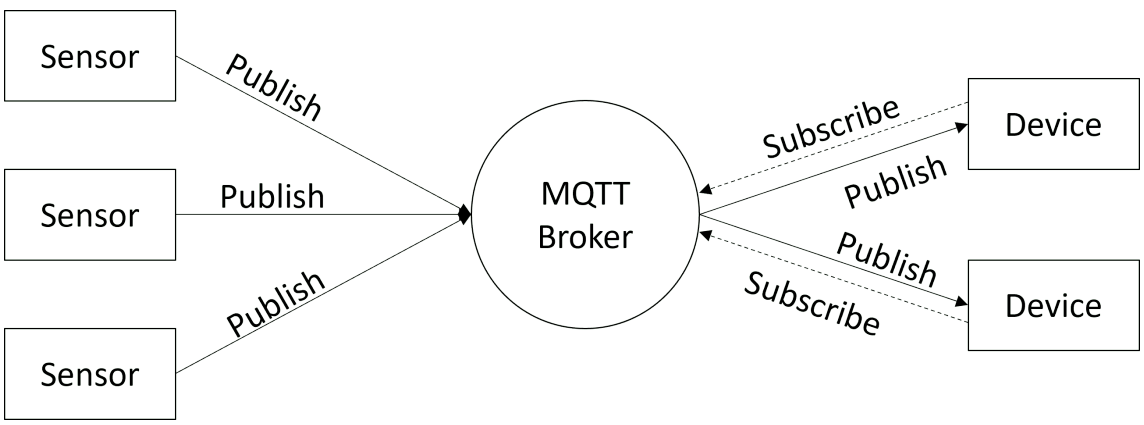


Figure 2: MQTT broker functioning schema.

MQTT Broker provides [68], [73]:

- Message routing: the broker receives messages from clients (publishers) and routes them to the appropriate subscribers based on the topics they are subscribed to.
- Session Management: The broker manages client sessions, including maintaining subscriptions and storing messages for clients that are temporarily offline, depending on the Quality of Service (QoS) level.
- Quality of Service: the broker ensures message delivery according to the specified QoS levels, handling message acknowledgement and retries as needed.
- Security: many brokers offer features like authentication, authorization, and encryption (e.g., using TLS) to secure communication between clients and the broker.
- Scalability: brokers can handle a large number of simultaneous connections, making them suitable for IoT applications where many devices communicate at once.
- Persistence: Brokers can store messages and client sessions, ensuring data is not lost in case of disconnections or server restarts.

An MQTT broker is essential for making easier communication in an MQTT architecture, ensuring reliable and efficient message delivery between clients in various applications, particularly in IoT environments. The most popular MQTT Brokers are:

- Mosquitto: an open-source MQTT broker that is lightweight and easy to install, commonly used for IoT applications.
- EMQX: a highly scalable, open-source broker designed for high availability and performance.
- HiveMQ: a commercial MQTT broker that provides features for enterprise-level applications, including monitoring and clustering.
- RabbitMQ: although primarily a message broker for various protocols, it supports MQTT as a plugin, allowing for integration with existing RabbitMQ setups.

5.1.2 REST API

The RESTful API (REST API) is an application programming interface (API) based on the design principles of the REST (Representational State Transfer) architectural style. The REST is an architectural style for a design that follows a set of rules, such as distributed

hypermedia systems. The Web has been designed following this approach and today Web services that use the REST architectural style are called RESTful Web services. REST APIs are the interfaces of those services. HyperText Transfer Protocol (HTTP) is the requested protocol for REST. The main HTTP methods are four and can be mapped to CRUD (create, read, update and delete) operations, implemented with these methods:

- GET: Retrieves data from the server.
- POST: Submits new data to the server, often resulting in the creation of a new resource.
- PUT: Updates an existing resource with new data.
- DELETE: Removes a specified resource from the server.

The data exchange leverages on top of HTTP internet protocol. One of the fundamental principles of REST is that each client request to the server must contain all the necessary information for the server to fulfil that request. This means that the server does not retain any client context between requests, which enhances scalability and simplifies server design.

In REST architecture, data and functionalities are considered resources, each uniquely identified by a URI (Uniform Resource Identifier). These resources can represent various entities, such as users, products, or transactions, allowing for a clear and organized structure. The REST API schema is reported in Figure 3.

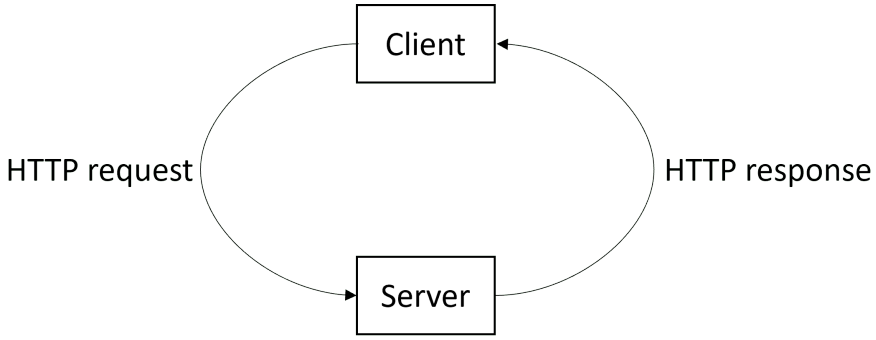


Figure 3: REST API functioning schema.

While REST APIs can support various data formats, JSON (JavaScript Object Notation) and XML (eXtensible Markup Language) are the most commonly used due to their readability and ease of use. Some advantages of REST APIs are:

- Scalability: the stateless nature of REST allows easy scaling, as servers can handle requests independently, making load balancing straightforward.
- Flexibility: since REST APIs can be accessed over standard HTTP, they can be easily

consumed by a wide range of clients, including web applications, mobile apps, and IoT devices.

- Interoperability: REST APIs support multiple data formats and can be integrated with various programming languages and platforms, promoting interoperability across systems.

5.1.3 Python

Python[3] is a high-level, dynamic, and interpreted programming language known for its clear and readable syntax. It was created by Guido van Rossum, and its first version was released in 1991. Since then, Python has become one of the most popular programming languages, due to its simplicity, versatility, and power.

Main Features of Python [74]:

- Simple and Readable Syntax: Python is designed to be easy to read and write. Its reduced and intuitive syntax allows developers to focus on logic rather than complex implementation details. Indentation is mandatory for defining code blocks, which makes the code more readable.
- Interpreted Language: Python is an interpreted language, meaning that code is executed directly by the machine without the need for pre-compilation. This makes debugging and modifying code in real-time easier.
- Dynamic: Python is a dynamic language, meaning that variable types are determined during execution (at runtime), rather than at compile-time.
- Strong and Dynamic Typing: Python is dynamic, it does not allow operations between incompatible types without raising an error. For example, you cannot add a string to a number without performing an explicit type conversion.
- Object-Oriented Programming (OOP) Support: Python fully supports object-oriented programming, allowing developers to create and manipulate objects and classes. However, Python does not force you to follow a specific programming paradigm, enabling you to also use imperative and functional programming styles.
- Extensive Libraries and Frameworks: Python has a vast range of libraries and frame-

[3]https://www.python.org

works for various applications, from data science and artificial intelligence (e.g., NumPy, Pandas, TensorFlow) to web development (e.g., Django, Flask), automation, scripting, and more.

- Active Community: Python has one of the largest and most active developer communities. This has led to a wealth of documentation, tutorials, and online resources that make learning and using Python even more accessible.
- Portability: Python is available on all major platforms (Windows, macOS, Linux, etc.), and Python code can be executed across different platforms with minimal modifications.
- Performance: Although Python is not known for its speed compared to compiled languages like C or C++, it offers good performance for most applications, especially with the use of C-based libraries to optimize computationally intensive tasks.
- Multi-Paradigm: Python supports several programming paradigms, including imperative, object-oriented, and functional programming.

In the literature there are different kinds of applications implemented in Python. It is used in a wide application for web development, thanks to frameworks like Django and Flask, that makes Python widely used for building robust and scalable web applications.

In Data Analysis and Data Science [75], Python is one of the leading tools for data analysis, with libraries like NumPy, Pandas, Matplotlib, SciPy, and Seaborn.

In the Artificial Intelligence and Machine Learning field [76] Python is used because it has libraries such as TensorFlow, Keras, Scikit-learn, and PyTorch.

Python is a powerful, versatile, and easy-to-learn language, suitable for both beginners and experienced professionals. Its large community and numerous libraries make it ideal for a wide range of applications, and a lot of company used it [77]. It is used in this application because it is possible interface it with different technology easily, allowing the development of an entire IoT architecture.

5.1.4 Node.js

Node.js[4] is a popular, open-source, cross-platform runtime environment that allows to run JavaScript code on the server side, outside of a web browser. It is built on Chrome's V8 JavaScript engine, which is known for its high performance in executing JavaScript code.

Traditionally, JavaScript was used for client-side development in web browsers, but Node.js allows developers to use JavaScript for backend/server-side development as well.

Node.js operates on a single-threaded, event-driven model, where non-blocking input/output (I/O) operations (such as file reading or database queries) allow the system to handle many requests simultaneously without waiting for one operation to complete before starting the next. This makes Node.js highly scalable and efficient, especially for applications that require high I/O operations like real-time applications (chat apps, live updates, etc.).

It uses asynchronous programming heavily, which means that instead of waiting for a function to return a result (which would block the next task), it continues with other tasks and processes the result when it's ready. This model is often more efficient in handling multiple requests than synchronous methods, where each request is processed one at a time.

Although Node.js runs on a single thread, its event loop enables it to manage many connections concurrently by delegating I/O operations to the underlying system and handling them asynchronously. This makes it capable of handling thousands of concurrent connections with minimal overhead.

It is cross-platform, meaning that it can run on various operating systems, including Linux, Windows, and macOS, with minimal changes required to the application code.

Due to its non-blocking and event-driven nature, Node.js is well-suited for building real-time applications such as chat applications, gaming servers, and collaborative tools that require fast communication between the server and client.

Node.js is also a good choice for microservice architectures because it is lightweight and enables the creation of small, efficient services that can be deployed independently.

Its use of JavaScript, non-blocking asynchronous operations, and large library ecosystem makes it highly popular among modern developers. Because Node.js is a powerful tool for building scalable, high-performance, and real-time web applications, it was selected to design

[4]https://nodejs.org

the web application.

5.2 IoT node architecture

The IoT node architecture is represented in Figure 4. Both asynchronous communication and synchronous communication are used in this system. The asynchronous communication is implemented with MQTT. The synchronous communication is reached thanks to REST API.

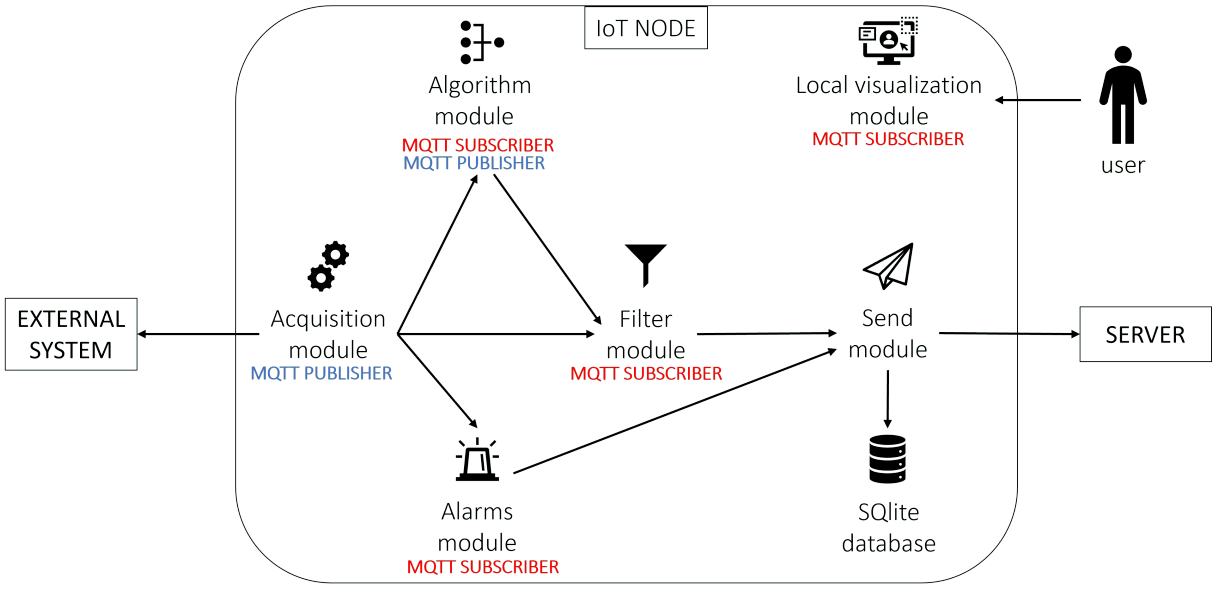


Figure 4: Generic system architecture for the IoT node.

The modules of the IoT architecture are:

- Acquisition module

  The first module necessary for this application is the acquisition module. The data are collected from different sensors and published in MQTT. This module is customized for every different application because it depends on the system. The MQTT broker used in this application is the Eclipse Mosquitto[5] broker, an open-source MQTT broker developed to run on different platforms.

  The data collected from the system can be categorized into two different typologies: functioning data and alarms. The functioning data are data regarding the system, such as temperature, speed measurement, pressure, etc. The alarms directly generated

[5]https://mosquitto.org

by the system are variables the user needs to know as soon as possible. In the next section, the alarms are described concerning the specific use case.

- Alarms module

  This module has the scope to manage the alarms. There are two categories of alarms.

  The first type of alarm is the alarm directly generated from the acquisition system, which in the following are referred to as "system alarms". Those alarms are generated from the system and normally describe a warning in the system's functioning.

  The second category of alarms is computed by the user. Normally the user wants to monitor a variable and wants to know when a defined threshold is reached. This module creates an event when the variable reaches out the defined threshold, and those alarms are called "user alarms".

- Algortihm module

  This module is configured in the server by the user. The scope of this module is to compute new variables interesting for the user, and the acquisition variables are the input of this module. For this reason, there is an MQTT subscriber. Then when the value is computed, it is published via an MQTT publisher. If two different variables are used to compute a new variable, the timestamp of variable acquisition is checked. If they are not the same, the value is not computed. The only exception is the moving average.

  It is not mandatory to install this software module in every IoT device. It is installed only in the application where the user has some interest in computing some variables in real-time.

- Filter module

  The filter module acquires the data from the acquisition module and the data from the algorithm module thanks to an MQTT subscriber. The data are filtered to decrease the dimension of the data to send to the server and later the number of data that are stored in the server database. Filter rules can be defined for all the different variables. The data are prepared and sent to the send module when a set timeout is triggered.

- Send module

  The send module is the only module that communicates with the server via REST

API. The data received from the filter module and the alert module are packed in a known format and sent to the server. For different types of data, there is a different REST API. The alarms are sent to the server as soon as they are received in the send module. On the other hand, the filtered data are sent when a timeout is reached; normally the timeout is set every 30 seconds, to maintain the server data updated.

The IoT node is connected to the internet. It can be connected directly to the Wi-Fi of the industrial environment where it is located, or otherwise, it is possible to add a SIM and communicate using it. The data are sent to the cloud server, but it can happen that there is not an internet connection available due to some failures. To overcome this issue, a database is included in the system. The send module sends the REST API to the server but if the server responds that the data are not well received or if the connection is lost, the data are saved locally. The send module, when the connection is established again, sends the saved data and deletes them from the local database. In this architecture, the selected database is SQLite[6], a C-language library that is developed to implement a small, fast, self-contained, high-reliability, full-featured, SQL database engine.

- Local visualisation module

  The last software module in the IoT node is the local visualisation module. It is a web page that the IoT node exposes and where the data available on MQTT and the active alarms are visualized. It is not possible to reach this page if the user is not close to the specific IoT node and it only shows the information related to the node it is linked. The main scope of this page is to allow the user to make the first configuration of the IoT node, but in some applications, this software module is not even imported.

5.3 IoT Server architecture

The IoT server architecture is represented in Figure 5. It is possible to use a cloud server or not, but the advantage of the cloud server is that it can be accessible everywhere.

The server hosts different processes to meet all the project requirements. The software components are:

[6]https://www.sqlite.org

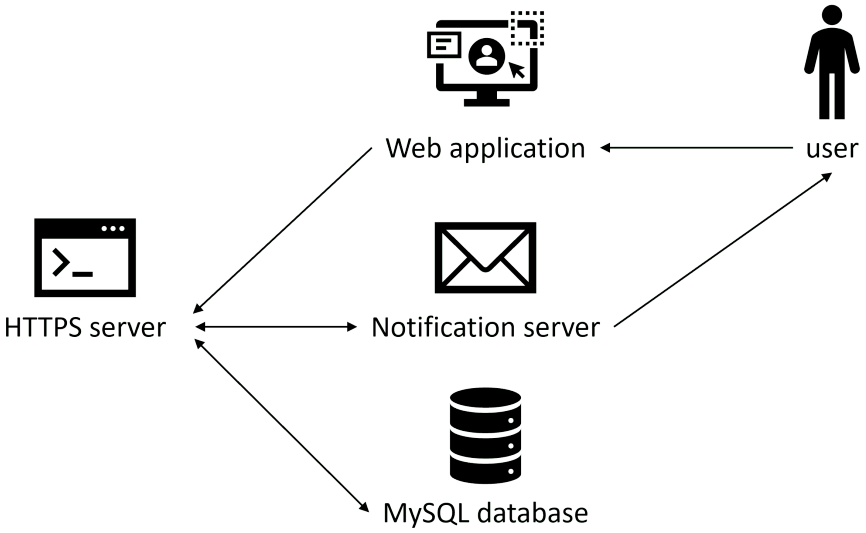


Figure 5: IoT server architecture.

1. HTTPS server;
2. Database;
3. Web application;
4. Notification server;

5.3.1 HTTPS server

The HyperText Transfer Protocol over Secure Socket Layer (HTTPS) is an evolution of the HTTP. The difference is that it implements encryption for secure communication over a computer network. It is used instead of HTTP to reach a major level of security.

The HTTPS server manages the communication with:

- all the IoT nodes connected to the server;
- the web application;
- the database;
- the notification server.

All the IoT nodes presented in the system communicate with the server via REST API. At the same time, the requests from the web application are sent via REST API. Thanks to the header of the API request explored in every request, the HTTPS server can recognise the specific request. When a new batch of data is received from the HTTPS server and the

data timestamp is not older than a specific threshold, the data are published on MQTT. This threshold is 5 minutes in the proposed system.

The batch of data is always saved in the database, and thanks to this architecture, the HTTPS server is the only access point to the database. This facilitates the control of the access to the database.

This module not only communicates with the IoT nodes receiving their data, but it responds to the requests coming from the IoT nodes regarding their configuration, allowing the user to change the IoT node's configuration without being physically connected to the device. To implement that, the IoT node periodically requested its configuration to the server. If there is a change with respect to the previous one, the server sends back the new configuration. Otherwise, the server responds that there is not a new configuration available.

