POLITECNICO DI TORINO

Master's Degree in ICTs for Smart Societies

Master's Degree Thesis

A COMPREHENSIVE APPROACH TO TEXTILE PLANT DIGITAL TWINS VIA CYBER-PHYSICAL SYSTEMS

Supervisor

Prof. Carla Fabiana CHIASSERINI

External Supervisors

Samuel David LÓPEZ

León Mauricio RIVERA

Candidate

Juan Pablo BENJUMEA MESA

Academic Year 2023-2024

Abstract

This work explores the practical implementation of Cyber-Physical Systems (CPS) within an industrial context, with a specific focus on enhancing data acquisition, automation, and monitoring processes to create a Textile Plant Digital Twin. The research is conducted in Medellín, Colombia, in collaboration with Leonisa, a prominent Colombian company specializing in underwear and women's clothing. The primary goal of this work is to improve the efficiency and reliability of industrial operations by applying CPS principles, a DevOps approach.

The study successfully develops an effective system for data acquisition, incorporating various methodologies such as Node-RED services, Python applications, RFID tag tracking, and water turbidity measurement. Furthermore, the adoption of DevOps principles, emphasizing Infrastructure as Code (IaC) with the use of Ansible, automates server and edge devices provisioning, configuration management, and periodic updates, reducing manual intervention and enhancing system reliability.

The importance of monitoring and visualization within the CPS framework is underscored by the seamless integration of Prometheus Node Exporter and Grafana. This integration facilitates extensive real-time system monitoring and proactive alerting mechanisms, particularly for the health status of numerous devices, like servers and edge computing devices, such as Raspberry Pis and other IoT devices. Additionally, the research delves into well-established CPS architecture models, including the 5C architecture, Reference Architecture Model Industry 4.0 (RAMI 4.0), and the Industrial Internet Reference Architecture (IIRA), providing guidelines for the integration of various technologies, such as digital twins, IIoT, and cloud computing, while addressing challenges in interoperability and security.

The research also categorizes industrial variables into four key groups: Process, Transactional, Reliability-Centered Maintenance (RCM), and Auxiliary variables, crucial for optimizing production, product quality, and operational efficiency. The implications of this work, measured by LoA, extend beyond the textile industry, offering valuable insights for CPS implementation in various industrial domains.

This project lays the foundation for the integration of CPSs, digital twins,

and DevOps practices in the industrial landscape, bringing new levels of efficiency, productivity, and sustainability in the company. The continuous assessment on Levels of Automation (LoA), employing the Dynamo methodology, reveals significant enhancements across multiple facets of industrial operations. The integration of Node-RED applications, Python developments, RFID tracking, and Turbidity Measurement demonstrates a noteworthy increase in the LoA, signifying improved automation and efficiency in data acquisition processes. The detailed assessment across various stages, including system configuration, data acquisition, data transformation, data transmission, error handling, maintenance and updates, and monitoring and reporting, highlights the specific areas of advancement within each technology implementation.

The average LoA for Node-RED applications has increased from 3 to 5, showcasing substantial improvements in data transformation, error handling, and data transmission. Similarly, Python applications exhibit a notable rise in average LoA from 2 to 5, emphasizing advancements in data transformation, data transmission, and error handling. RFID tracking and Turbidity Measurement applications show increases in average LoA from 3 to 5 and from 1 to 5, respectively, underlining their growing importance in supporting comprehensive data acquisition systems.

Future work should focus on adapting CPS principles to diverse industrial contexts, refining data acquisition techniques, and exploring advanced analytics for predictive maintenance, cybersecurity and process optimization. Also, the potential of a microservice architecture and Kubernetes in further enhancing system scalability, resilience, and deployment efficiency in order to leverage the use of multiple resources from servers and edge computing devices like Raspberry Pis.