5.3.2 Database

A database is used to store the data in the cloud server. Databases can be divided in two categories: relational database and non-relational databases.

Relational Database Management Systems (RDBMS) are systems that function based on relational theory, which was developed by British computer scientist Edgar F. Codd. This theory asserts that a system should manage data through the relationships among various tables. In the relational model, data is organized into interconnected tables within a database. All the information processed by an RDBMS is stored in these tables, which can be linked through keys. Each table in an RDBMS is made up of columns and rows. Columns represent specific attributes, and the data is structured into records. Each record is usually uniquely identified by a primary key, enabling distinct identification. Structured Query Language (SQL) is a domain-specific language used to manage data, especially in a RDBMS. It is used in handling structured data that are incorporating relations among entities and variables.

Non-relational databases, often referred to as NoSQL databases, are designed to handle large volumes of unstructured or semi-structured data that don't fit neatly into tables. Unlike traditional relational databases, which use a fixed schema and tables, non-relational databases offer greater flexibility in data modelling and storage. The Key Characteristics are:

- Schema Flexibility: NoSQL databases allow for dynamic schemas, meaning you can store data without a predefined structure. This is particularly useful for applications

where data types and structures may evolve over time.

- Data Models: There are several types of non-relational databases, including:
  - Document Stores: store data as documents (e.g., JSON, XML). Each document can have a different structure (e.g., MongoDB).
  - Key-Value Stores: use a simple key-value pair for data storage. This is highly efficient for certain types of queries (e.g., Redis).
  - Column-Family Stores: organize data into columns rather than rows, which can improve performance for analytical queries (e.g., Cassandra).
  - Graph Databases: focus on relationships and connections between data points, making them ideal for social networks or recommendation systems (e.g., Neo4j).
- Scalability: non-relational databases are designed to scale out horizontally, allowing for the addition of more servers to handle increased load, which makes them suitable for large-scale applications.
- Performance: they are optimized for specific data access patterns, often providing faster read and write operations compared to traditional RDBMS, especially when dealing with large datasets.
- Distributed Architecture: many NoSQL databases are inherently distributed, which enhances availability and fault tolerance by replicating data across multiple nodes.

Non-relational databases provide flexibility and scalability for handling diverse and rapidly changing data, making them an essential choice for modern applications.

Due to the requirement for related data, a relational database was chosen. MySQL is provided by Oracle and is widely used worldwide and for its performances [78]; it is an open-source relational database management system (RDBMS) that uses SQL. Every variable received has its own table, automatically created. The data are saved with the timestamp information and the variable value. Then all the information regarding the users, the system, and the IoT node configurations are saved in the database.

Using an SQL database allows to avoid data redundancy, but before starting the data acquisition is important to know the variable structure that needs to be saved in the application, because the SQL database has a fixed schema that must be pre-defined before data insertion. To automatize the process, the server is configured with some typologies of variables, and

when a new variable is received, if it is not present in its table in the database, it checks its typology and then it creates its table with the specific structure for the variable saving process.

5.3.3 Web application

The web application is the link between the user and the system, and it allows the user to manage the entire system from a single access point. It is possible to see the stored data, the stored alarms, the real-time data published from the HTTPS server via MQTT, the IoT node configurations, manage the user properties, add or delete features, etc.

Thanks to the web application, the user can manage the IoT nodes. Different web pages are developed, and the website map of the HMI application is reported in Figure 6. To allow access to this web application only to some users, username and password are required on the login page. If a user that corresponds with the input data is in the database, the system gives a JSON Web Token (JWT) to the connected user, and he can access the server.

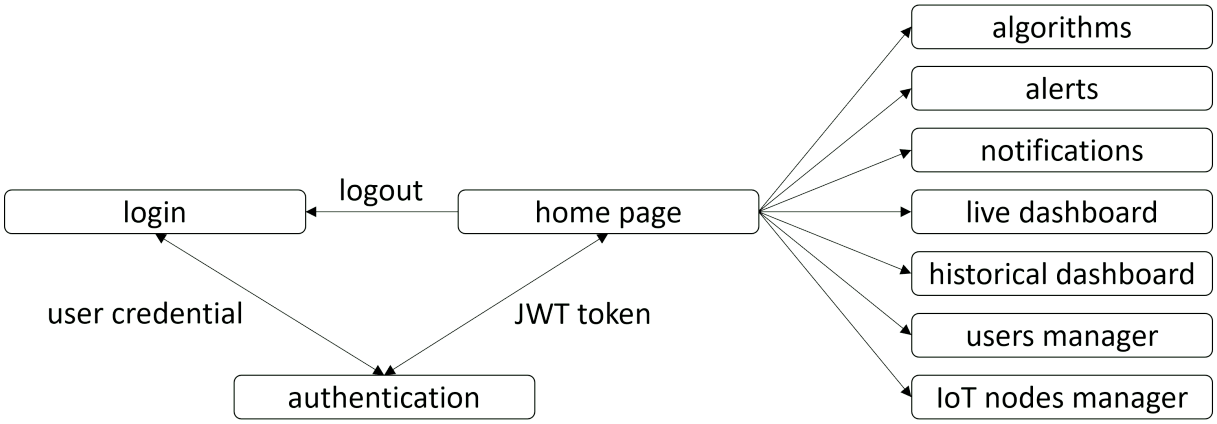


Figure 6: Website map of the HMI application.

The different pages manage the different aspects of the IoT nodes.

- Users manager

  The user manager manages the creation of new users, with name, surname, email, username and password. Email is useful for resetting the password when the user forgets it. A user can change his personal information.

- IoT nodes manager

  In this portion of the web application, the user manages the IoT nodes. To link the

physical IoT node with the server, the user creates a virtual object on this page with a unique ID and a random password; these are encrypted and exported to the IoT node. The IoT node knows the key to decrypt the encrypted ID and the encrypted password. The IoT node can communicate with the server via REST API and be recognized by the authentication module thanks to the ID and password.

- Algorithms

  On this page, the user can implement specific algorithms for the IoT nodes. The system knows the IoT node configuration and the available options, and some operations are available to be computed. Those operations are:

  - Sum;
  - Multiply;
  - Square root;
  - Moving average.

  The user selects the operation and the input variables for the operation. These are saved in the database and computed directly on the IoT node whenever the data are available.

- Alerts

  This module allows the creation and management of user alarms, as well as the visualization of both user alarms and system alarms received from the IoT nodes saved in the database. The user can set a threshold that a specific variable should not exceed. If the variable exceeds this value, an alarm is triggered. The possible threshold settable in the system are:

  - max: the value exceeds a set threshold;
  - min: the value is lower than a set threshold;
  - equal to: the value is equal to a set threshold;
  - different to: the value is different than a set threshold;
  - in the range: the value is inside a settled range of threshold;
  - outside the range: the value is outside a settled range of threshold;

  The received alarms are shown in a table with the information of the specific alarm.

- Notifications

  The user can select an alert and decide to link it with a notification. When the selected alarm is received from an IoT node, the notification is sent to the interested user. The notification can be an email or a telegram bot.

- Live dashboard

  The live dashboard shows the data received in the last 5 minutes from a selected IoT node. The user can view the most recent data from the selected node and change the variable displayed on the graph. This page helps the operator understand the data trends. The data are automatically updated when a new batch of data is received. If the IoT node is not connected and there is no data from the last 5 minutes, no data are shown.

- Historical dashboard

  The user can select a specific IoT node, a range of time and a variable and this page shows the data collected in a graph. On the x-axis there is the time, on the y-axis there is the variable. The user can see the data trends, and it is possible to download those data in a CSV file. An example is reported in Figure 11 and explained better in the next section.

5.3.4 Notification server

If an alarm is received from an IoT node and it is linked to a request of notification, the HTTPS server publishes this information via MQTT and it is received by the notification server that has an MQTT subscriber. Then the user receives an email or a telegram message, depending on how the system was previously configured, with the alarm information. To send a telegram message, an automatic Telegram chatbot is used, that sents a message containing the alarm and the specific IoT node that triggers the alarm. A Telegram chatbot is a software application and it is designed to performe an automated task, such as answering questions or providing information using the Telegram app. In this application it is designed to send a message with the alarm information. Using the website, the user can configure new alerts and create notifications linked to a specific alert.

5.4 Conclusion

In this section, the proposed IoT architecture is presented. First, the basic concepts regarding the software and the communication protocol adopted are explained. Then, both the IoT node and the IoT server architecture are described. In the following sections, the IoT architecture is applied to two different use cases in order to understand the system's functionality.

6. First use case: AGV application

This section applies the proposed architecture in a real industrial use case. In particular, the IoT architecture is applied to monitor an Automatic Guided Vehicles (AGVs) fleet and to support the operator in undesrtanding the system and the faults occurred. Most aspects described in this section were previously published in [79].

6.1 System background

An AGV is a mobile robot used in the industrial environment to transport materials and products. It is programmed to travel on predefined routes. AGVs operate using predefined paths and rely on various navigation technologies, such as magnetic strips, lasers, or computer vision, to move autonomously within a facility. AGVs can be equipped with multiple load-handling mechanisms, such as forks, conveyors, or hooks, to transport different objects. They can be integrated with warehouse management systems (WMS) and other automated systems to streamline operations and improve efficiency.

AGVs have sensors to detect obstacles and prevent collisions, ensuring safe operation around people and other machines. AGVs can be programmed for various tasks, making them adaptable to changing workflows and production demands. Compared to a forklift, an AGV achieves more efficiency and productivity and does not require an operator.

AGVs are widely used in sectors such as:

- Manufacturing: to move both raw materials and finished products.
- Warehousing: to transport goods within distribution centers.
- Healthcare: to deliver supplies and medications within hospitals.
- Food and Beverage: to assist in production lines and distribution.

Overall, AGVs enhance efficiency, reduce labour costs, and improve safety in material handling processes.

AGVs move thanks to smart guidance and control tools; different technologies are available

to implement navigation systems, but they can be categorized into Trajectory techniques and Autonomous navigation [80]; moreover, those techniques can be merged and a hybrid technique is obtained. In the trajectory-based method, AGV paths are pre-defined according to the logistical flows. These trajectories are established during the system layout design and can only be altered through a complete redesign of the logistics flows; this means stopping the industrial plant for the time requested to redesign the system.

On the other hand, in the autonomous navigation approach, individual AGVs autonomously plan their routes based on the localization methods available in the facility and independently determine their missions. The AGV knows the plant layout, but the routes are not predefined. This approach often uses Simultaneous Localization and Mapping algorithms, enabling the AGVs to incrementally construct a path in the facility to perform the desired mission.

In the scenario used to validate the system, the AGVs use laser technology to understand its collocation inside the environment, based on the trajectory-based method. Reflectors are available and known by the AGVs, and the number of reflectors visible by the AGV at a specific point defines the position inside the plant.

While this system supports the operator's work, it becomes difficult for the operator to understand some aspects of the system's functioning. The AGV data are normally not collected, and those data are not easily accessible from non-expert users.

Fault diagnosis involves monitoring and identifying faults as they arise. With the advent of Industry 4.0 and advancements in IoT technologies, a variety of solutions have emerged to facilitate remote fault diagnosis within AGV systems [81]. For instance, Chen [82] proposed a system architecture for wireless data acquisition in AGV systems, recognizing its potential for implementing fault diagnosis procedures. The maintenance strategy for AGVs considers various factors, including system layout, maintenance site location [83], [84], and maintenance scheduling [85]. While environmental factors are considered in maintenance strategies, fault diagnosis applications often focus specifically on faults related to the AGV itself.

Wang [86] utilized a Convolutional Neural Network to diagnose faults by analyzing a single vibration signal from an AGV. Stetter [87] employed virtual sensors to estimate the forces and torques acting on an AGV. Mrugalska [88] assessed the Remaining Useful Life of AGV batteries to maintain a robust power system monitoring approach. Dares [89] developed an

AGV to simulate fault conditions and implement a fault detection strategy, emphasizing the importance of accounting for both internal and external factors.

Additionally, some studies examine AGV fault diagnosis from a layout perspective. Witczak in [90] introduced a mathematical model to optimize transportation tasks within AGV systems, employing a model predictive control algorithm and fault-tolerant strategies. While this model enhances transportation efficiency, it does not address potential faults arising from AGV interactions with the environment or internal malfunctions. In [91], a mathematical model is proposed to tolerate faults, focusing on scheduling constraint violations as the primary fault effect. Although past research provides valuable insights, the interaction between AGVs and their industrial environments remains underexplored.

The previous research does not consider the interaction with the environment. The system can easily know AGV's expected mission time in advance, but at the end of the mission, it is not clear why the real mission time and the expected mission time are different. This work presents a methodology to understand system faults resulting from AGV interactions with their surroundings. The proposed systems are designed to be user-friendly for non-specialist operators, enabling them to understand system faults better and actively engage in troubleshooting, thereby serving as valuable tools for operational support.

6.2 System setup

The AGV communicates with a central server. This server is the key access point between the users and the AGVs, and in particular, it manages three aspects:

- it sets the user's commands;
- it monitors the AGV's traffic;
- it plans the AGV's mission.

The same server manages all the AGVs inside the same industrial plant. A Raspberry Pi, a single-board computer, is added to the system to implement an IoT infrastructure following the architecture previously described. Other devices can be used instead of a Raspberry Pi, such as a PC, a mini PC, a Banana Pi, etc. The choice to use a Raspberry Pi is because this is a small solution, and it does not interfere with the environment. The system IoT schema is represented in Figure 7. This device reads the data directly from the central server via Modbus protocol, a standard industrial communication protocol.

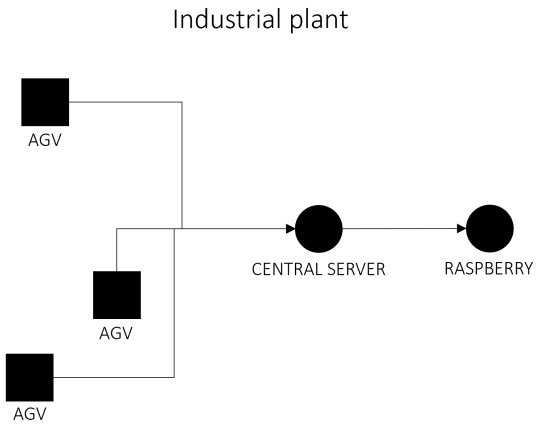


Figure 7: System schema for the IoT node.

6.3 IoT node application

The software architecture of the IoT node is represented in Figure 8. Considering the architecture shown in Figure 4, the local visualization module is removed, because it is not a user requirement in this project.

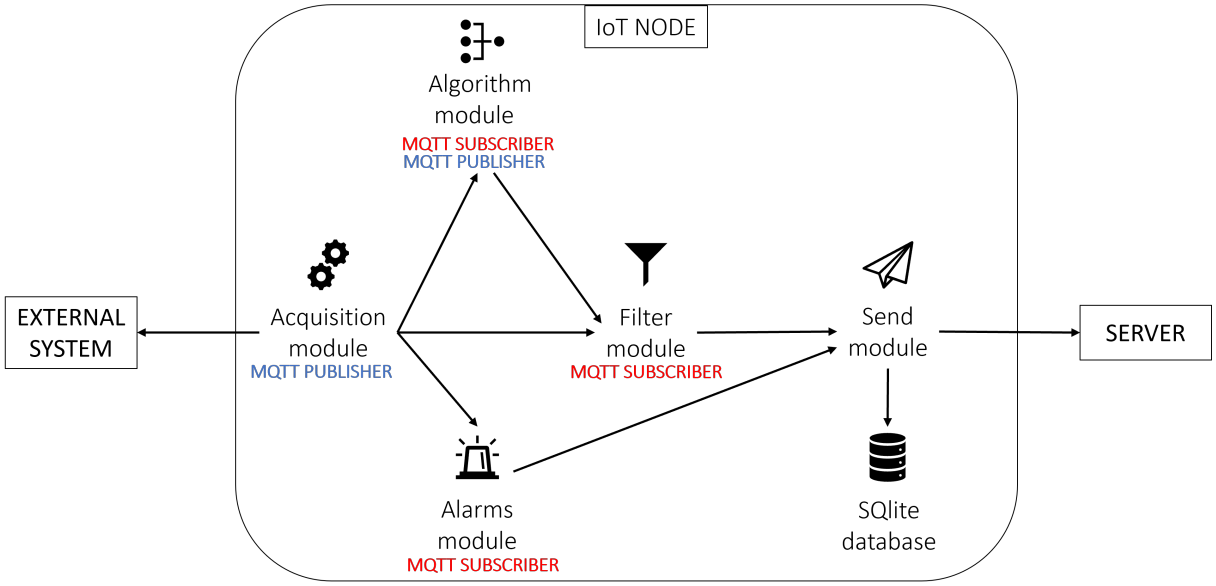


Figure 8: IoT node system architecture for the AGVs use case.

The data collected are reported in Table 1. All the data are collected at the same moment, once every second. The variable of the position is an array containing three float values: the x position of the AGV, the y position of the AGV and an angle theta, corresponding to the AGV orientation in the plant.

VARIABLE	TYPE
Effective AGV speed	float
Expected AGV speed	float
Instant current	float
Instant electrical tension	float
Position	array
Number of reflectors	integer
Inverter temperature	float
Motor temperature	float
Active emergency	boolean
Warning state active	boolean

Table 1: List of variables acquired.

Those data are collected via Modbus protocol from the central server by the Rasberry Pi and then published on MQTT. The algorithm module can compute more information interesting from the user's point of view.

An important variable of interest that is not directly available in the system is the electric power of the AGV. Since both electric current and voltage are part of the system's available variables, it became possible, on the cloud server side, to create an algorithm that calculates the power as the product of these two variables. This algorithm uses the current and voltage readings to compute the electrical power.

On the IoT node side, the algorithm is received from the server as one of the variables to be calculated. Once received, the algorithm module subscribes to the voltage and current variables, so it can access the real-time data as they are published. The system ensures that each variable, including voltage and current, is published with its respective acquisition timestamp, which is essential for synchronizing the calculation and maintaining data accuracy.

When the IoT node receives the voltage and current values from the broker, it performs a comparison of their respective timestamps to ensure the data's temporal coherence. Specifically, the algorithm calculates the power product only if the timestamps of the two variables differ by a value that is below a predefined threshold. This control mechanism is crucial to ensure that the calculation is based on synchronized data, avoiding the possibility of generating a result that would be meaningless due to a mismatch in timing. If the timestamps

are sufficiently close, the algorithm proceeds to compute the power as the product of the voltage and current values.

The resulting power value, along with its corresponding timestamp, is then published back to the system via MQTT. The timestamp of the power value is calculated as the arithmetic mean of the two input timestamps, ensuring that it accurately represents the time at which the power calculation was performed. This method of calculating and publishing the power value is essential to maintain data consistency and to allow accurate monitoring and analysis of the electrical consumption of the AGV.

6.4 Server application

The data are sent via REST API in a cloud server where they are stored. A web application to access the data is developed. The user can access the web application through a user-friendly graphical user interface (GUI). Using a cloud server allows the user to access those data everywhere, without being physically connected to the AGV central server in the plant. The data are stored in the database with the timestamp information, allowing the correlation between multiple pieces of information. The most important pages are now described.

6.4.1 Login page and Registry page

The login page is the first visible page and manages access to the web app. The operator can log in using username and password, as visible in Figure 9.

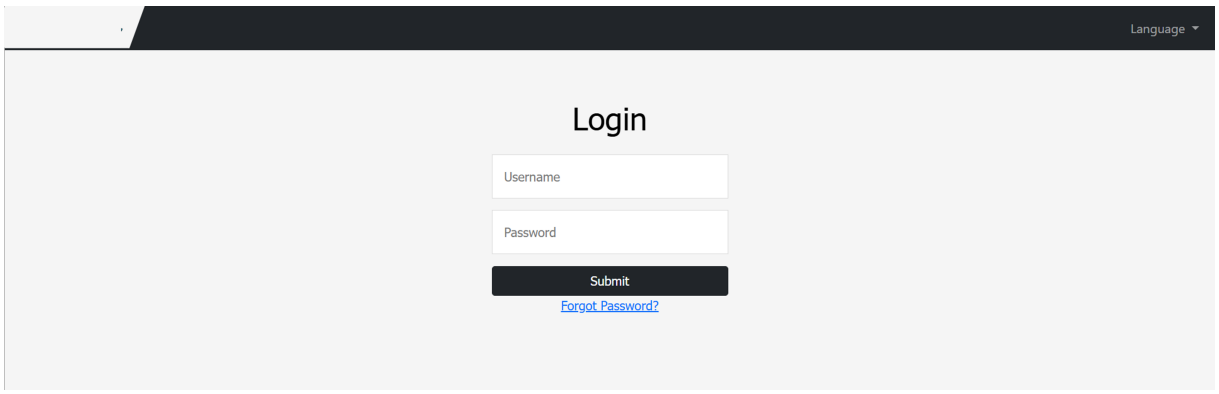


Figure 9: Login Page.

The registry page manages the creation of new users that can access the web application and the management of new AGVs. Here the operator creates a virtual AGV selecting all

the components that are embedded in the real one, and then the new AGV data when it is received from the IoT node is linked with the virtual one thanks to an ID that connects the virtual one with the real one.

6.4.2 Data page

There are two kind of data page, that are the live dashboard page and the historical dashboard page. On the historical dashboard page, it is possible to view the collected data for each AGV. The user can access the page and click a button that opens the following form, shown in Figure 10.

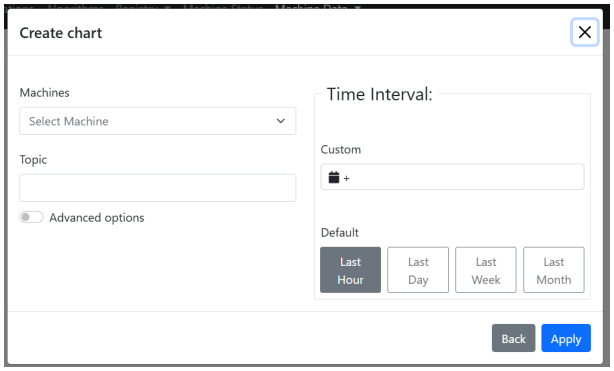


Figure 10: Add Graph Modal Dialogue.

By completing the modal dialogue, the operator obtains the requested graph from the server database. The operator can select the desired time interval, as well as choose both the data and algorithms to be visualized. An example is shown in Figure 11.

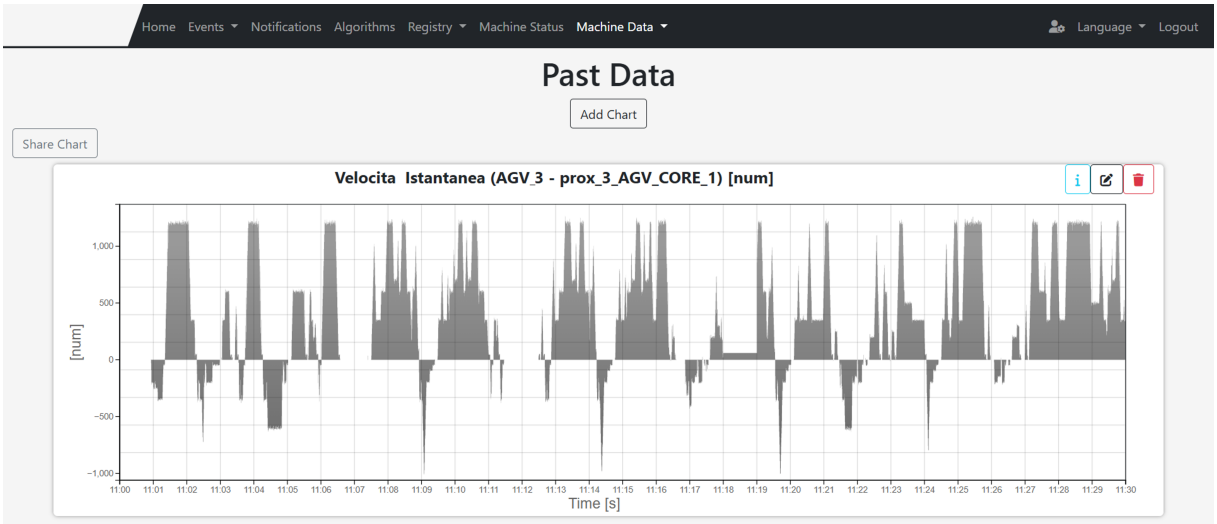


Figure 11: Historical dashboard Page.

The live dashboard page is similar to this page. The user can select the specific AGV and the specific variable, and the live data received are shown.

6.4.3 Algorithm page

On the page, there are the algorithms created visible as cards, as shown in Figure 12.

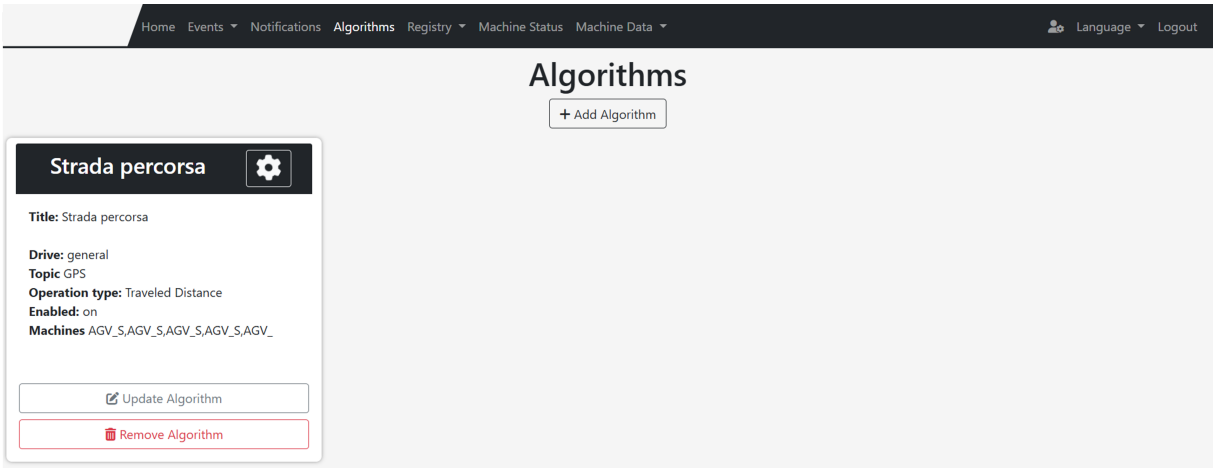


Figure 12: Algorithm Page.

To add a new algorithm, the operator can click on the "Add Algorithm" button and then complete the data requested in the form as shown in Figure 13.

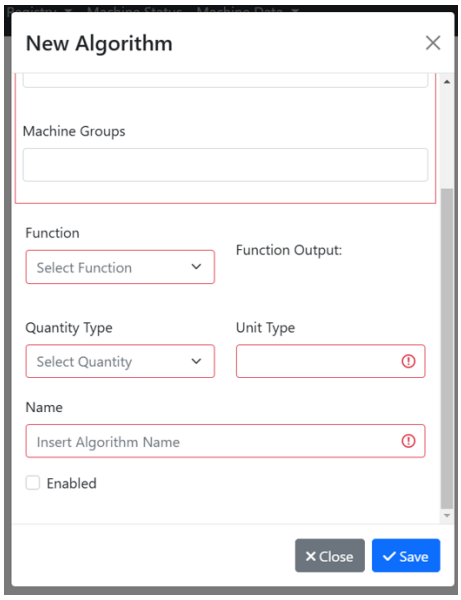


Figure 13: New Algorithm Modal Dialogue.

As previously described, in the Algorithm page the user can create an algorithm and then

the IoT node every 5 minutes requests the server the new algorithms created linked with it. To create the power consumption algorithm, the user needs to:

- Select AGV: the user selects all the AGVs for which they want the algorithm they are creating to be applied.
- Select the Function: the function to compute the electrical power is the multiply. The available functions can be selected from a drop-down menu.
- Select the Unit Type: the user can decided in wich unit of measurement the obtained data is shown in the live dashoard and historical dashboard.
- Give a name to the algorithm: the user gives a name to the algorithm to clearly identify it in the page.

Those information are saved in the database, and the IoT nodes received all those information via RestAPI. The algorithm subribers subscribes to the input variables, and publishes the result as a new topic on MQTT.

6.4.4 Alert page

The alert page, or Event page, has two different sections. The first section is a page where the user can create new alerts (or events) and as in the algorithm page, they are visible as cards. To create a new alert, the user clicks on the "Add Event" button and then a form appears. The data requested are reported in Figure 14. The operator can select the AGV or more AGVs to link with the alert, the threshold type, and the variable and give a name to the alert to recognise it. The second section contains a table where all the alerts collected from the IoT node and saved in the database are shown with the information regarding:

- the time when the alert was triggered.
- the time when the alert was triggered off.
- the alert triggered.
- the AGV that is considered.
- the value that triggers the alert.

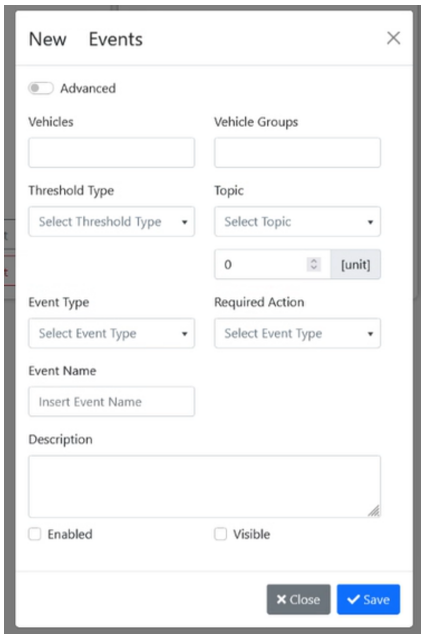


Figure 14: New Alert Modal Dialogue.

6.4.5 Notification page

The notification page is similar to the alert page and the algorithm page. The notifications created are reported as cards. To add a new notification the operator can click on the "Add Notification" button and complete the modal dialogue in in Figure 15. As visible the user needs to select an event to be linked with the notification.

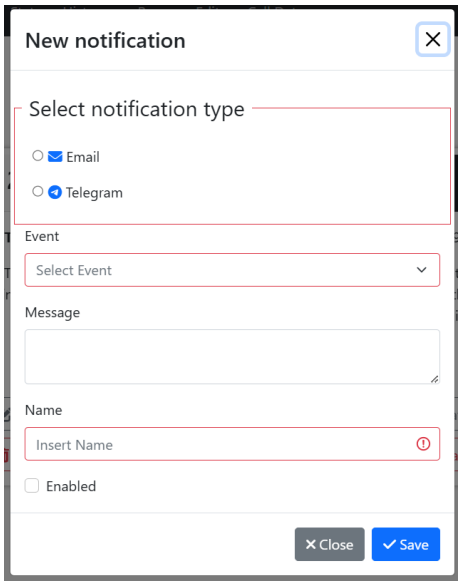


Figure 15: New Notification Modal Dialogue.

6.4.6 Status page

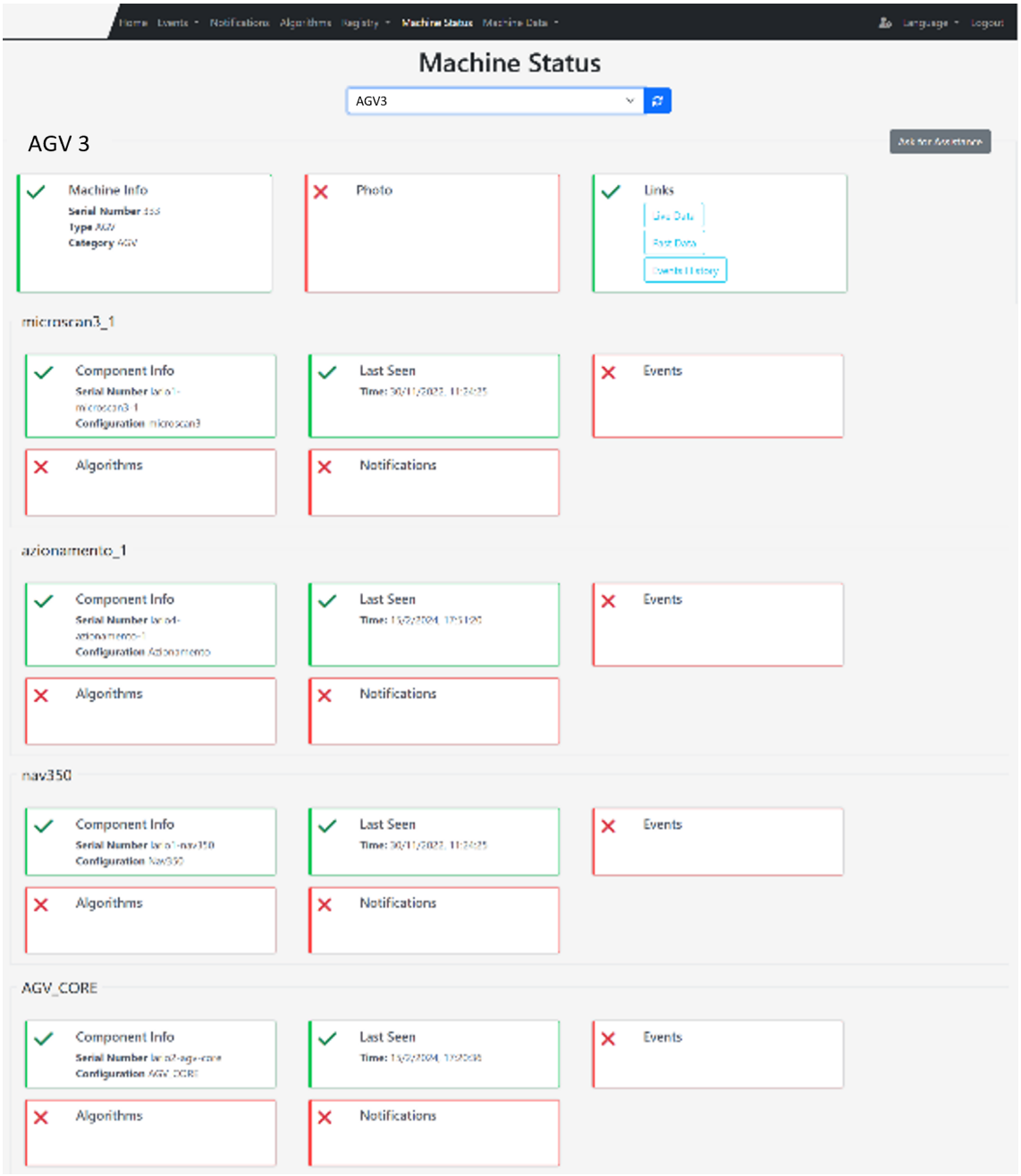


Figure 16: Status Page.

On this page, the user can select a specific AGV, and then all its components with their specific status will be displayed. This helps the user better understand what has been implemented for each component. For all the components, the "Last Seen" information is provided, which indicates the last time a specific component communicated with the server. This allows the operator to see if a component has not communicated for a long time and investigate the

issue. For each component, the algorithm, alert, and notification implemented are shown, along with information such as the serial number and some embedded configuration details. The status of the AGV3 is shown in Figure 16.

6.5 Fault analysis

In this section, the fault analysis for this specific use case is reported. The scope of this system is to support the operator in understanding the system and its relationship with the environment. In particular, the interaction between the AGV and the environmental elements can cause the AGVs to lose efficiency because unnecessary stops or unexpected decreases in speed cause an increase in average mission time. For this reason, the losing efficiency is considered a fault in this research. The main problem is that losing efficiency does not occur for just one cause, but thanks to the expert knowledge three different events that cause loss of efficiency are identified and evaluated. Those are:

- A low number of reflectors available. Reflectors are collocated inside the industrial plant to allow the AGVs to locate themselves using laser technology. When there are not enough reflectors the AGV decrease its speed and in the worst scenario stops its mission.
- Active "warning state". The warning state is a critical state for the AGV, which as a precaution decreases its speed.
- The real speed is lower than the expected one. During the plant setup, the speed to reach every plant section is set. When the AGV moves the real speed should be similar to the expected one.

The next step consists of understanding the fault's impact on the system and its consequences. For the proposed events, they are:

- The number of the reflectors. If the reflectors are insufficient, the AGV will either halt or reduce its speed. The AGV will come to a stop if it cannot detect three reflectors, as it cannot accurately determine its position within the facility. If the AGV detects three reflectors but still has uncertainty about its exact location, it will slow down. This uncertainty could be due to interactions with other objects or operators, which might accidentally obstruct the reflectors while the AGV is in motion.
- Number of times the AGV is in “warning”. When the AGV enters a “warning” state,

its speed must be reduced since “warning” is classified as a critical condition for the AGV’s microscan system. Similar to the previous issue, this warning often results from interactions with objects in the AGV’s path.

- Difference between the set speed and the real speed. When there is a significant discrepancy, it indicates a problem with the system configuration. The system specifies the speed for each point, so if the actual speed is lower than the set speed, something is impacting the AGV’s performance (for example, a person or an object like a box near the AGV’s path). A large difference between the set and actual speed leads to a considerable reduction in efficiency.

The third step is to analyze the system and to identify the variable and the condition to detect the system fault. For the proposed events, they are:

- The number of the reflectors: The key variable to track is the count of visible reflectors for the AGV. You should focus on the condition when this count falls below three.
- Number of times the AGV is in “warning”: The variable to monitor is the AGV’s "warning" state. You should pay attention to the condition when this "warning" state is active.
- Difference between the set speed and the real speed: The variables to monitor are the actual speed and the set speed of the AGV at a given point. The condition to observe is when these two values do not match.