Table of Contents

List of Tables

List of Figures

List of Acronyms

AAS
: Asset Administration Shell

AI
: Artificial Intelligence

API
: Application Programming Interface

CPS
: Cyber-Physical System

HA
: High Availability

IaC
: Infrastructure as Code

IIRA
: Industrial Internet Reference Architecture

IIoT
: Industrial Internet of Things

IoT
: Internet of Things

LoA
: Levels of Automation

OPC UA

Open Platform Communications Unified Architecture

OS

Operative System

PLC

Programmable Logic Controller

RAMI 4.0

Reference Architecture Model Industry 4.0

RCM

Reliability-Centered Maintenance

RF

Radio Frequency

RFID

Radio Frequency Identification

SCADA

Supervisory Control and Data Acquisition

Chapter 1

Introduction

In recent years, the concepts of digital twins and Cyber-Physical Systems (CPSs) have gained significant attention in various industries, including manufacturing, energy, healthcare, and smart cities. A digital twin can be described as the virtual representation of a physical system or process, seamlessly integrated with real-time data from sensors and connected devices [1]. It allows for the bi-directional exchange of information between the physical and virtual worlds, enabling real-time monitoring, analysis, and optimization of the physical system [2].

The implementation of digital twins has been made possible by advancements in technologies such as artificial intelligence (AI), the internet of things (IoT), big data analytics, and cybernetics [2]. These technologies provide the necessary infrastructure and tools for creating and managing digital twins, facilitating the integration of data from multiple sources, and enabling advanced analysis and decision-making.

To appreciate the transformative power of digital twins and CPSs in today's manufacturing industry, it is essential to consider the historical context of industrial revolutions. The journey of manufacturing through several distinct phases underscores the profound impact that technological advancements have had on production processes and systems. These industrial revolutions serve as milestones in the evolution of manufacturing:

The First Industrial Revolution The rise of mechanization and steam power in the late 18th century marked the beginning of the first industrial revolution. It transformed manual labor-intensive industries into mechanized ones, paving the way for increased production and economic growth [3].

The Second Industrial Revolution In the late 19th and early 20th centuries, the second industrial revolution introduced electricity, mass production, and

the assembly line. These innovations drastically improved manufacturing efficiency and output [4].

The Third Industrial Revolution The latter half of the 20th century brought about the third industrial revolution, characterized by the rise of computers and automation. Manufacturing systems became more sophisticated, with computers controlling various processes [4].

The Fourth Industrial Revolution (Industry 4.0) Today, we find ourselves on the cusp of the fourth industrial revolution, often referred to as Industry 4.0. This phase is defined by the convergence of physical systems with digital technologies, creating smart factories where CPSs, IoT devices, and digital twins play pivotal roles. Industry 4.0 promises unprecedented levels of automation, data-driven decision-making, and connectivity across manufacturing processes [3].

Digital twins have the potential to revolutionize various aspects of industries, from product design and development to operation and maintenance. They enable better understanding, prediction, and control of systems, leading to improved efficiency, productivity, and sustainability [5]. The application of digital twins in different domains has already shown promising results.

In the textile industry, for example, the implementation of a digital twin can bring significant benefits. By capturing and analyzing data from the production process, including machine performance, material quality, and environmental conditions, a digital twin can optimize production schedules, minimize waste, and reduce energy consumption [6]. It can also enable real-time monitoring of equipment health, enabling predictive maintenance and preventing unplanned downtime.

Furthermore, the use of open-source IoT technologies enhances the accessibility, flexibility, and scalability of digital twin implementations. Open-source platforms provide a collaborative environment for the development and deployment of digital twin solutions, allowing for customization and integration with existing systems [1]. This approach enables the cost-effective and efficient implementation of digital twins in various industries, including textile manufacturing.

However, the implementation of digital twin technology in the textile industry and other sectors has its difficulties and challenges. These challenges include data integration, security, interoperability, and privacy concerns [7]. Additionally, the design and development of digital twins require expertise in modeling, simulation, data analytics, and domain knowledge [8]. Therefore, thorough research and understanding of the principles, applications, and enablers of digital twins are crucial for successful implementation.

Leonisa is a Colombian underwear and women's clothing company with a broad presence in the global market. Founded in 1956, Leonisa has managed to position itself as a leading brand in the underwear industry thanks to its commitment to quality, design, and innovation.

This thesis aims to explore the implementation of a CPSs for textile processes in Leonisa using IoT technologies. It will investigate the challenges, opportunities, and benefits of utilizing digital twin and CPS technology in the textile industry. Additionally, it will explore the underlying technologies, including IoT, Infrastructure as Code (IaC) and data monitoring software, that play pivotal roles in the creation and functionality of digital twins. Furthermore, specific examples and case studies from related industries will be analyzed to provide insights into the potential applications and best practices for implementing CPSs.

This research will add to the existing body of knowledge on digital twins and their implementation in the textile industry at Leonisa through CPS structuring and understanding. It will focus on the vertical integration of the system in the hopes of contributing to the company's future growth and success. The thesis will provide practical recommendations and guidelines for further implementation of this system in the company.

Figure 1.1 shows the research focus of this work, where the pivotal aspects of Integration and Coordination, with a specific emphasis on their application in Industrial Automation are applied. This research will delve into the complex process of seamlessly combining heterogeneous components and orchestrating their operations effectively in industrial settings. Integration will involve bridging the gap between different hardware and software elements, ensuring they communicate and cooperate harmoniously. Coordination, on the other hand, will encompass real-time synchronization, control algorithms, and resource allocation strategies to optimize the functioning of CPS in industrial automation scenarios. By focusing on these critical components in the context of industrial automation, this thesis aims to contribute insights and solutions to enhance the efficiency, reliability, and adaptability of CPS in real-world manufacturing and production environments.

Central to this exploration is the concept of Levels of Automation (LoA), a nuanced approach to human-machine collaboration. Traditionally, automation decisions have been perceived in a binary manner within industrial contexts, either favoring complete autonomy or maintaining a reliance on human intervention. As we start off on a digital transformation through the implementation of CPS, it becomes imperative to reassess this dichotomous perspective [9]. While promising increased productivity and reduced labor costs, industrial automation often encounters challenges requiring human expertise during disturbances and system failures. Recognizing the need for a more adaptable and strategic approach, this thesis

Figure 1.1: Cyber-Physical Systems Research Areas.

proposes the use of the concept of LoA as an indicator of the impact of the automation improvements.

LoA signifies a continuum in the collaboration between humans and machines, ranging from total manual work to complete automation. The approach suggests that the interaction and task allocation should be considered as a dynamic factor rather than an all-or-nothing decision [10]. This introduction of LoA is particularly relevant to the textile manufacturing context, where the delicate balance between mechanized processes (mechanical LoA) and cognitive activities (information LoA) plays a crucial role in system effectiveness.

Table 1.1 illustrates the levels of automation [11], providing a spectrum from

totally manual work to fully automatic processes in both mechanical and information domains. This framework serves as a guide for evaluating and categorizing the degree of automation within textile manufacturing processes, laying the foundation for a more informed and context-aware implementation of CPS and digital twin technologies.

Levels	**Mechanical**	**Information**
1	Totally manual	Totally manual
2	Static hand tool	Decision giving
3	Flexible hand tool	Teaching
4	Automated hand tool	Questioning
5	Static workstation	Supervising
6	Flexible workstation	Intervene
7	Totally automatic	Totally automatic

Table 1.1: Levels of Automation for mechanical and information applications [11]

Chapter 2

The Transformation of the Manufacturing Industry through Cyber-Physical Systems

The manufacturing industry has undergone significant transformations throughout history, and the advent of Cyber-Physical Systems (CPSs) has brought about another wave of change. CPS refer to the integration of physical and digital systems, enabled by technologies such as the IoT, AI, advanced robotics, and additive manufacturing [12]. This integration allows for real-time monitoring, decision-making, and control of physical processes, leading to increased efficiency, flexibility, and productivity in the manufacturing industry [13]. By using cutting-edge information analytics, networked machines will be able to operate more efficiently, collaboratively, and resiliently. This trend is transforming the manufacturing industry into the next generation, known as Industry 4.0 [14].

2.1 Characteristics of Smart Manufacturing

Smart manufacturing is characterized by the integration of physical and digital systems, real-time data-driven operations control, and the use of advanced technologies for monitoring, simulation, and prediction of manufacturing operations [13, 15]. It involves the deployment of CPSs, which monitor and control physical processes, create a virtual copy of the physical world, and make decentralized decisions [16]. Additionally, smart manufacturing emphasizes connectivity, interoperability, and

intelligence, enabling seamless communication and collaboration between machines, humans, and data [17]. The goal of smart manufacturing is to achieve increased efficiency, flexibility, and agility in the production process, leading to improved productivity and competitiveness [18].