For every fault, the specific analysis made is reported.

6.5.1 Number of the reflectors

From the database, all records where the number of reflectors is less than or equal to three are extracted. This information is critical, as it not only captures the instances of low reflector counts but also includes the specific position of the Automated Guided Vehicle (AGV) within the plant at the time each event occurred. By associating the fault data with the AGV’s location, we can better understand the environmental contexts in which these faults transpired.

To effectively visualize this data, a heatmap is used. A heatmap serves as a graphical representation that allows for the easy identification of patterns and anomalies within the dataset. Typically, heatmaps display data values through a colour gradient, where different

colours represent varying magnitudes of data. This method facilitates quick recognition of areas of concern. In this analysis, the plant is virtually segmented into 400 distinct zones, as visible in Figure 17.

This division is essential for pinpointing the specific areas where the highest frequency of errors has been observed. For each zone, we tally the instances where the number of reflectors fell below or equalled three during the selected observation period. The results of this counting process are then converted into percentage values.

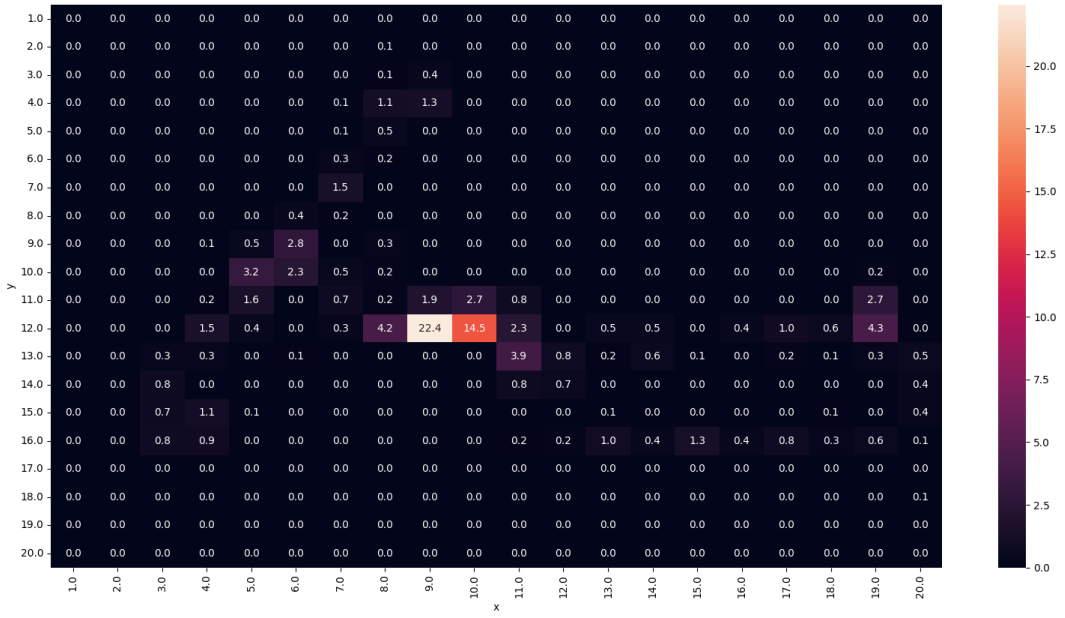


Figure 17: Reflector analysis results.

This visual representation not only highlights the zones with the greatest number of faults but also provides a clear overview of where improvements may be necessary to enhance the overall performance and reliability of the AGV system.

To better understand the result, it is possible to add the map layout in the heatmap visualization, considering the AGV position at each moment. The result is shown in Figure 18.

In the heatmap, lighter colours indicate a higher percentage of instances where the number of reflectors was less than or equal to three. This visual cue effectively communicates areas of concern within the plant.

The AGVs navigate through the facility by following predefined arcs, a method that allows

the system to maintain a comprehensive understanding of all available paths. As the AGVs carry out their tasks, the system continuously communicates the designated route for each operation.

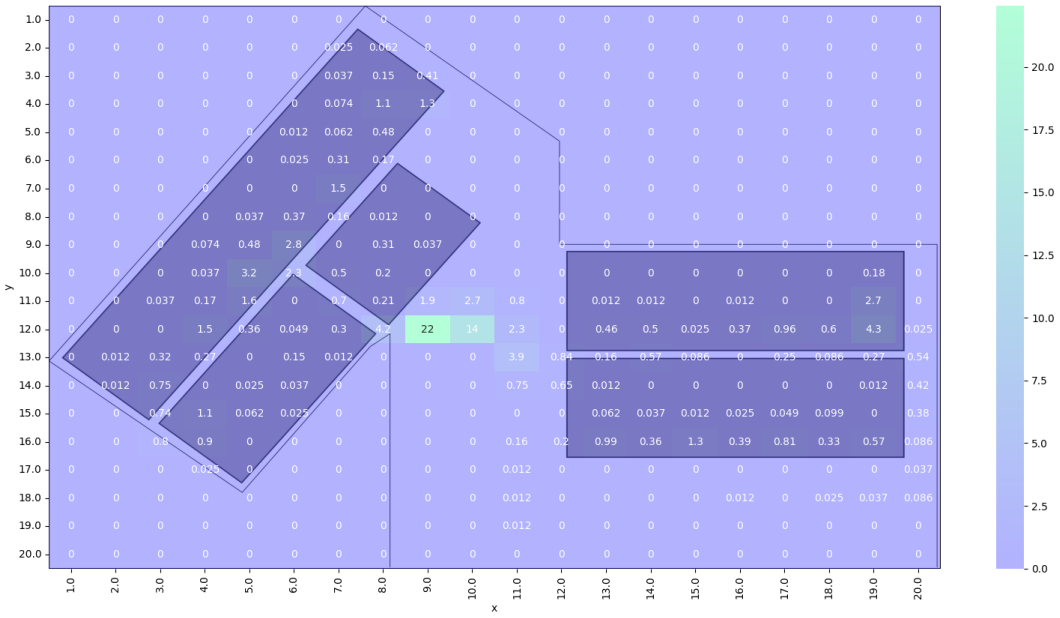


Figure 18: Reflector analysis results with industrial plant layout.

Each route consists of a series of arcs, and at any given moment, an AGV is aware of its starting node and its destination node along the current arc. The starting node is defined as the point where the AGV initiates its movement, while the destination node is the point the AGV is travelling toward. For this reason, the starting node of a new arc is the same as the destination node of the previous arc, creating a seamless flow of movement throughout the plant. An interesting aspect of this analysis is the ability to correlate the previous data to identify which destination nodes the AGVs were approaching during critical situations, such as when the number of reflectors dropped to three or fewer.

Specifically, by examining the data presented in Figure 19, we can know wich node was the destination nodes that were being targeted by the AGVs at those critical moments. Collecting data from the database, it is possible to extract all destination nodes corresponding to instances where the AGV's reflector count reached a critical level. The results depicted in Figure 19 highlight that two particular destination nodes number 5 and number 6 stand out as critical.

A closer examination of the AGV's layout reveals that these two nodes are near one another, suggesting a potential area of concern regarding reflector visibility. This clustering of critical nodes indicates a specific zone within the plant that may warrant further investigation and targeted interventions to enhance the reliability and performance of the AGV system, particularly in terms of maintaining adequate reflector visibility during operations.

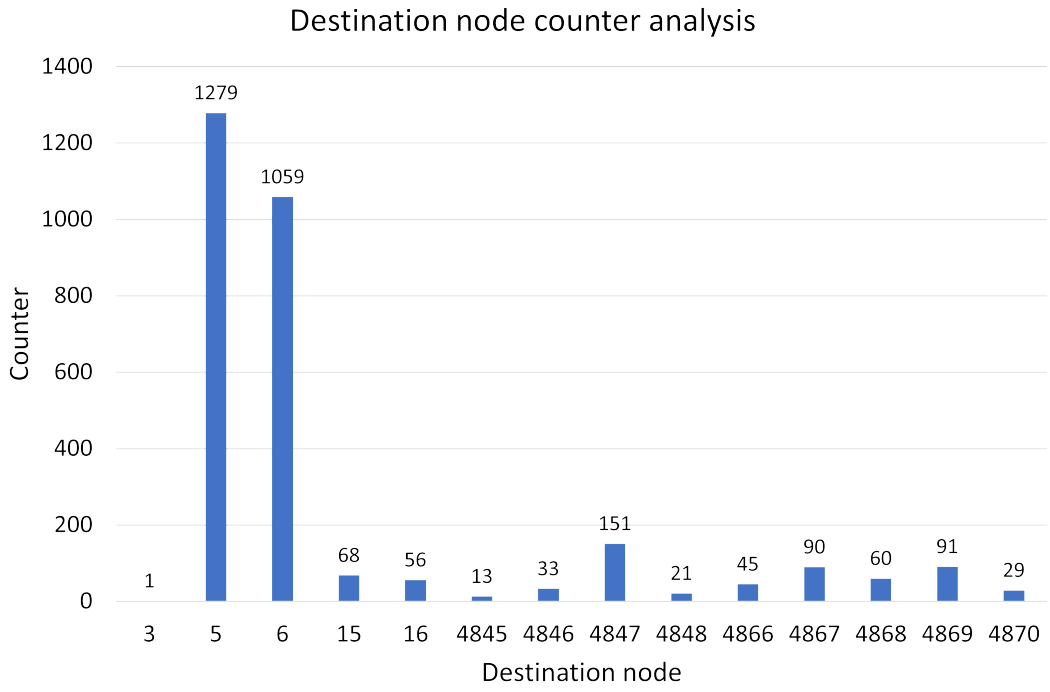


Figure 19: Reflector analysis with the destination nodes.

6.5.2 Number of times the AGV is in "warning"

In the context of the "warning" state for the AGV system, a novel approach has been implemented to enhance operational safety and monitoring. Thanks to the alarms module implemented both in the cloud server and the IoT node the warning error is detected. The interest in the "warning" state is because when the microscan embedded within the AGV detects a condition that requires a 'warning' state, the vehicle automatically reduces its speed. This automatic response is crucial for preventing potential accidents and ensuring the safety of both the AGV and its surroundings.

To facilitate this process, a trigger event has been established in the web application on the alerts page. In the cloud server, it is possible to manage the activation of the notification linked with the "warning" state alarms. This alarm is automatically collected from the

AGV. The IoT node downloads every 5 minutes its configuration from the cloud server, and when this alarm is collected the IoT Alarms module is designed to recognize when the AGV transitions into the “warning” state. Upon detection of this condition, the IoT node software promptly sends a REST API with the alarm information, along with the specific value that activated the threshold, to the cloud server. This seamless communication between the AGV and the cloud infrastructure is vital for maintaining an effective monitoring system. Once the cloud server receives the trigger, if the notification is linked with the "warning" state alarm is active it initiates a notification protocol that alerts users via email about the warning state. This immediate notification allows operators to stay informed of any issues as they arise, enhancing their ability to respond quickly. The email screenshot is visible in Figure 20.

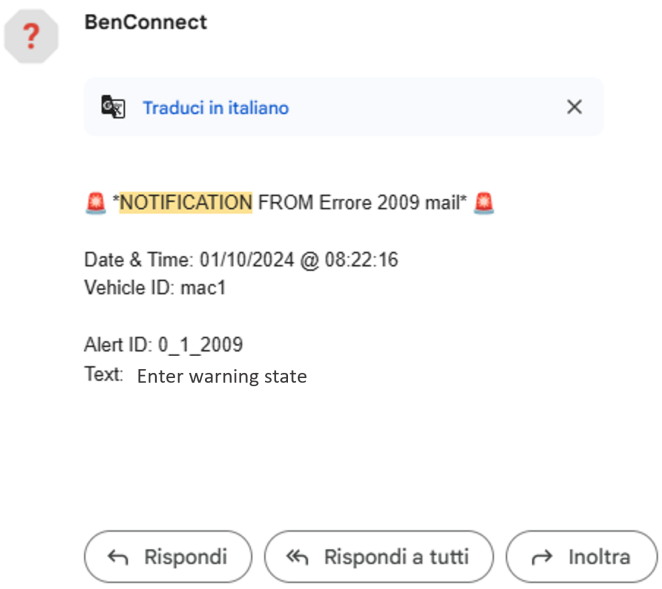


Figure 20: Mail alarm screenshot.

Furthermore, the cloud server features a web application that maps the location of the warning event. As illustrated in Figure 21, this map provides a visual representation of the plant layout, with a designated ping indicating where the warning state was triggered.

Users can access this map remotely, which offers significant flexibility for monitoring operations from various locations.

The data collected regarding the warning state, along with the associated location information, are readily available through the user interface of the cloud server. This accessibility empowers operators to quickly understand the context of the warning and to take appropri-

ate action. The advantage of this comprehensive system lies in its ability to provide real-time information regarding malfunctions. This proactive approach not only enhances operational safety but also supports the operator in pinpointing the precise location of the warning state, enabling timely intervention and reducing potential downtime in the system. Overall, this innovative mechanism significantly contributes to the effective management and oversight of AGV operations within an industrial environment.

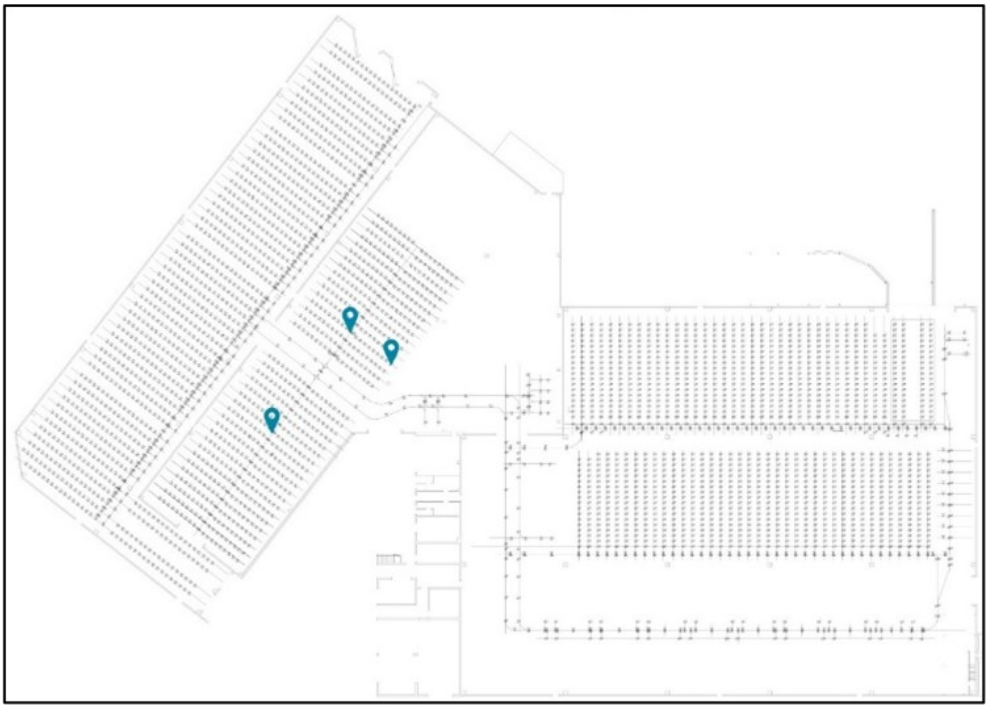


Figure 21: Warning state analysis with the map layout.

6.5.3 Difference between the set speed and the real speed

In analyzing the speed differences of the AGV system, a methodology analogous to that employed for tracking the number of reflectors is applied. The developed system systematically records critical parameters, including the actual speed of the AGV, the expected speed, and the AGV's positional data within the operational environment. This comprehensive data collection aims to identify instances where the AGV's speed falls below the anticipated threshold, particularly when it is significantly lower than a pre-established minimum value.

This information is invaluable as it aids in optimizing the layout during the speed-setting process. By understanding the conditions under which the AGV operates below its expected speed, operators can make informed decisions about potential adjustments to the operational environment, thereby enhancing overall efficiency and safety.

The results of this analysis are illustrated in Figure 22, which presents a detailed breakdown of the plant layout. The layout has been segmented into 400 distinct areas, and for each section, the system counts the number of occurrences where speed discrepancies arise. Similar to the previous tests conducted on reflector numbers, a heatmap is generated directly over the plant map to visually represent the data. The values depicted in this heatmap are expressed as percentages, providing a clear overview of where speed-related issues are most prevalent.

Furthermore, as with the prior analysis, this method allows for the identification of specific sections within the layout where speed discrepancies are more frequent. By pinpointing these areas, operators can focus their attention on optimizing those regions, thus ensuring that the AGV operates within its desired speed parameters more consistently. This targeted approach not only enhances the overall efficiency of the AGV system but also contributes to the development of best practices for managing and configuring the operational layout to facilitate smoother vehicle performance. Overall, the integration of this data-driven analysis represents a significant advancement in the operational management of AGVs within industrial settings.

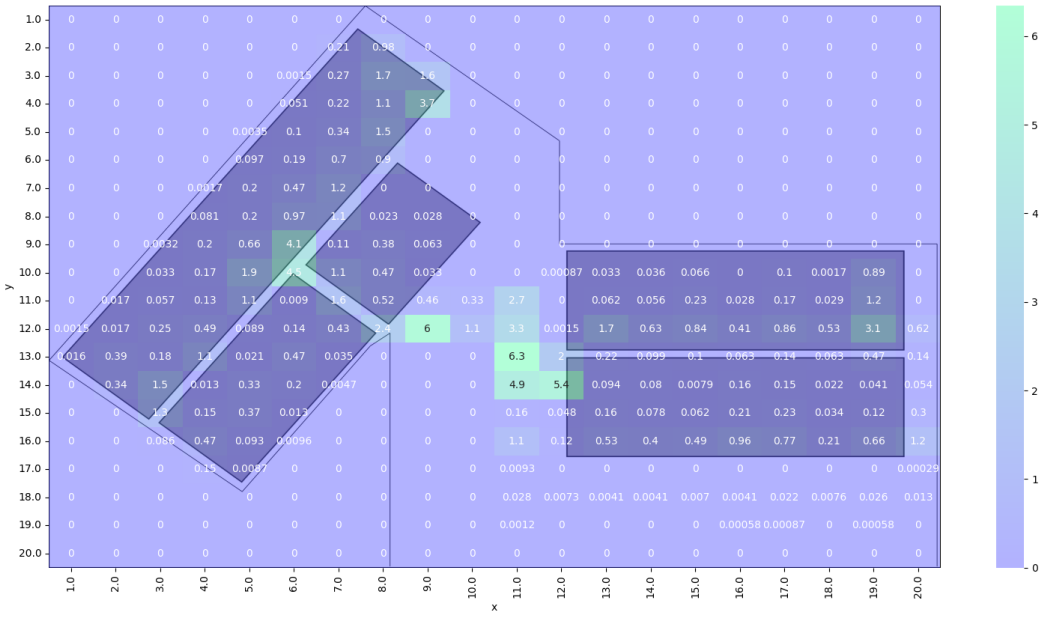


Figure 22: Speed analysis with the map layout.