2.2 Technologies Driving Smart Manufacturing

Several key technologies are driving the transformation of the manufacturing industry through CPS. One such technology is the IoT, which allows for the connection and communication between physical devices and the digital world [12]. The IoT enables real-time monitoring, data collection, and analysis of manufacturing processes, facilitating better decision-making and optimization [19]. Advanced robotics is another crucial technology in smart manufacturing, enabling the automation of repetitive and complex tasks, improving precision, reducing errors, and increasing productivity [12].

3D printing, also known as additive manufacturing, is transforming the manufacturing industry by making it possible to create complex and customized products [12]. This technology provides flexibility, reduces waste, and lowers costs by eliminating the need for traditional manufacturing processes such as molding and machining [18]. AI plays a crucial role in smart manufacturing, enabling machines to learn, reason, and make predictions based on large volumes of data [19]. AI enables advanced analytics, machine vision, natural language processing, and predictive maintenance, leading to improved efficiency and quality in manufacturing processes [20].

2.3 Enabling Factors for Smart Manufacturing

Several factors enable the development and implementation of smart manufacturing in the industry. One such factor is the concept of Industry 4.0, which refers to the integration of CPSs, IoT, and advanced technologies, reshaping traditional manufacturing processes. Industry 4.0 highlights the need for real-time connection and communication between physical and digital systems, enabling the exchange of information and decentralized decision-making [12].

Edge computing is also considered an important enabler for smart manufacturing, providing the necessary support for data processing, communication, and control in this context. It plays a crucial role in realizing the vision of smart cities [21]. By utilizing edge computing, fog computing, and cloud computing, a hierarchy reference architecture can be established for smart manufacturing,

enabling a smart manufacturing service system [22]. Edge computing is also seen as one of the key technologies in the field of machine learning and Industry 4.0 applications in manufacturing [23]. It facilitates the integration of manufacturing and information communication technologies, including computing, communication, and control, which is essential for enabling higher value-added manufacturing [24]. Moreover, edge computing, along with other technologies like blockchain and AI, can significantly support the development of smart manufacturing [19].

Edge computing complements the capabilities of IoT platforms by bringing computation and data storage closer to edge devices. This enables real-time data processing and analysis, reducing latency and bandwidth requirements. Edge computing is particularly beneficial in manufacturing environments where real-time decision-making is critical. By processing data at the edge, manufacturers can respond quickly to changing conditions and optimize their operations in real time [25].

Also, with the softwarization of the networking processes and services using network function virtualization (NFV), it is possible to dynamically and opportunistically integrate the whole set of diverse resources which are available locally at the edge, without compromising QoS provisioning to the users in a vertical industry [26].

Similarly, digital twins are essential enablers of smart manufacturing. As a virtual representation of a physical object or system, they provide real-time insights, predictions, and simulations [19]. Digital twins bridge the gap between the physical and digital worlds, allowing for better monitoring, analysis, and optimization of manufacturing processes [19]. Additionally, the availability of big data capabilities and cloud services plays a crucial role in smart manufacturing. The integration and analysis of vast amounts of data from sensors, machines, and systems enable real-time decision support, predictive analytics, and optimization of manufacturing processes [18].

In the textile industry specifically, digital twins can play a crucial role in predicting and controlling the self-folding behavior of weft-knit fabrics. Traditional computer-aided design software cannot anticipate and predict this behavior accurately. However, by characterizing the mechanical forces driving these behaviors and analyzing fabric deformations, digital twins can be used to purposely control the self-folding behavior of fabrics [27].

2.4 Challenges and Opportunities

Although smart manufacturing through CPSs has many advantages, such as increased efficiency, productivity, and quality, some challenges need to be addressed.

One such challenge is the integration and interoperability of diverse technologies, systems, and processes [15]. The seamless communication and collaboration between different devices, machines, and systems require standardized protocols and interfaces to ensure compatibility and interoperability [15, 28].

Another challenge is the security and privacy of data in smart manufacturing systems. As manufacturing operations become increasingly connected and data-intensive, ensuring the confidentiality, integrity, and availability of data becomes crucial. Robust cybersecurity measures and protocols need to be implemented to protect against cyber threats and ensure the privacy and security of sensitive information [29].

Despite these challenges, smart manufacturing offers significant opportunities for the industry. It enables increased efficiency, flexibility, and agility in manufacturing processes, leading to reduced costs, improved productivity, and enhanced competitiveness [13, 18]. Smart manufacturing also opens up new possibilities for customization and personalization, as well as the development of smart products and services [18]. The integration of advanced technologies, data-driven decision-making, and automation can lead to new business models and revenue streams for manufacturers [29].

Chapter 3

CPS Architecture Models Review

In order to implement CPS architecture models, various reference models and architectures have been proposed. The 5C architecture, the Reference Architecture Model Industry 4.0 (RAMI 4.0) and the Industrial Internet Reference Architecture (IIRA) are three major reference architectures that have been developed by industry-driven initiatives. These reference architectures provide guidelines and standards for the design and implementation of Industry 4.0 systems [30]. They define the structure and methodology of CPSs and provide a framework for integrating various technologies, such as digital twins, IIoT, and cloud computing [31].

Despite the advancements in Industry 4.0 architectures, there are still challenges and research issues that need to be addressed. These include the development of efficient methodologies for creating digital twins, ensuring interoperability and integration of different systems and technologies, and addressing security and privacy concerns [32, 33]. Additionally, the implementation of Industry 4.0 architectures requires organizational and managerial changes, as well as the development of new skills and competencies [34].

3.1 5C Architecture overview

The 5C Architecture model is a framework proposed for Cyber-Physical Systems (CPS) in the context of Industry 4.0 manufacturing systems [35]. It was first introduced by [19] and consists of five levels: connection, conversion, cyber, cognition, and configuration [36]. This architecture aims to support plug and play smart sensor connectivity, enable cyber-physical smart interactions, and provide a layered and interconnected structure for CPS [19] as shown in Figure 3.1.

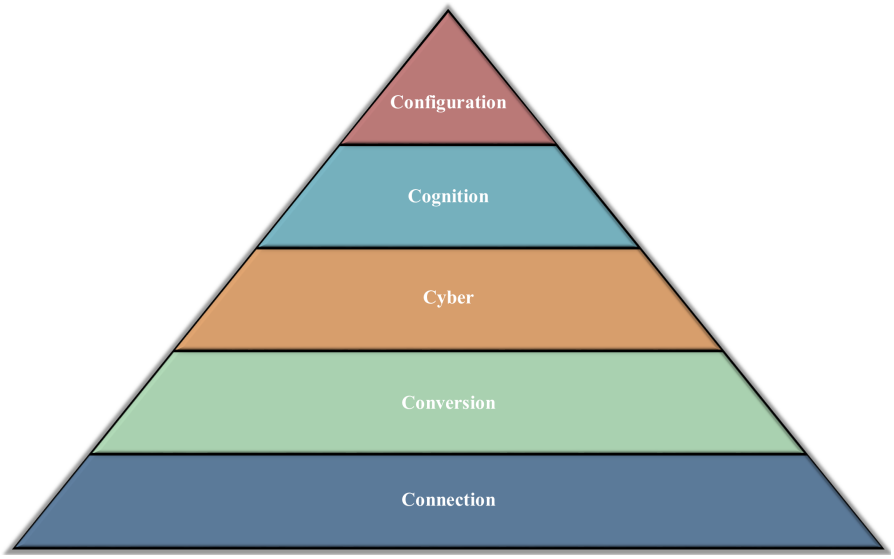

Figure 3.1: 5C Architecture of CPS.

Connection level focuses on the physical connectivity of sensors and actuators in the CPS [36]. It involves the integration of physical processes with communication networks and computing infrastructures [37].

Conversion level is responsible for converting raw data from sensors into meaningful information [36]. This level involves data-to-information conversion and processing [38].

Cyber level deals with the communication and networking aspects of the CPS [36]. It includes the integration of software components in the cyberspace and IoT devices in the physical environment [39].

Cognition level involves the analysis and interpretation of data to make intelligent decisions [36]. This level can incorporate artificial intelligence and machine learning techniques to enable cognitive capabilities in the CPS [40].