Additional insights can be gained through the analysis of the gathered data. For instance,

it is possible to determine how many times the number of reflectors is less than or equal to 3 while the AGV is en route to destination node 5. This node has been identified as the most critical, as illustrated in Figure 19. The analysis reveals that this situation occurs 2226 times with the same dataset. To enhance the information available to the user, it is also relevant to assess how many times the number of reflectors is less than or equal to 3 while the AGV is heading to destination node 5, the "warning" state is active, and the AGV is located within the green area shown in Fig. 18. This scenario occurs 711 times.

6.6 Conclusion

AGVs are widely used in industrial environments to move materials and products in different positions. In the literature, many studies focus on monitoring the health status of the system components to predict the failure of the AGV, but there is no focus on Understanding of the interaction of the AGV with the environment. In this section, the proposed IoT architecture with the new diagnosis methodology is applied in order to understand this relationship. The section describes the technical implementation of the system and how the operator can easily access the AGV data from a GUI that can be accessed everywhere. The fault analysis section describes how the system can be used by the operator and how the operator's empowerment is reached thanks to this application.

7. Second use case: MyWelder

In this section, the proposed architecture to collect, store, visualise and analyze data is applied in an industrial use case involving a robot-assisted welding solution for automated MIG/MAG welding. This product is produced by "Industria Tecnologica Italiana", an Italian SME company, with the brand name "MyWelder"[1] and some more information about this product is available online. The welding process is described in [92]. Most aspects described in this section were previously published in [93].

7.1 System background

In the context of Industry 4.0, the interaction between humans and robots is becoming progressively more frequent. Several factors contribute to the growing integration of robots in industrial settings, and this trend is now well-established. The increasing presence of robots can be attributed to the enhanced adaptability and flexibility that these machines have gained through recent innovations. Advances in computational power, coupled with the compact size of a diverse range of sensors, have enabled the development of autonomous robots with increasingly improved performance.

A collaborative robot (cobot) is a robot that physically interacts with humans in a workspace. The robot and the operator share the same working space and they have to cooperate to complete the desired task [94]. The term collaborative robots or 'cobot' refers to any type of collaborative robot, which can be mobile, stationary, or anthropomorphic. The human-robot collaboration allows to use the of human decisions with the robot's strengths, or it allows the robot to complete some tasks under the operator's control, that are potentially dangerous for the operator's safety. The task repeatability, the physical strength required, a high-speed process, and other requirements can decrease the operator work quality; with a cobot, it is possible to overcome the human limitations [95].

The rise of Industry 4.0 and its innovative technologies have opened doors for a wide range

[1] https://mywelder.it-i.it

of applications in PdM, including the field of robotics. One of the main needs is to identify failures in robots that operate in work cycles involving physical contact, such as collaborative robots [96], which often rely on physical interaction to detect potentially hazardous situations for the operator.

PdM utilizes machine learning to foresee breakdowns before they occur, helping to save costs by avoiding unnecessary repairs and reducing costly downtime [65]. PdM is based on data collected from the system itself to evaluate its condition. The advent of Industry 4.0, along with IoT and machine learning, has made PdM a viable solution [45]. IoT connects various devices to the internet, assigning them unique digital identifiers, which allows data sharing within a centralized network. These devices can include sensors that monitor aspects like system health, vibration, temperature, and other critical factors [12]. The extensive data collected facilitates advanced maintenance analysis, enabling the prediction and resolution of potential issues before they become serious.

The emergence of Industry 4.0 and its cutting-edge technologies has created opportunities for diverse applications in PdM, particularly in robotics. Prior studies have examined the use of different data types for PdM in robotic systems, including electrical power consumption, end-effector positions [97], motor current signals [98], and historical fault records [99]. In [100], PdM is applied to a welding robot by analyzing various data compared to our proposed system, achieving different goals.

The proposed diagnostic approach focuses on integrating data from the robot's operating cycle and its control within an IoT framework. This combined data feeds into a system capable of identifying faults [101]. Additionally, a user support approach has been implemented to minimize the production of defective parts and assist in production decision-making.

Our approach to PdM extends beyond conventional component failure prediction. We also consider a failure to occur when the robot fails to achieve the desired outcome on a workpiece, resulting in product rejection. This can occur in two scenarios: first, if the process is interrupted due to a malfunction in the robotic system (triggering an alarm); and second, if the process completes but the final product does not meet quality standards.

7.2 System setup

The system is designed as a collaborative robotic solution specifically for robot-assisted MIG/MAG welding applications previously described in [92]. The system is visible in Figure 23.

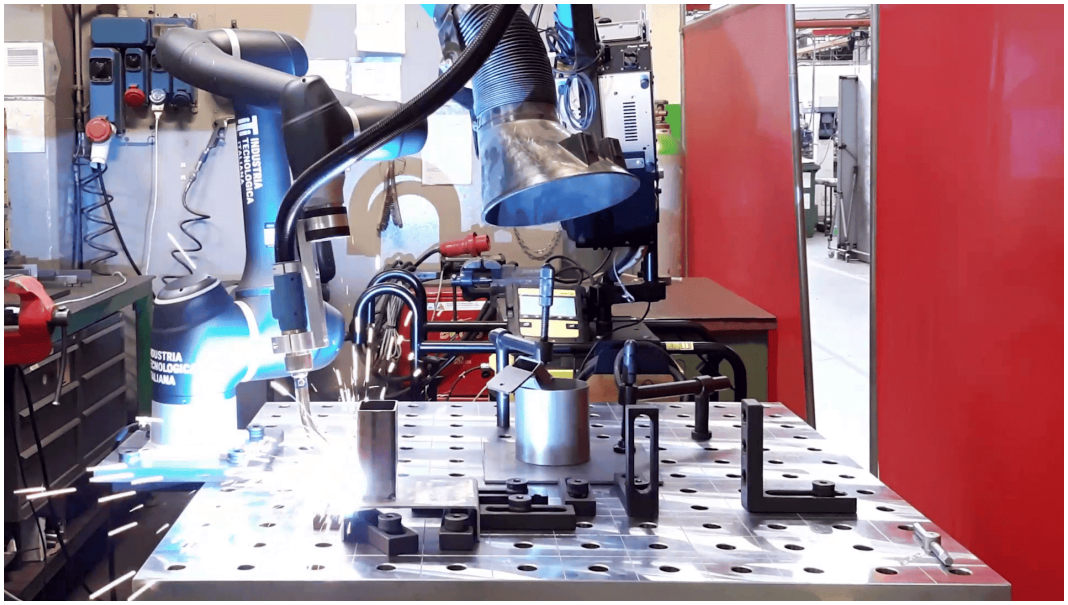


Figure 23: System photo.

The system comprises three primary components that work seamlessly together: first, there is the collaborative robot itself, equipped with a fixing flange to hold and manipulate the welding torch. This flange is essential for securely holding and manipulating the welding torch during the welding process, allowing for precise and effective operations. In this implementation is used a Doosan[2] collaborative robot with six degrees of freedom.

Second, the system includes a human-machine interface designed for ease of use, which is installed on an external industrial mini PC. This interface facilitates straightforward programming of the robot and communicates with the robot controller using Modbus, a standard protocol widely used in industrial environments for communication between devices. This connectivity ensures that operators can interact with the robot's functions efficiently and conveniently from the robot's teach pendant.

Finally, the system encompasses the necessary welding equipment, which includes the welding power source, the torch, and a stable work table. The integration of these components allows

[2]https://www.doosanrobotics.com

for a comprehensive welding solution that can be easily managed.

One of the standout features of this system is an intuitive and user-friendly graphical user interface (GUI) that significantly streamlines the process of programming the robot. Operators can input welding instructions directly through the human-machine interface, which eliminates the complexities typically associated with coding or creating intricate instruction blocks and enabling them to effectively utilize the system.

This GUI is designed to make robot programming accessible and efficient, even for users without extensive technical experience. The interface presents all the available movement primitives, which are fundamental building blocks that can be combined to automate even the most complex welding trajectories. Each movement primitive is defined by interpolation through a specific sequence of via-points, which can be conveniently selected by the operator using the GUI.

The movement primitives available in the system include common types such as single points, straight lines, circles, and arcs, which are frequently used in industrial welding applications. For instance, when the operator wants to implement a linear movement, they can simply select the start and end points of the movement. These points are then saved into the robot's program, and the robot automatically interpolates the trajectory, executing the movement smoothly between the two specified points. The operator can further combine different types of movement primitives, such as linear paths, circular arcs, and others, to create a welding path that meets the desired specifications for the job. Then welding information related to the welding torch power, and the type of welding can be defined. The operator can save the welding path obtained by the combination of the desired primitives available; this information is saved in a specific welding program and allows the operator to run the same program on different products without redefine the welding path. This allows the production of batches of the same products.

Once the operator has defined the complete welding path using the interface, the robot is capable of executing the programmed welding task autonomously. This autonomy not only increases efficiency but also enhances safety by reducing the need for human intervention during the welding process. Overall, this collaborative robotic system represents a significant advancement in the field of industrial welding, combining innovation, user accessibility, and operational efficiency.

In addition to simplifying robot programming, the system also includes two distinct simula-

tion modes that help operators test and verify their programs without activating the actual welding machine. The first simulation mode provides a comprehensive, full-program simulation with the welding machine turned off. This allows the operator to visualize and inspect the movements of the robot, ensuring that the entire program functions as expected before execution. The second simulation mode is a step-by-step approach, in which the operator manually triggers each movement within the program individually. This mode is especially useful for diagnosing and refining specific sections of the program, offering greater control over the testing process.

While these simulation modes greatly assist in program validation, they also introduce some challenges. During the simulation, the system is capable of detecting and logging any alarms that arise. However, pinpointing the exact movement or part of the program that caused a specific alarm can sometimes be difficult, requiring additional troubleshooting and analysis by the operator to resolve the issue. This aspect of the system could benefit from further enhancements to improve alarm diagnostics and troubleshooting efficiency.

7.3 IoT node application

The IoT node architecture is represented in Figure 24.

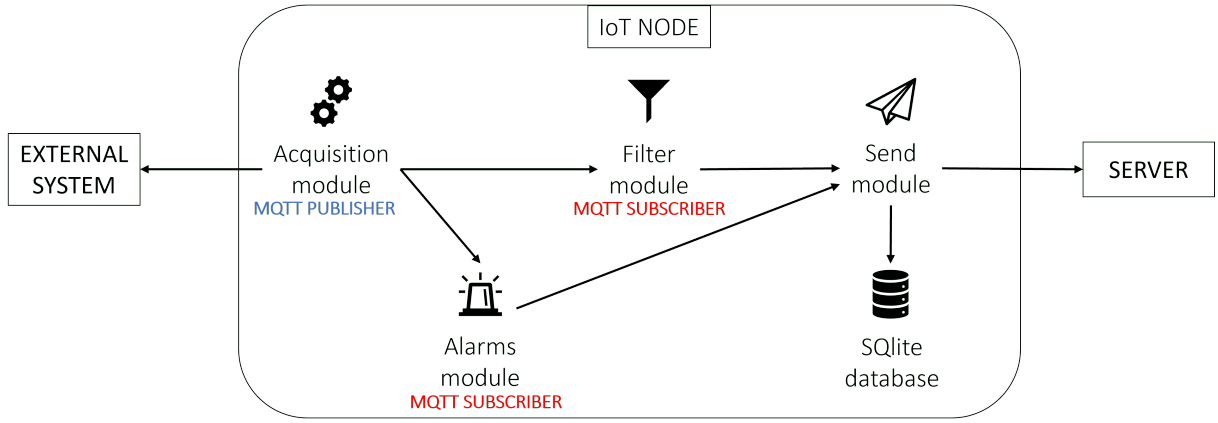


Figure 24: Software architecture for the IoT node in the welding application.

Considering the complete IoT node architecture reported in Figure 4, this one has a lower number of modules because they are not requested in this implementation. The data are generated and published by the software that manages the welding machine, the robot and the user-friendly interface where the user programs the system. Two possible categories of data can be generated, that are the functioning data and the alarms. The functioning

data are the data correlated with the welding process, the welding information, etc. The alarms are the robot alarms, with their specific information. Both these categories of data are published in MQTT and in the IoT software module there is an MQTT subscriber that collects the data filters them, and prepares them for the "Send module". The "Send module" sends them to the cloud server via REST API protocol.

The node system architecture implemented in the specific robotic cell is reported in Figure 25. The acquisition data are reported.

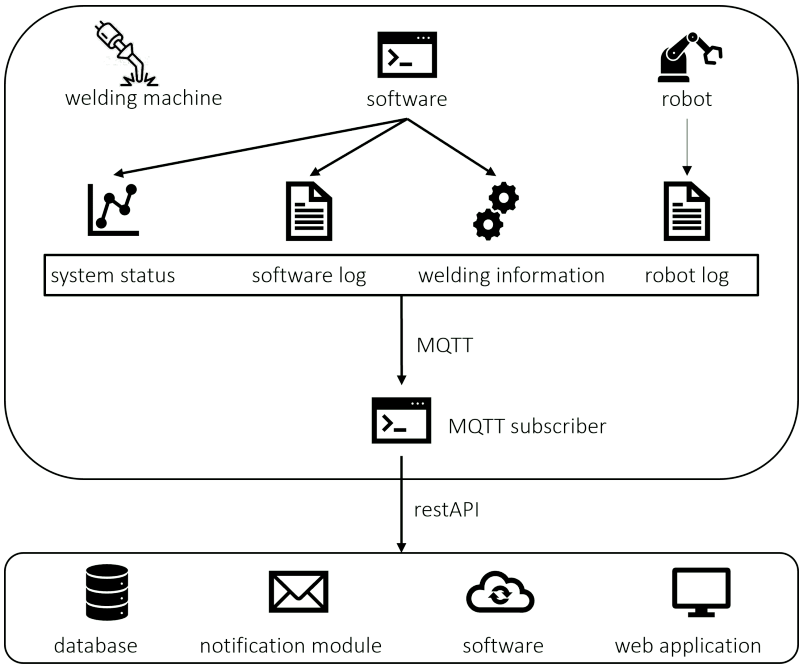


Figure 25: Software architecture implemented in the specific robotic cell.

For this application, this portion of the code is embedded in an industrial mini PC, which is the same one that hosts the software for the welding process. The alarms available in this use case are the "system alarms", and in particular the robot's alarms are considered system alarms. Those data are collected from the robot log visible in Figure 25. The "user alarms" are not implemented because in this use case, the main focus is on the robot and its information, which is the most difficult element in the system for the non-expert user to understand.

The information collected for every robot alarm concerning it are:

- The alarm ID;
- The robotic cell;

- Timestamp;
- Generic information available related to the alarm.

In the MyWelder application, the operator can save some welding programs, composed of different operations. The operations available are:

- Single point;
- Line;
- Arc;
- Circle.

The user chooses the movement necessary and then moving the robot following the expected trajectory. The MyWelder software saves those movement and the robot position, to be able to perform it. The data received are the cell status, the program information and the operation information. The cell status information contains:

- Status of the cell: It can be different values with different meanings, and the values are obtained by mixing the robot status with the welding status. Those values are:
  - Starting: the system is turned on.
  - Stopped: the system is turned off.
  - Initialization: the system initializes some parameters necessary for its operation, it occurs after the starting state.
  - Standby: the system is waiting commands in standby mode.
  - Moving: the robot receives a moving command and it is moving without the operator manually moving it.
  - Teaching: the robot moves because the operator is manually moving it in free drive.
  - Play: a myWelder program is executed.
  - Homing: the robot moves to the home position, usually to perform a joint calibration.
  - Emergency stop: the robots stops because the emergency button is clicked.
  - Safe stop: the robot stops the execution of the movement due to a fatal error. It

is a safety state.

  - Safe off: this state occurres after the safe stop state and the robot turns off its engines.
  - Safe stop 2: it is the same of safe stop, but it requires a recovery.
  - Safe off 2: it is the same of safe off, but it requires a recovery.
  - Recovery: after a fatal error that moves the robot out of its operational range, recovery allows the robot to move inside the operational range. It is permormed normally after a safe off 2 state.

- Start time: it is the time when the status starts.
- End time: it is the time when the status ends.

The program information is received when the program execution finishes. It contains:

- Program name: it is the name that the user gives to a specific program.
- Product id: it is the counter of the product for the specific program in the day.
- Program execution result: the program execution can be a success or not, and this field contains 1 when the execution is successful, 0 when something happens that does not allow to finish the program execution.
- Type of execution: it is possible to execute the program in three different modalities:
  - Welding execution: in this way, the welding program is executed with the welding machine on.
  - Continuous simulation: the welding program is executed, but the welding machine is off; the scope is to see how the robot moves.
  - Step by-step simulation: a specific movement of the welding program is executed, and at the end of the movement simulation the operator can choose to execute the next one or stop the simulation.
- Start time: it is the time when the program starts.
- End time: it is the time when the program ends.

The program information is received when the program execution finishes. It contains:

- Program name: it is the name of the program that has this movement.

- Movement type: it can be a single point, line, circle, or arc.
- Movement execution result: the operation execution can be a success or not, and this field contains 1 when the execution is successful, 0 when something happens that does not allow to finish the operation execution.
- Type of execution: it is the same value reported in the program "type of execution".
- Start time: it is the time when the operation starts.
- End time: it is the time when the operation ends.

7.4 Server application

The web map is here reported Figure 26. Making a comparison with Figure 6, some pages are not useful and are not implemented in this project.

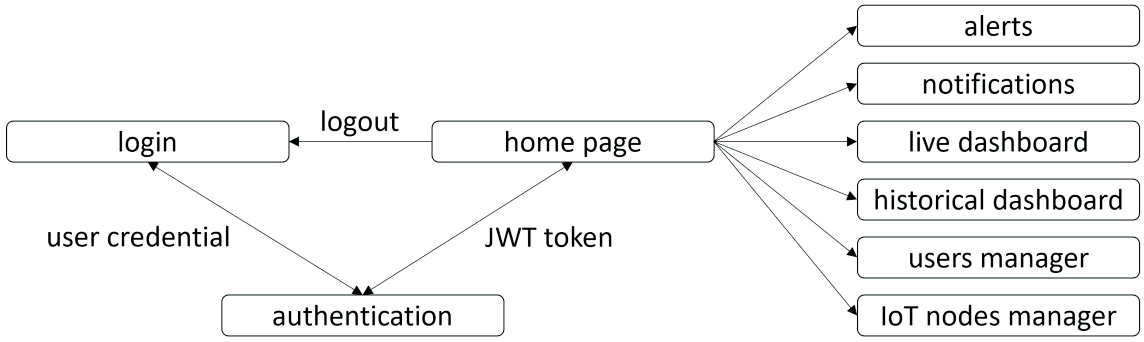


Figure 26: Website map of the HMI application.

In the following, the pages that are different from the page described in the previous use case are reported.