Configuration level focuses on the configuration and management of the CPS [36]. It includes the design and configuration of the CPS components and their interactions [41].

The 5C Architecture model provides a framework for the integration of these five levels to build a comprehensive CPS [42]. This architecture model has been widely discussed and applied in various domains. For example, it has been used in the context of smart manufacturing [19], smart workplaces [38], ambient assisted living [40], precision agriculture [43], smart grids [44], and intelligent transportation

systems [45]. It has also been used as a reference framework for digital twins in the context of CPS [46].

3.2 Reference Architecture Model Industry 4.0 (RAMI 4.0) overview

RAMI 4.0 is represented by a three-dimensional map as shown in Figure 3.2, which is composed of three axes: Hierarchy Levels, Product Life-cycle, and Architecture Layers. The Hierarchy Levels axis represents the different levels of automation and control within the system, ranging from the field level to the management level. The Product Life-cycle axis focuses on the different stages of a product's life-cycle, including design, production, operation, and maintenance. The Architecture Layers axis represents the different layers of the system architecture, including the physical layer, communication layer, information layer, and application layer [47].

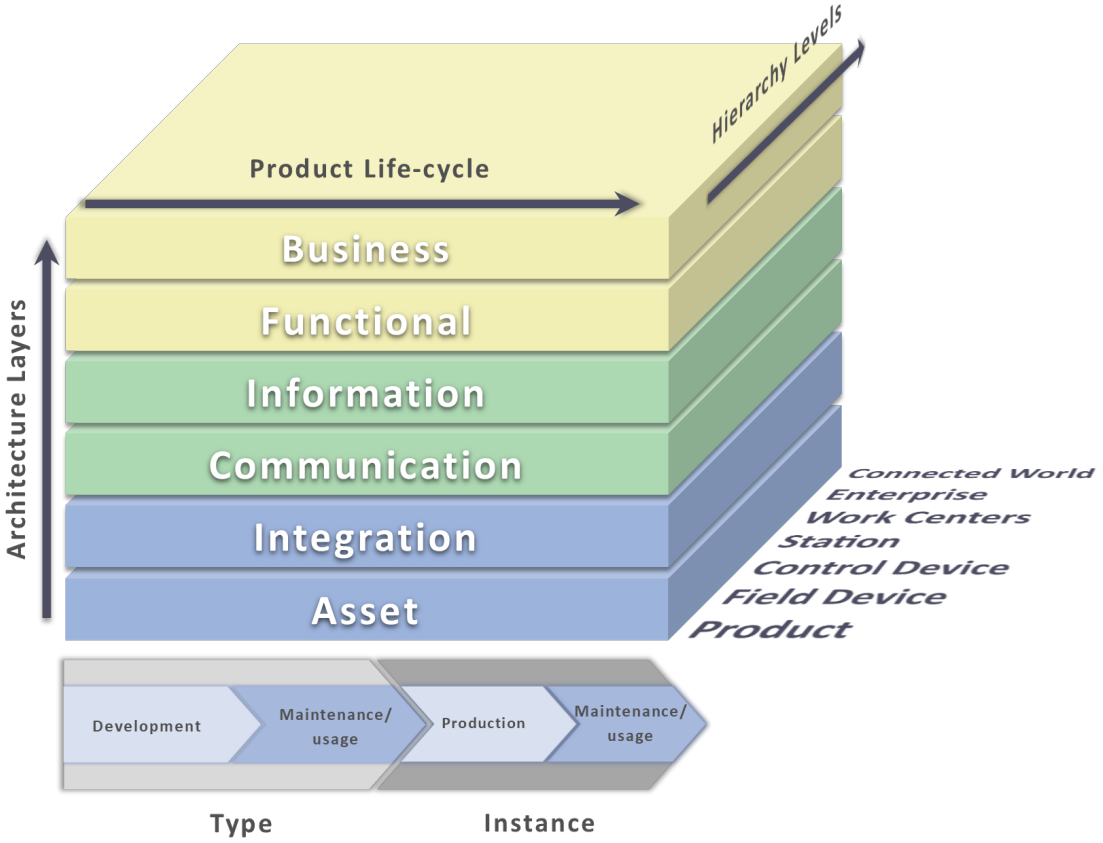

Figure 3.2: RAMI 4.0 three-dimensional map.