7.4.1 Alerts page

The alerts or alarms page is reported in Figure 27. The data from the database shows the alarms collected with their description, the alarm ID, the robotic cell where the alarms occurred, the date and the hour when the alarm was triggered and the alarm typology. The alarm ID is the same as the one indicated in the troubleshooting manual. The data are shown from the newest to the oldest, to be able to do a comparison with the past data, to highlight better what is happening at the moment. It is possible to download the data in a CSV format to allow the user to make an analysis.

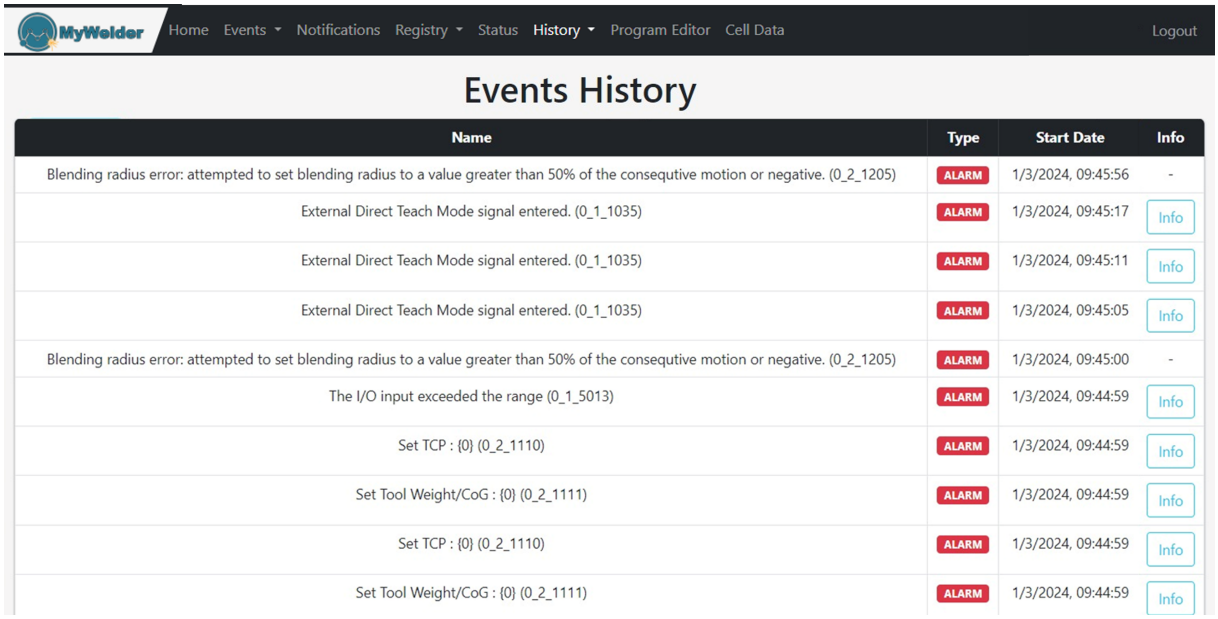


Figure 27: Alarms page of the web application.

The system includes an "Info" button that provides the operator with more detailed information regarding a specific alarm. When the operator clicks on this button, a new window appears, offering a comprehensive breakdown of the alarm, how is illustrated in Figure 28.

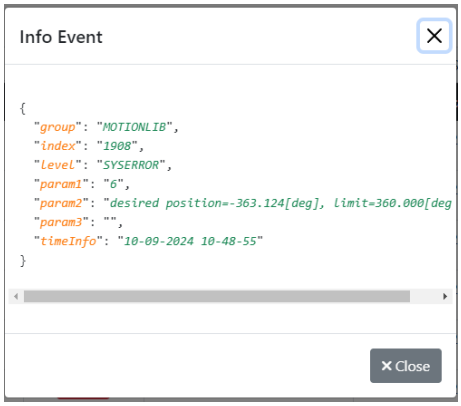


Figure 28: Alarms Info button structure.

This additional information can be extremely helpful for the operator, as it offers deeper insights into the nature of the problem that triggered the alarm. By providing more context about the specific issue, the Info window enables the operator to better understand the underlying cause of the alarm, allowing them to take the appropriate corrective actions. This feature serves as a valuable troubleshooting tool, as it not only helps in diagnosing the issue but also guides the operator in making informed decisions about how to resolve the problem effectively.

7.4.2 Notification page

In Figure 29 the notification page is reported. There is the possibility to create more notifications that are represented as cards with all the info associated. The user can create the notification and decide to enable or disable it whenever he needs it.

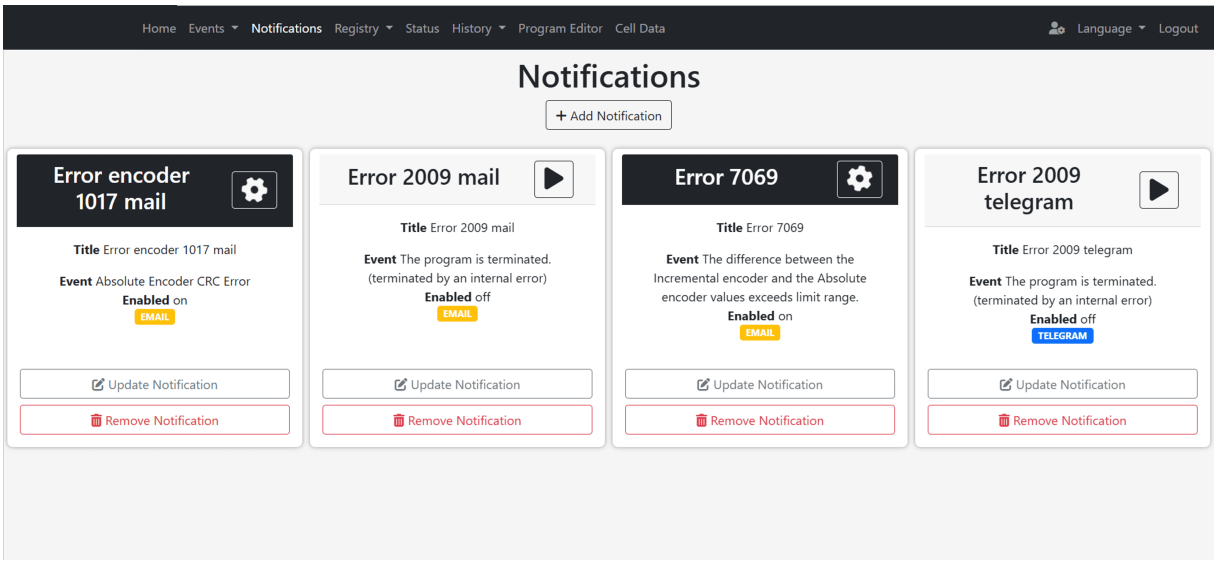


Figure 29: Notification page of the web application.

7.4.3 Historical data page

The historical dashboard page is splitted into four different sections, that are:

- Cell history page.
- Program history page.
- Operations history page.
- OEE chart page.

This division allows to split the different data and help the operator in understanding specific robot activities permomed.

7.4.3.1 Cell history page

The cell history page is reported in Figure 30. The user can select the desired robotic cell and all the historical data of the status cell are shown.

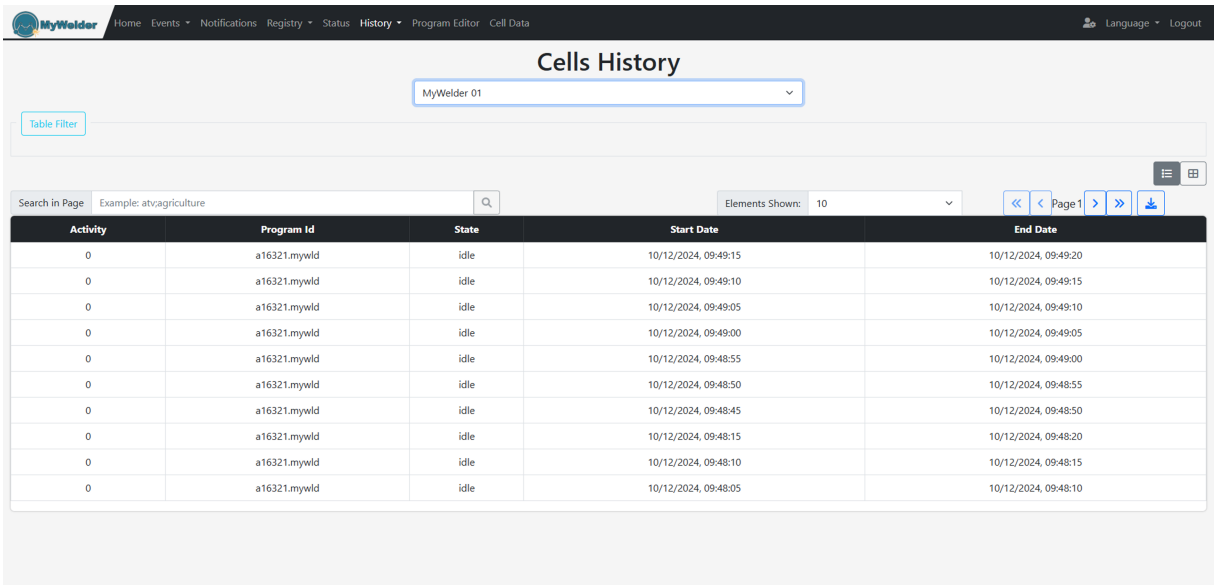


Figure 30: Cell status historical dashboard page of the web application.

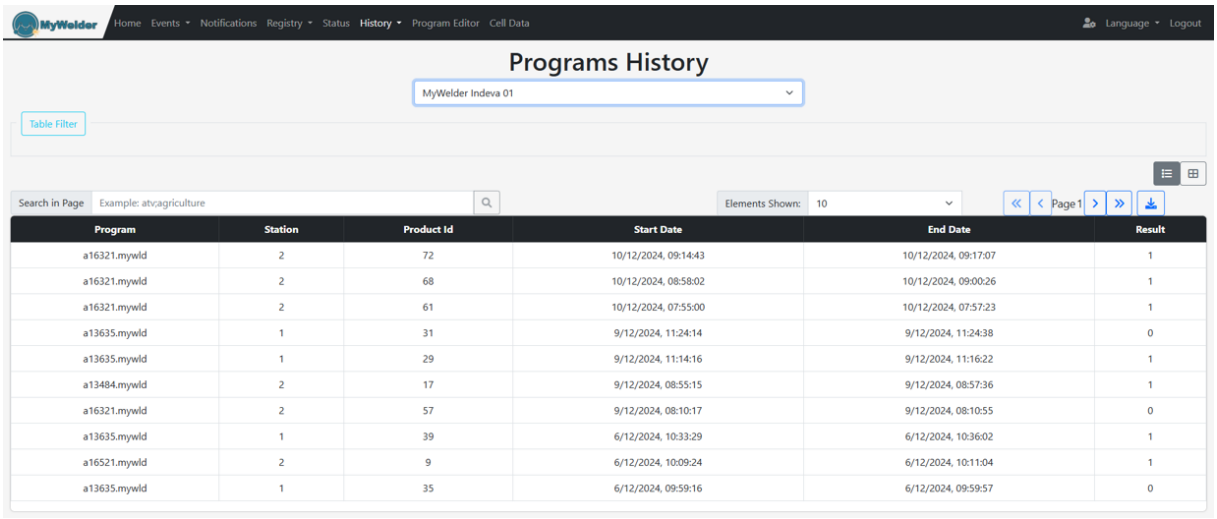
As visible in the figure, the data collected are:

- the Program ID: it is the program executed in the specific time interval, and it can be executed in simulation mode or during the welding process.
- the cell state: the cell states are the same as previously described.
- the time interval: when the state is collected, the beginning time and the finish time of the state are collected too.

The cell state is collected every 5 seconds and whenever the state changes. This allows us to know exactly when a new state is reached.

7.4.3.2 Program history page

The program history page contains all the programs executed by a specific robotic cell and is shown in Figure 31.

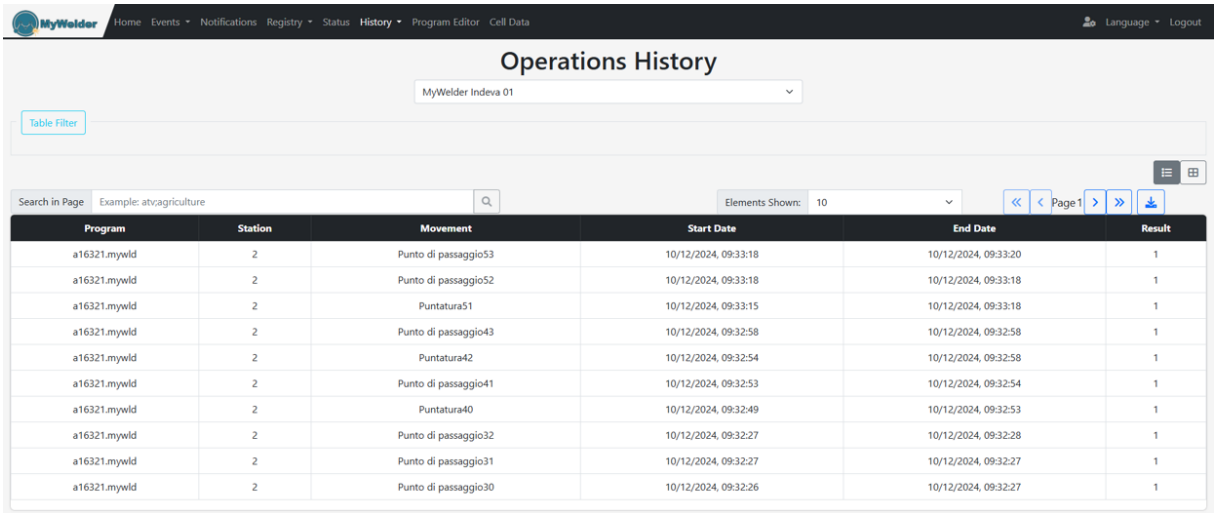


Program	Station	Product Id	Start Date	End Date	Result
a16321.mywld	2	72	10/12/2024, 09:14:43	10/12/2024, 09:17:07	1
a16321.mywld	2	68	10/12/2024, 08:58:02	10/12/2024, 09:00:26	1
a16321.mywld	2	61	10/12/2024, 07:55:00	10/12/2024, 07:57:23	1
a13635.mywld	1	31	9/12/2024, 11:24:14	9/12/2024, 11:24:38	0
a13635.mywld	1	29	9/12/2024, 11:14:16	9/12/2024, 11:16:22	1
a13484.mywld	2	17	9/12/2024, 08:55:15	9/12/2024, 08:57:36	1
a16321.mywld	2	57	9/12/2024, 08:10:17	9/12/2024, 08:10:55	0
a13635.mywld	1	39	6/12/2024, 10:33:29	6/12/2024, 10:36:02	1
a16521.mywld	2	9	6/12/2024, 10:09:24	6/12/2024, 10:11:04	1
a13635.mywld	1	35	6/12/2024, 09:59:16	6/12/2024, 09:59:57	0

Figure 31: Program historical dashboard page of the web application.

7.4.3.3 Operations history page

The operations history page contains all the operations executed by a specific robotic cell and is shown in Figure 32.



Program	Station	Movement	Start Date	End Date	Result
a16321.mywld	2	Punto di passaggio53	10/12/2024, 09:33:18	10/12/2024, 09:33:20	1
a16321.mywld	2	Punto di passaggio52	10/12/2024, 09:33:18	10/12/2024, 09:33:18	1
a16321.mywld	2	Puntatura51	10/12/2024, 09:33:15	10/12/2024, 09:33:18	1
a16321.mywld	2	Punto di passaggio43	10/12/2024, 09:32:58	10/12/2024, 09:32:58	1
a16321.mywld	2	Puntatura42	10/12/2024, 09:32:54	10/12/2024, 09:32:58	1
a16321.mywld	2	Punto di passaggio41	10/12/2024, 09:32:53	10/12/2024, 09:32:54	1
a16321.mywld	2	Puntatura40	10/12/2024, 09:32:49	10/12/2024, 09:32:53	1
a16321.mywld	2	Punto di passaggio32	10/12/2024, 09:32:27	10/12/2024, 09:32:28	1
a16321.mywld	2	Punto di passaggio31	10/12/2024, 09:32:27	10/12/2024, 09:32:27	1
a16321.mywld	2	Punto di passaggio30	10/12/2024, 09:32:26	10/12/2024, 09:32:27	1

Figure 32: Operations historical dashboard page of the web application.

7.4.3.4 OEE chart page

This page shows the Overall Equipment Effectiveness (OEE) [102] of the cell. The OEE is a metric that is used to understand the efficiency of the system and is used in a specific time range. The user selects the robotic cell of interest and the range time via a modal dialogue similar to that shown in Figure 10. Then the server computes the OEE of the system and shows some metrics to the user, the activity done by the cell in the specific time interval and the OEE breakdown.

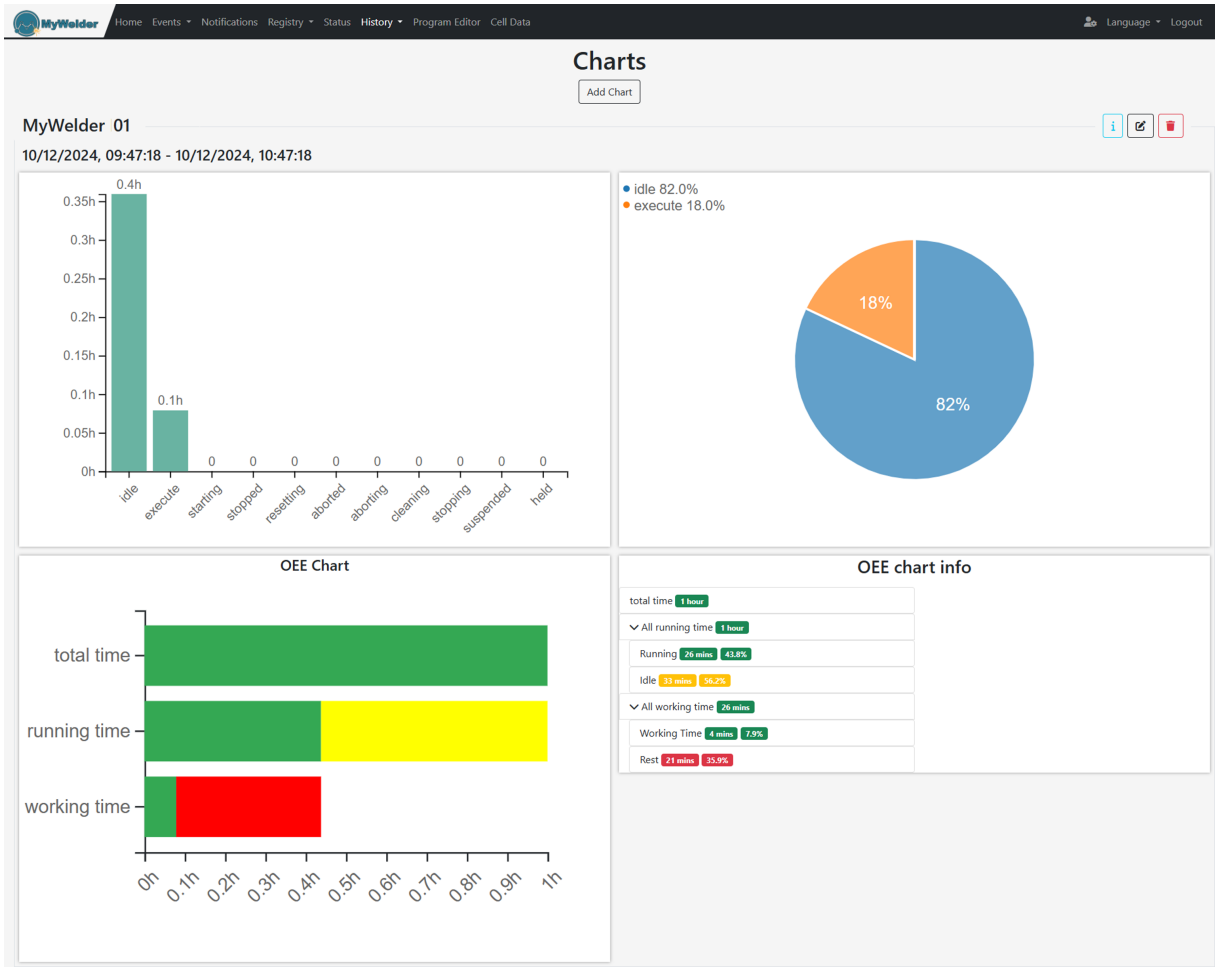


Figure 33: OEE dashboard page of the web application.

7.4.4 Live data dashboard page

The live data dashboard page contains all the information regarding the last data received from a specific cell, as shown in Figure 34.

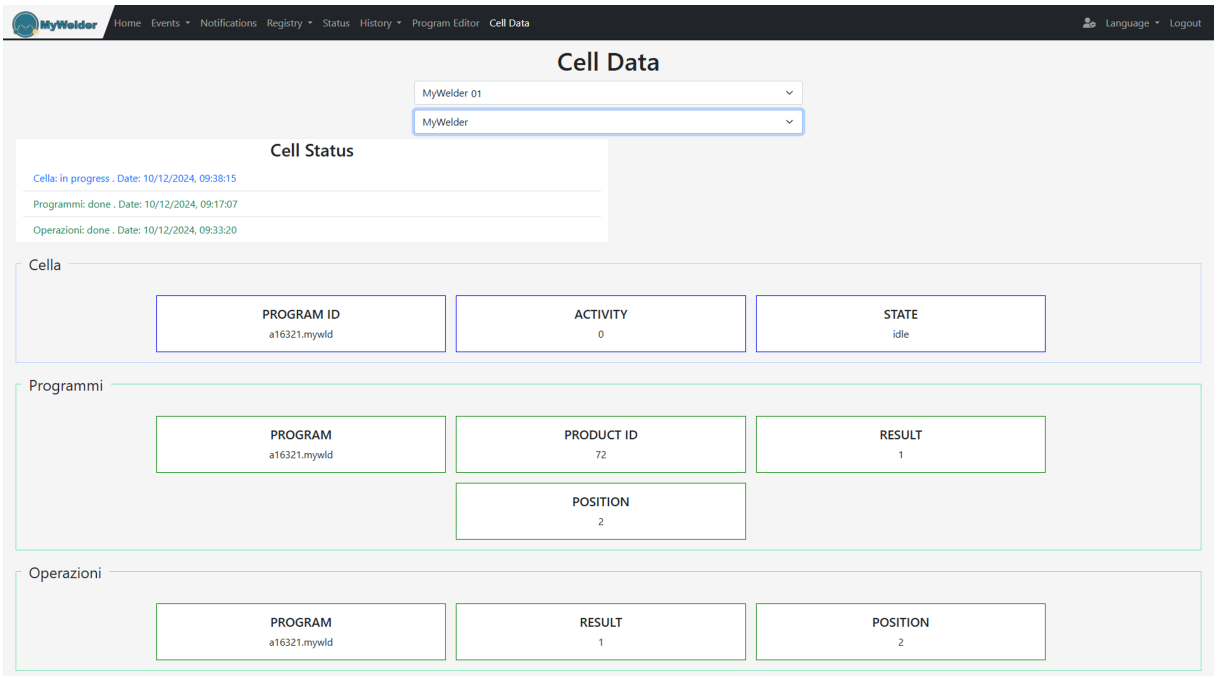


Figure 34: Live dashboard page of the web application.

7.5 Fault analysis

During welding operations, it is not uncommon for the robot to halt unexpectedly due to various faults or issues that can arise during its execution. One of the most critical and disruptive scenarios is when the robot stops executing the program entirely. For operators who may not have extensive expertise in robotics, it can be extremely difficult to quickly identify the underlying cause of the robot's stop. The nature of robotic systems, with their complex components and intricate functions, makes it challenging to diagnose such faults without a detailed understanding of the robot's inner workings. Traditionally, to identify the cause of an error, the operator would need to manually access the robot's log files, a process that is not only technically demanding but also time-consuming, because during this process the robot is stopped and it is not possible to perform other operations. This adds to the complexity of troubleshooting, particularly for those who may not be familiar with interpreting log data.

There are various potential faults that can lead to the robot's program being halted, and some of the most common errors include issues like a physical collision, the robot's joints reaching or exceeding their defined limits, or problems with the encoders, such as a CRC error. Each of these errors results in a stop of the robot's operation, but they may occur for

entirely different reasons, which complicates fault diagnosis. For example, a collision might happen if the robot's arm unexpectedly interacts with an object in its environment, while an encoder CRC error suggests a communication problem or data corruption with the robot's encoder, which measures the joint's position. These errors, though leading to the same result - a stopped robot - can have very different causes and solutions.

The software architecture proposed in this system effectively addresses this issue by eliminating the need for operators to manually access the robot log files. Instead, the system is designed to automatically collect and interpret alarms that are generated when a fault occurs. As shown in Figure 27, when an error occurs, an alarm is triggered, and the system not only halts the robot's operation but also provides detailed feedback about the nature of the fault. This functionality is particularly useful because it allows the operator to immediately understand the specific fault that caused the robot to stop, rather than having to troubleshoot blindly or rely on external technical support.

For example, as illustrated in Figure 27, the system can identify that the robot's stoppage was caused by a Blending radius error. This clear identification means that the operator does not need to spend time searching through logs or performing complex diagnostics. In a different occasion, the system might report that the robot has stopped due to a collision error, providing the operator with enough information to understand what happened and begin addressing the issue. This approach significantly simplifies the troubleshooting process, as the operator can directly see the cause of the problem through the user-friendly interface, without needing to delve into the robot's technical log files.

Ultimately, the proposed system enhances the overall user experience by making fault diagnosis more accessible and efficient. It empowers operators to quickly and accurately identify issues, which reduces downtime, minimizes the potential for errors, and increases the overall productivity of the robotic system. The ability to diagnose faults directly through the interface also improves the efficiency of repairs and maintenance, contributing to smoother and more reliable robot operation.

Another feature to diagnose the fault and prevent them is implemented, regarding the singularity error. A singularity occurs when the robot's configuration limits its movement, and it prevents the end-effector from precise manoeuvring [103]. During the welding of a product, a singularity can alter the trajectory of the robot's end-effector, leading to an incorrect weld and, as a result, causing a defective product that must be discarded. Many techniques can be

implemented to manage singularities. In the Doosan cobot used for this application, when a singularity occurs, the robot attempts to overcome it by maintaining a trajectory as close as possible to the planned path, rotating around the singularity. There are other possibilities that can be selected, but we prefer to use this solution.

In the presented system, the operator creates a new welding program and verified its functionality through a step-by-step simulation process. The system allowed the operator to review the program's performance by prompting for confirmation before executing each individual movement. Initially, the system did not flag any potential singularities in the program, which are positions or configurations where the robot's kinematics could cause it to lose degrees of freedom, often resulting in erratic or unpredictable behaviour. Despite the absence of warnings, a singularity did occur during the second movement of the program. However, the robot was able to continue executing the program successfully because it was equipped with the capability to overcome the singularity. The robot achieved this by adjusting its pre-planned trajectory, effectively navigating around the singularity and completing the program without any further interruptions.

A significant challenge arises when a singularity occurs during the actual execution of the program while the welding machine is active. In such cases, diagnosing the problem becomes particularly difficult. This is especially problematic because the presence of a singularity during a welding operation can lead to suboptimal or even defective welds. If the issue is not identified and resolved promptly, it can result in scrapped products, which wastes both materials and time. Furthermore, the operator may be forced to intervene manually, potentially requiring them to stop the process, analyze the robot's movements, and correct the program, all of which introduce delays and inefficiencies.

To prevent from this situation, and in order to avoid the product discharge, a new strategy was implemented. The system architecture proposed in this paper provides a significant improvement over traditional methods by addressing these challenges. The system is capable of detecting singularities during the execution of the program and logging them in the database for further analysis. This feature ensures that the operator is made aware of any potential issues that could affect the robot's performance, enabling them to take proactive measures before the issue leads to product defects or downtime. Identifying and recording these singularities, the system helps prevent situations in which welding defects could result from unnoticed kinematic issues.

However, despite these advancements, a key hurdle remains, particularly for operators who lack extensive expertise in robotics. While the system provides detailed information about the occurrence of singularities, interpreting the specific movement or part of the program responsible for the singularity can still be challenging. Non-expert operators may struggle to understand the exact cause of the issue, making it difficult for them to modify or fix the program appropriately. This underscores the need for further improvements in the system's diagnostic tools, potentially including more intuitive feedback that helps guide operators in identifying and addressing singularities more effectively. By enhancing the system's ability to present clearer, more actionable information, it could empower operators with the tools they need to resolve these issues independently, reducing reliance on specialized knowledge and improving the overall efficiency of the welding process.

While the proposed system offers the capability to notify users about singularities via web alerts, it is important to note that industrial environments often restrict operator access to the web during operational activities. This limitation can make it difficult for operators to receive these alerts in real-time. Additionally, operators may overlook or miss these notifications entirely, which could result in undetected errors affecting the welding process. To overcome these challenges, we implemented a more efficient and practical solution.

The system already has the ability to detect singularity errors, but the new implementation further enhances this feature by performing a check for singularities at the end of each movement. This allows the system to notify the operator immediately if a singularity is detected, and this notification appears directly during the simulation phase, as illustrated in Figure 35. The notification is presented as a pop-up window within the graphical user interface (GUI), ensuring that the operator is informed of the issue without having to rely on external web-based alerts.

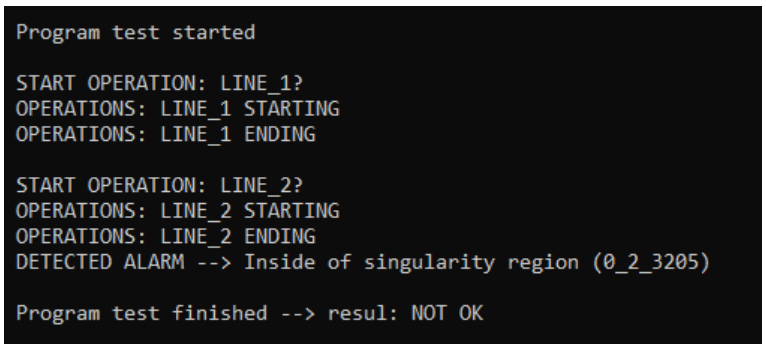


Figure 35: Command prompt of the execution of simulation step-by-step with singularities real-time detection.

This real-time feedback feature greatly improves the operator's ability to identify and address problematic movements as they arise. By being notified immediately during the simulation, the operator can take corrective actions right away, such as repositioning the workpiece, modifying the operation sequence, or making other necessary adjustments to the program. This proactive approach helps to prevent errors that could lead to defective welds or scrapped products, improving the efficiency and reliability of the welding process.

Looking ahead, we plan to enhance this feature further by not only notifying the operator of the occurrence of a singularity but also providing additional suggestions related to the specific fault. These recommendations will guide the operator on how to overcome the issue, offering actionable steps for resolving the singularity and preventing future occurrences. This enhancement will be extended to cover all types of alarms, not just singularities, allowing for more comprehensive troubleshooting and support throughout the operation.

In the case of a singularity, for example, the pop-up window will also include advice on how to adjust the pose of the workpiece on the table. In the specific test scenario we conducted, the system alerted the operator to change the position of the workpiece in the event of a singularity. The operator promptly responded by halting the program execution and adjusting the orientation of the workpiece on the table, demonstrating how the system's real-time notifications can effectively guide operators in preventing errors and improving the overall quality of the welding process. This implementation not only empowers the operator with timely information but also helps ensure more accurate and efficient operation of the robotic system.

7.6 Conclusion

Collaborative robots are becoming more present in industrial environments to complete tasks that were done by the operator. Their use increases the operator's safety, but it can increase the technical knowledge required. MyWelder is an industrial application where the cobot is used to perform a welding procedure, and its advantage is the easy GUI that supports the operators during their task. The functioning data of the robotic cell are collected, stored and now available to be analysed. The fault analysis section describes how the system can be diagnosed by the operator when something unexpected happens and how the operator's empowerment is reached thanks to this application.

8. Conclusions

Industry 4.0 integrates advanced technologies such as big data, autonomous robots, the IoT, and additive manufacturing to enhance production and automation. While Industry 4.0 is driven by technological advancements, Industry 5.0 is value-driven, prioritizing human well-being, ethics, and sustainability. Industry 5.0 seeks to address the limitations of Industry 4.0 by emphasizing human-machine collaboration, sustainability, personalization, and social innovation. It focuses on human well-being, ethical production, and the integration of AI and advanced robotics.

The aim of this research, according to Industry 5.0, is to present a fault methodology that can be easily implemented by non-expert operators, facilitated by the proposed industrial IoT architecture.

The IoT refers to a network of physical objects, that are equipped with sensors, software, actuators, and other technologies. The primary objective is to enable these objects to connect with one another and exchange data with other devices and systems via the internet or other communication networks. Various wireless technologies can be used in IoT architectures, such us internet, RFID, ZIBgee, Bluetooth, etc. The choice of a particular wireless network protocol depends on the specific problem domain. Since there is no universal architecture that fits all IoT applications, this lack of standardization can slow down the growth of IoT.

A main acticity in the industrial field is maintenance and system monitoring to prevent failures and minimizing downtime. The key maintenance strategies have been outlined to help readers grasp the role of maintenance and its effects on system performance. Fault diagnostics continue to be a critical concern, and ongoing research is focused on finding solutions to industrial challenges. The proposed methodology aims to contribute to this effort.

Then the proposed IoT architecture is presented, detailing both the IoT node and IoT server structures. To better understand the system's functionality it is applied to two different industrial use cases.

The first use case is implemented with AGVs. AGVs are commonly used in industrial set-

tings to transport materials and products across various locations. While many studies in the literature focus on monitoring the health status of AGV system components to predict failures, there has been less emphasis on understanding the interaction between the AGV and its environment. In this section, the proposed IoT architecture and the new diagnostic methodology are applied to explore this relationship. The section outlines the technical implementation of the system and explains how operators can easily access AGV data through a GUI that is accessible from any location. The fault analysis section illustrates how the system can be utilized by operators, emphasizing how the application enhances the operator's ability to manage and diagnose the system effectively.

The second use case is implemented with cobots; they are increasingly utilized in industrial environments to perform tasks traditionally carried out by operators. While their use enhances operator safety, it may also introduce additional technical complexity. MyWelder is an industrial application where a cobot is employed to carry out a welding procedure, offering the benefit of an intuitive GUI that assists the operator throughout the task. The operational data from the robotic cell are collected, stored, and are now available for analysis. The fault analysis section explains how the system can be diagnosed by the operator in the event of an unexpected issue, highlighting how the application empowers the operator to effectively manage and troubleshoot the system.

In the future, it will be interesting to observe how these applications have influenced the work of operators, and whether it will be possible to develop new features that can better support them in their daily tasks. Furthermore, it will be important to define other emerging failures, understand their causes, and find effective ways to prevent them. Another significant opportunity is to apply the architecture to new industrial cases, which would not only allow for further improvement of the system but also enhance its flexibility, making it even more versatile and adaptable to a wide range of contexts and industries. This could lead to the creation of a more robust, universally applicable solution capable of meeting diverse operational challenges.

Bibliography

[1] R. Kurt, «Industry 4.0 in terms of industrial relations and its impacts on labour life», *Procedia computer science*, vol. 158, pp. 590–601, 2019.

[2] O. Bayraktar and C. Ataç, «The effects of industry 4.0 on human resources management», *Globalization, institutions and socio-economic performance*, pp. 337–359, 2018.

[3] Y. Guo, C. Langer, F. Mercorio, and F. Trentini, «Skills mismatch, automation, and training: Evidence from 17 european countries using survey data and online job ads», in *EconPol Forum*, Munich: CESifo GmbH, vol. 23, 2022, pp. 11–15.

[4] L. Bonekamp and M. Sure, «Consequences of industry 4.0 on human labour and work organisation», *Journal of business and media Psychology*, vol. 6, no. 1, pp. 33–40, 2015.

[5] P. Rikala, G. Braun, M. Järvinen, J. Stahre, and R. Hämäläinen, «Understanding and measuring skill gaps in industry 4.0—a review», *Technological Forecasting and Social Change*, vol. 201, p. 123 206, 2024.

[6] A. Sharma and B. J. Singh, «Evolution of industrial revolutions: A review», *International Journal of Innovative Technology and Exploring Engineering*, vol. 9, pp. 66–73, 11 2020.

[7] M. Xu, J. M. David, and S. H. Kim, «The fourth industrial revolution: Opportunities and challenges», *International journal of financial research*, vol. 9, pp. 90–95, 2 2018.

[8] A. Rojko, «Industry 4.0 concept: Background and overview.», *International journal of interactive mobile technologies*, vol. 11, pp. 77–90, 5 2017.

[9] X. Xu, Y. Lu, B. Vogel-Heuser, and L. Wang, «Industry 4.0 and industry 5.0—inception, conception and perception», *Journal of Manufacturing Systems*, vol. 61, pp. 530–535, 2021.

[10] R. Ziatdinov, M. S. Atteraya, and R. Nabiyev, «The fifth industrial revolution as a transformative step towards society 5.0», *Societies*, vol. 14, 19 2024.

[11] M. Rüßmann *et al.*, «Industry 4.0: The future of productivity and growth in manufacturing industries», *Boston consulting group*, vol. 9, pp. 54–89, 1 2015.

[12] S. Vaidya, P. Ambadb, and S. Bhosle, «Industry 4.0 – a glimpse», *2nd International Conference on Materials Manufacturing and Design Engineering*, pp. 233–238, 2018.