One of the key aspects of RAMI 4.0 is the concept of the Asset Administration Shell (AAS). The AAS represents the digital representation of a physical asset or entity within the system. It contains all the relevant information and data about the asset, including its characteristics, behavior, and interfaces. The AAS enables interoperability and communication between different assets and systems within the Industry 4.0 environment [48].

RAMI 4.0 also emphasizes the importance of standardization, particularly in the Communication Layer. It recommends the use of Open Platform Communications Unified Architecture (OPC UA) as the standard for communication between different components of the system. OPC UA provides a secure and reliable communication framework that enables interoperability and integration of different devices and systems [49].

3.3 Industrial Internet Reference Architecture (IIRA) overview

The IIRA is being standardized by the Object Management Group (OMG) and provides a framework for designing and implementing industrial internet systems [30]. It aims to ensure interoperability, scalability, and security in the context of the Industrial Internet of Things (IIoT) [50].

Numerous publications in fields such as IIoT, Cyber-Physical Production Systems, Enterprise Architectures, Enterprise Integration, and Cloud Manufacturing have explored and discussed the IIRA framework. However, it is worth noting that the adoption of reference architectures for Industry 4.0 has been limited, and there is a lack of awareness and discussion about the compatibility of proposed architectures with internationally standardized reference architectures [30].

The IIRA aims to address the challenges of interoperability and integration in the context of Industry 4.0. It provides a bottom-up view of an industrial process, from the physical transducers at the physical layer to the business layer. Emphasizing data exchange, communication protocols, and standardization, it promotes the seamless integration of devices and systems in industrial settings [51].

As shown in Figure 3.3 the IIRA encompasses four key Viewpoints, each addressing distinct aspects of IIoT implementations. The Business Viewpoint centers on the business aspects, including business models, value propositions, and revenue streams, ensuring that IIoT solutions align with broader business strategies. The Usage Viewpoint focuses on practical usage scenarios, encompassing use cases, user stories, and user requirements, ensuring user-centric design and functionality. The Functional Viewpoint delves into the technical details, specifying functional

components, interfaces, and protocols, while the Implementation Viewpoint addresses the practical deployment aspects, including hardware, software, and network infrastructure [52].