[13] G. Erboz, «How to define industry 4.0: Main pillars of industry 4.0.», *Managerial trends in the development of enterprises in globalization era 761*, vol. 761, pp. 761–767, 2017.

[14] D. S. Singh and G. Singh, «Big data-a review», *International Research Journal of Engineering and Technology*, vol. 4, no. 4, pp. 822–824, 2017.

[15] A. C. Pereira and F. Romero, «A review of the meanings and the implications of the industry 4.0 concept.», *Procedia manufacturing*, vol. 13, pp. 1206–1214, 2017.

[16] E. Kaasinen *et al.*, «Empowering and engaging industrial workers with operator 4.0 solutions», *Computers & Industrial Engineering*, vol. 139, p. 105 678, 2020.

[17] D. Romero *et al.*, «Towards an operator 4.0 typology: A human-centric perspective on the fourth industrial revolution technologies», in *proceedings of the international conference on computers and industrial engineering (CIE46), Tianjin, China*, 2016, pp. 29–31.

[18] J. Leng *et al.*, «Industry 5.0: Prospect and retrospect», *Journal of Manufacturing Systems*, vol. 217, pp. 279–295, 2022.

[19] M. Golovianko, V. Terziyan, V. Branytskyi, and D. Malyk, «Industry 4.0 vs. industry 5.0: Co-existence, transition, or a hybrid», *Procedia Computer Science*, vol. 217, pp. 102–113, 2023.

[20] S. Huang, B. Wang, X. Li, P. Zheng, D. Mourtzis, and L. Wang, «Industry 5.0 and society 5.0—comparison, complementation and co-evolution», *Journal of manufacturing systems*, vol. 64, pp. 424–428, 2022.

[21] J. Cotta, M. Breque, L. De Nul, and A. Petridis, «Towards a sustainable, human-centric and resilient european industry», *Luxembourg: Publications Office of the European Union*, 2021.

[22] B. Martini, D. Bellisario, and P. Coletti, «Human-centered and sustainable artificial intelligence in industry 5.0: Challenges and perspectives», *Sustainability*, vol. 16, no. 13, p. 5448, 2024.

[23] M. Ghobakhloo, M. Iranmanesh, M. Fathi, A. Rejeb, B. Foroughi, and D. Nikbin, «Beyond industry 4.0: A systematic review of industry 5.0 technologies and implications for social, environmental and economic sustainability», *Asia-Pacific Journal of Business Administration*, 2024.

[24] A. Raja Santhi and P. Muthuswamy, «Industry 5.0 or industry 4.0 s? introduction to industry 4.0 and a peek into the prospective industry 5.0 technologies», *International Journal on Interactive Design and Manufacturing (IJIDeM)*, vol. 17, no. 2, pp. 947–979, 2023.

[25] A. Grybauskas, A. Stefanini, and M. Ghobakhloo, «Social sustainability in the age of digitalization: A systematic literature review on the social implications of industry 4.0», *Technology in Society*, vol. 70, p. 101 997, 2022.

[26] D. Romero and J. Stahre, «Towards the resilient operator 5.0: The future of work in smart resilient manufacturing systems», *Procedia cirp*, vol. 104, pp. 1089–1094, 2021.

[27] S. Madakam, R. Ramaswamy, and S. Tripathi, «Internet of things (iot): A literature review.», *Journal of Computer and Communications*, vol. 3, pp. 164–173, 2015.

[28] H. Yang, S. Kumara, S. Bukkapatnam, and F. Tsung, «The internet of things for smart manufacturing: A review», *IISE transactions*, vol. 51, no. 11, pp. 1190–1216, 2019.

[29] B. B. Gupta and M. Quamara, «An overview of internet of things (iot): Architectural aspects, challenges, and protocols», *Concurrency and Computation: Practice and Experience*, vol. 32, no. 21, 2020.

[30] E. Sissini, 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.

[31] G. Vieira, J. Barbosa, P. Leitão, and L. Sakurada, «Low-cost industrial controller based on the raspberry pi platform», in *2020 IEEE international conference on industrial technology (ICIT)*, IEEE, 2020, pp. 292–297.

[32] P. Suresh, J. V. Daniel, V. Parthasarathy, and R. Aswathy, «A state of the art review on the internet of things (iot) history, technology and fields of deployment», in *2014 International conference on science engineering and management research (ICSEMR)*, IEEE, 2014, pp. 1–8.

[33] X. Jia, Q. Feng, T. Fan, and Q. Lei, «Rfid technology and its applications in internet of things (iot)», in *2012 2nd international conference on consumer electronics, communications and networks (CECNet)*, IEEE, 2012, pp. 1282–1285.

[34] S. Scanzio, M. Rosani, M. Scamuzzi, and G. Cena, «Qr codes: From a survey of the state-of-the-art to executable eqr codes for the internet of things», *IEEE Internet of Things Journal*, 2024.

[35] S. Safaric and K. Malaric, «Zigbee wireless standard», in *Proceedings ELMAR 2006*, IEEE, 2006, pp. 259–262.

[36] A. M. Lonzetta, P. Cope, J. Campbell, B. J. Mohd, and T. Hayajneh, «Security vulnerabilities in bluetooth technology as used in iot», *Journal of Sensor and Actuator Networks*, vol. 7, no. 3, p. 28, 2018.

[37] C. A. Rosati, A. Cervo, A. Bertoli, M. Santacaterina, N. Battilani, and C. Fantuzzi, «Self-configuring ble deep sleep network for fault tolerant wsn», *IFAC-PapersOnLine*, vol. 55, pp. 193–198, 6 2022.

[38] H. D. Kotha and V. M. Guptaa, «Iot application: A survey», *Int. J. Eng. Technol*, vol. 7, pp. 891–896, 2018.

[39] S. A. Al-Qaseemi, H. A. Almulhim, M. F. Almulhim, and S. R. Chaudhry, «Iot architecture challenges and issues: Lack of standardization», *Future Technologies Conference (FTC)*, pp. 731–738, 2016.

[40] R. A. R. A. Mouha *et al.*, «Internet of things (iot)», *Journal of Data Analysis and Information Processing*, vol. 9, no. 02, p. 77, 2021.

[41] S. G. H. Soumyalatha, «Study of iot: Understanding iot architecture, applications, issues and challenges», *1st International Conference on Innovations in Computing & Net-working (ICICN16), CSE, RRCE. International Journal of Advanced Networking & Applications*, vol. 478, 2016.

[42] Y. N. Malek *et al.*, «On the use of iot and big data technologies for real-time monitoring and data processing», *Procedia computer science*, vol. 113, pp. 429–434, 2017.

[43] J. Lee, B. Bagheri, and H.-A. Kao, «A cyber-physical systems architecture for industry 4.0-based manufacturing systems», *Manufacturing letters*, vol. 3, pp. 18–23, 2015.

[44] L. Da Xu, W. He, and S. Li, «Internet of things in industries: A survey», *IEEE Transactions on industrial informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.

[45] M. Compare, P. Baraldi, and E. Zio, «Challenges to iot-enabled predictive maintenance for industry 4.0», *IEEE Internet of Things Journal*, vol. 7, pp. 4585–4597, 2019.

[46] C. A. Rosati, A. Cervo, and C. Fantuzzi, «Air quality monitoring in a bim model by means of a iot sensors network», in *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, IEEE, 2020, pp. 110–115.

[47] F. Civerchia, S. Bocchino, C. Salvadori, E. Rossi, L. Maggiani, and M. Petracca, «Industrial internet of things monitoring solution for advanced predictive maintenance applications», *Journal of Industrial Information Integration*, vol. 7, pp. 4–12, 2017.

[48] T. Ruppert, S. Jaskó, T. Holczinger, and J. Abonyi, «Enabling technologies for operator 4.0: A survey», *Applied sciences*, vol. 8, no. 9, p. 1650, 2018.

[49] P. Asghari, A. M. Rahmani, and H. H. S. Javadi, «Internet of things applications: A systematic review», *Computer Networks*, vol. 148, pp. 241–261, 2019.

[50] Y. Ran, X. L. Zhou, Y. Wen, and R. Deng, «A survey of predictive maintenance: Systems, purposes and approaches», *IEEE Communications Surveys Tutorials*, pp. 1–36, 2019.

[51] O. Motaghare, A. S. Pillai, and K. I. Ramachandran, «Predictive maintenance architecture», *IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, pp. 1–4, 2018.

[52] T. Zonta, C. A. da Costa, R. R. Righi, M. L. de Lima, E. S. da Trindade, and G. P. Li, «Predictive maintenance in the industry 4.0: A systematic literature review», *Computers Industrial Engineering*, p. 106 889, 2020.

[53] D. S. Mungani and J. K. Visser, «Maintenance approaches for different production methods», *South African Journal of Industrial Engineering*, vol. 24, no. 3, pp. 1–13, 2013.

[54] N. T. Tran, H. T. Trieu, N. H. Ngo, and Q. K. Dao, «An overview of the application of machine learning in predictive maintenance», *Petrovietnam Journal*, pp. 47–61, 2021.

[55] E. Hupjè, *Types of maintenance: How to choose the right maintenance strategy*, 9.

[56] M. M. Hamasha *et al.*, «Strategical selection of maintenance type under different conditions», *Scientific Reports*, vol. 13, no. 1, p. 15 560, 2023.

[57] H. Erbiyik, «Definition of maintenance and maintenance types with due care on preventive maintenance», in *Maintenance Management-Current Challenges, New Developments, and Future Directions*, IntechOpen, 2022.

[58] R. Ahmad and S. Kamaruddin, «An overview of time-based and condition-based maintenance in industrial application», *Computers & industrial engineering*, vol. 63, no. 1, pp. 135–149, 2012.

[59] A. Olsen, «Failure finding maintenance», in *Safety Culture and Leading Indicators for Safety in the Maritime and Offshore Environment*, Springer, 2024, pp. 679–683.

[60] N. Arunraj and J. Maiti, «Risk-based maintenance—techniques and applications», *Journal of hazardous materials*, vol. 142, no. 3, pp. 653–661, 2007.

[61] M. Pech, J. Vrchota, and J. Bednář, «Predictive maintenance and intelligent sensors in smart factory», *Sensors*, p. 1470, 2021.

[62] L. Pinciroli, P. Baraldi, and E. Zio, «Maintenance optimization in industry 4.0», *Reliability Engineering & System Safety*, vol. 234, p. 109 204, 2023.

[63] M. Achouch *et al.*, «On predictive maintenance in industry 4.0: Overview, models, and challenges», *Applied Sciences*, vol. 12, no. 16, p. 8081, 2022.

[64] C. S. Longo, C. Fantuzzi, F. Monica, L. Manfredotti, and M. Sorge, «Big data for advanced monitoring system: An approach to manage system complexity», in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, 2018, pp. 341–346.

[65] S. Vollert, M. Atzmueller, and A. Theissler, «Interpretable machine learning: A brief survey from the predictive maintenance perspective», *IEEE International conference on Emerging Technologies and factory automation (ETFA)*, pp. 01–08, 2021.

[66] A. Bertoli, A. Cervo, C. A. Rosati, and C. Fantuzzi, «Smart node networks orchestration: A new e2e approach for analysis and design for agile 4.0 implementation», *Sensors*, vol. 21, pp. 1–25, 2021.

[67] R. A. Atmoko, R. Riantini, and M. K. Hasin, «Iot real time data acquisition using mqtt protocol», in *Journal of Physics: Conference Series*, IOP Publishing, vol. 853, 2017, p. 012 003.

[68] D. Soni and A. Makwana, «A survey on mqtt: A protocol of internet of things (iot)», in *International conference on telecommunication, power analysis and computing techniques (ICTPACT-2017)*, vol. 20, 2017, pp. 173–177.

[69] A. Munshi, «Improved mqtt secure transmission flags in smart homes», *Sensors*, vol. 22, no. 6, p. 2174, 2022.

[70] C. Sun, K. Guo, Z. Xu, J. Ma, and D. Hu, «Design and development of modbus/mqtt gateway for industrial iot cloud applications using raspberry pi», in *2019 Chinese automation congress (CAC)*, IEEE, 2019, pp. 2267–2271.

[71] H. H. Alshammari, «The internet of things healthcare monitoring system based on mqtt protocol», *Alexandria Engineering Journal*, vol. 69, pp. 275–287, 2023.

[72] M. B. Yassein, M. Q. Shatnawi, S. Aljwarneh, and R. Al-Hatmi, «Internet of things: Survey and open issues of mqtt protocol», in *2017 international conference on engineering & MIS (ICEMIS)*, Ieee, 2017, pp. 1–6.

[73] F. Azzedin and T. Alhazmi, «Secure data distribution architecture in iot using mqtt», *Applied Sciences*, vol. 13, no. 4, p. 2515, 2023.

[74] A. Saabith, M. Fareez, and T. Vinothraj, «Python current trend applications-an overview», *International Journal of Advance Engineering and Research Development*, vol. 6, no. 10, 2019.

[75] S. Saabith, T. Vinothraj, and M. Fareez, «A review on python libraries and ides for data science», *Int. J. Res. Eng. Sci*, vol. 9, no. 11, pp. 36–53, 2021.

[76] S. Raschka, J. Patterson, and C. Nolet, «Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence», *Information*, vol. 11, no. 4, p. 193, 2020.

[77] A. S. Saabith, T. Vinothraj, and M. Fareez, «Popular python libraries and their application domains», *International Journal of Advance Engineering and Research Development*, vol. 7, no. 11, 2020.

[78] S. Tongkaw and A. Tongkaw, «A comparison of database performance of mariadb and mysql with oltp workload», *IEEE conference on open systems (ICOS). IEEE*, 2016.

[79] A. Bertoli, N. Battilani, and C. Fantuzzi, «Fault diagnosis and identification in agvs system», *IFAC-PapersOnLine*, vol. 58, pp. 246–251, 4 2024.

[80] G. T. Aguiar, G. A. Oliveira, K. Tan, N. Kazantsev, and D. Setti, «Sustainable implementation success factors of agvs in the brazilian industry supply chain management.», *Procedia Manufacturing*, vol. 39, pp. 1577–1586, 2019.

[81] J. Yin, B. Xu, and X. Hao, «Design and implementation of intelligent manufacturing system based on cloud service platform», *Mechatronics*, vol. 27, no. 06, pp. 35–42, 2021.

[82] T. Chen, Z. Qiu, and Z. Xu, «Design of agv fault diagnosis data acquisition system based on cloud platform [j]», *Academic Journal of Engineering and Technology Science*, vol. 5, no. 11, pp. 26–32, 2022.

[83] R. Yan, S. Dunnett, and L. Jackson, «Novel methodology for optimising the design, operation and maintenance of a multi-agv system», *Reliability Engineering & System Safety*, pp. 130–139, 2018.

[84] R. Yan, S. Dunnett, and L. Jackson, «Maintenance modelling of complex automated guided vehicle systems», *Annual Reliability and Maintainability Symposium (RAMS)*, pp. 1–6, 2019.

[85] I. Németh, J. Püspöki, A. B. Viharos, L. Zsóka, and B. Pirka, «Layout configuration, maintenance planning and simulation of agv based robotic assembly systems», *IFAC-PapersOnLine*, vol. 52, no. 13, pp. 1626–1631, 2019.

[86] B. Wang, D. Huo, Y. Kang, and J. Sun, «Agv status monitoring and fault diagnosis based on cnn», *Journal of Physics: Conference Series*, vol. 2281, no. 1, p. 012 019, 2022.

[87] R. Stetter, M. Witczak, and M. Pazera, «Virtual diagnostic sensors design for an automated guided vehicle», *Applied Sciences*, vol. 8, no. 5, p. 702, 2018.

[88] B. Mrugalska and R. Stetter, «Health-aware model-predictive control of a cooperative agv-based production system», *Sensors*, vol. 19, p. 532, 2019.

[89] M. Dares, K. W. Goh, Y. S. Koh, C. F. Yeong, E. Su, and P. H. Tan, «Development of agv as test bed for fault detection», *6th International Conference on Control, Automation and Robotics (ICCAR)*, pp. 379–383, 2020.

[90] M. Witczak, P. Majdzik, R. Stetter, and B. Lipiec, «Multiple agv fault-tolerant within an agile manufacturing warehouse», *IFAC-PapersOnLine*, vol. 52, no. 13, pp. 1914–1919, 2019.

[91] M. Witczak, P. Majdzik, R. Stetter, and B. Lipiec, «A fault-tolerant control strategy for multiple automated guided vehicles.», *Journal of Manufacturing Systems*, vol. 55, pp. 56–68, 2020.

[92] F. Ferraguti, V. Villani, and C. Storchi, «Mywelder: A collaborative system for intuitive robot-assisted welding», *Mechatronics*, vol. 89, p. 102 920, 2023.

[93] A. Bertoli, F. Ferraguti, and C. Fantuzzi, «An iot-enabled software architecture for user-friendly fault diagnosis and identification: The welding cobot use case», in *2024 7th Iberian Robotics Conference (ROBOT)*, IEEE, 2024, pp. 1–6.

[94] F. Sherwani, M. M. Asad, and B. S. K. K. Ibrahim, «Collaborative robots and industrial revolution 4.0 (ir 4.0)», *2020 International Conference on Emerging Trends in Smart Technologies (ICETST)*, pp. 1–5, 2020.

[95] S. El Zaatari, M. Marei, W. Li, and Z. Usman, «Cobot programming for collaborative industrial tasks: An overview», *Robotics and Autonomous Systems*, vol. 116, pp. 162–180, 2019.

[96] A. Cherubini, R. Passama, R. Crosnier, and P. Fraisse, «Collaborative manufacturing with physical human–robot interaction», *Robotics and Computer-Integrated Manufacturing*, pp. 1–13, 2016.

[97] T. Borgi, A. Hidri, B. Neef, and M. S. Naceur, «Data analytics for predictive maintenance of industrial robots», *International Conference on Advanced Systems and Electric Technologies (IC_ASET)*, pp. 412–417, 2017.

[98] A. Bonci, S. Longhi, G. Nabissi, and F. Verdini, «Predictive maintenance system using motor current signal analysis for industrial robot», *24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 1453–1456, 2019.

[99] C. Nentwich and G. Reinhart, «Towards data acquisition for predictive maintenance of industrial robots», *Procedia CIRP*, pp. 62–67, 2021.

[100] R. Pinto and T. Cerquitelli, «Robot fault detection and remaining life estimation for predictive maintenance», *Procedia Computer Science*, pp. 709–716, 2017.

[101] D. Kwon, M. R. Hodkiewicz, J. Fan, T. Shibutani, and M. G. Pecht, «Iot-based prognostics and systems health management for industrial applications», *IEEE Access*, vol. 4, pp. 3659–3670, 2016.

[102] A. Sohal, J. Olhager, P. O'Neill, and D. Prajogo, «Implementation of oee–issues and challenges», *Competitive and sustainable manufacturing products and services*, pp. 1–8, 2010.

[103] L. Sciavicco and B. Siciliano, *Modelling and control of robot manipulators*. Springer Science & Business Media., 2012.

Acknowledgments

Thank you all.