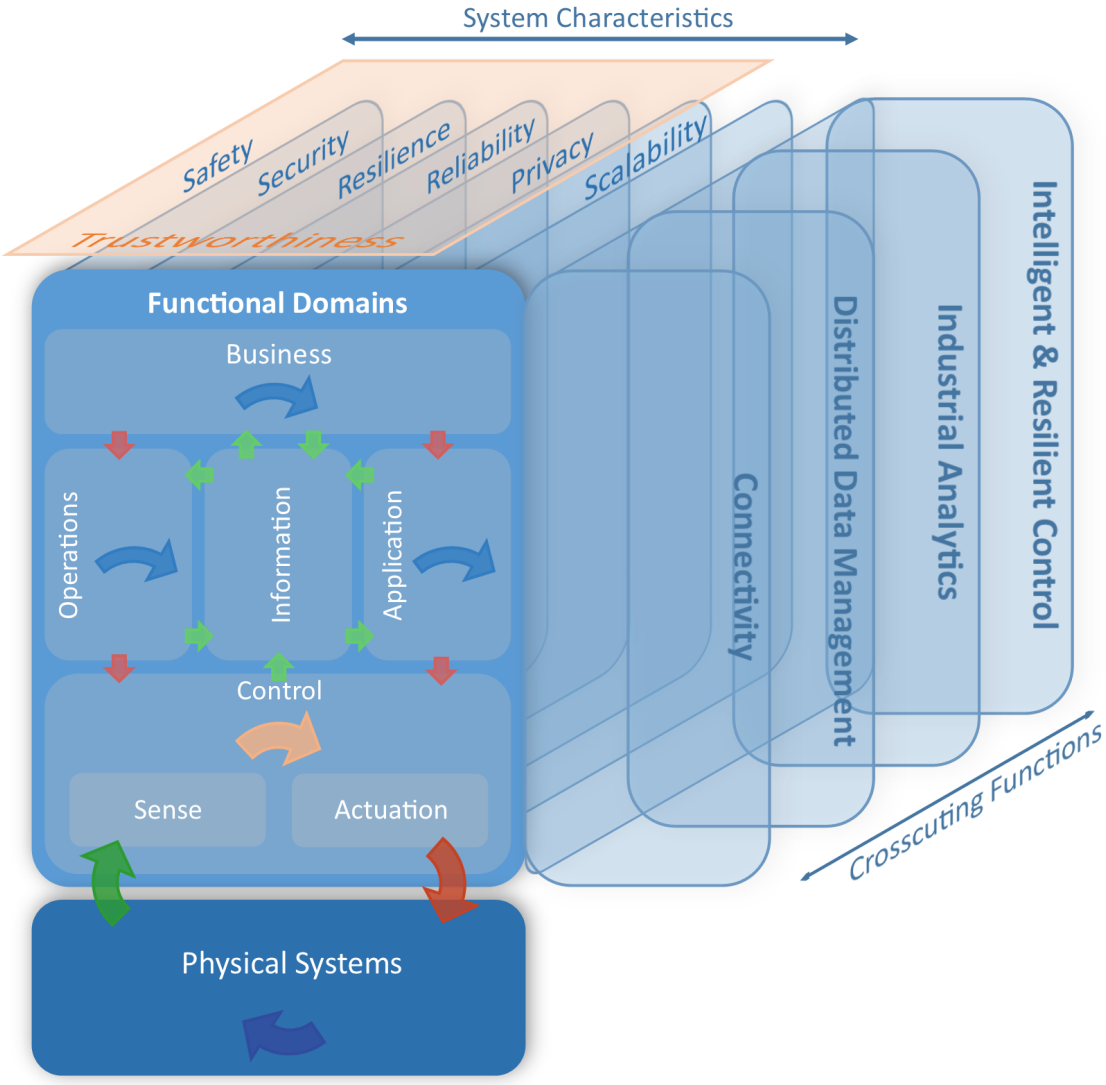

Figure 3.3: IIRA functional domains, crosscutting functions and system characteristics.

Primarily designed to support the implementation of CPSs in smart factories, it provides a blueprint for constructing the CPS, encompassing components like the IIoT gateway and the smart factory data center. It also emphasizes the importance of both vertical and horizontal integration in the design of CPSs for smart factories [36].

The IIRA is closely related to other concepts and technologies in the context of Industry 4.0. For example, it is often mentioned in the context of OPC UA [30]. It also intersects with other standards and technologies such as the IEEE 1451 and IEC 61499 standards, which enable information exchange and control in industrial processes [51].

3.4 Correlation among 5C Architecture, RAMI 4.0 and IIRA

It is essential to consider the correlation between the 5C, RAMI 4.0, and IIRA Industry 4.0 architectures. This integration of different systems and technologies allows greater efficiency, productivity, and innovation in the industrial sector toward the realization of Industry 4.0's objectives.

The Functional Mapping among these architectures shown in Figure 3.4 reveals significant similarities in terms of their domains, levels, and layers. To elaborate, the domains defined in the IIRA architecture exhibit analogous functions with the levels specified in the 5C Architecture and the layers established in the RAMI 4.0 framework. This convergence implies that these architectures are not mutually exclusive but can be effectively integrated and employed together to advance the objectives of Industry 4.0 [52].

Notably, RAMI 4.0 and IIRA collect significant attention in the industrial community due to their strong focus on Industry 4.0 proposals. They foster the development of applications, services, and business concepts to facilitate integration among industries and the entire manufacturing sector. To ensure interoperability, concepts such as standardized functions, semantics, and unique identifiers for properties and assets are crucial. This entails the standardization of identification, networking, semantics, and functional mapping, all of which are fundamental for seamless interoperability between IIoT and Industry 4.0 systems. For example, for services like operation and maintenance under IIoT systems (IIRA), the technical data must align with materials, components, and the entire manufacturing process, which are accessible from manufacturers (RAMI 4.0). Standardized parameters are essential for recognizing the same product and its associated data in both RAMI 4.0 and IIRA architectures, facilitating effective interoperability between the two systems [52].

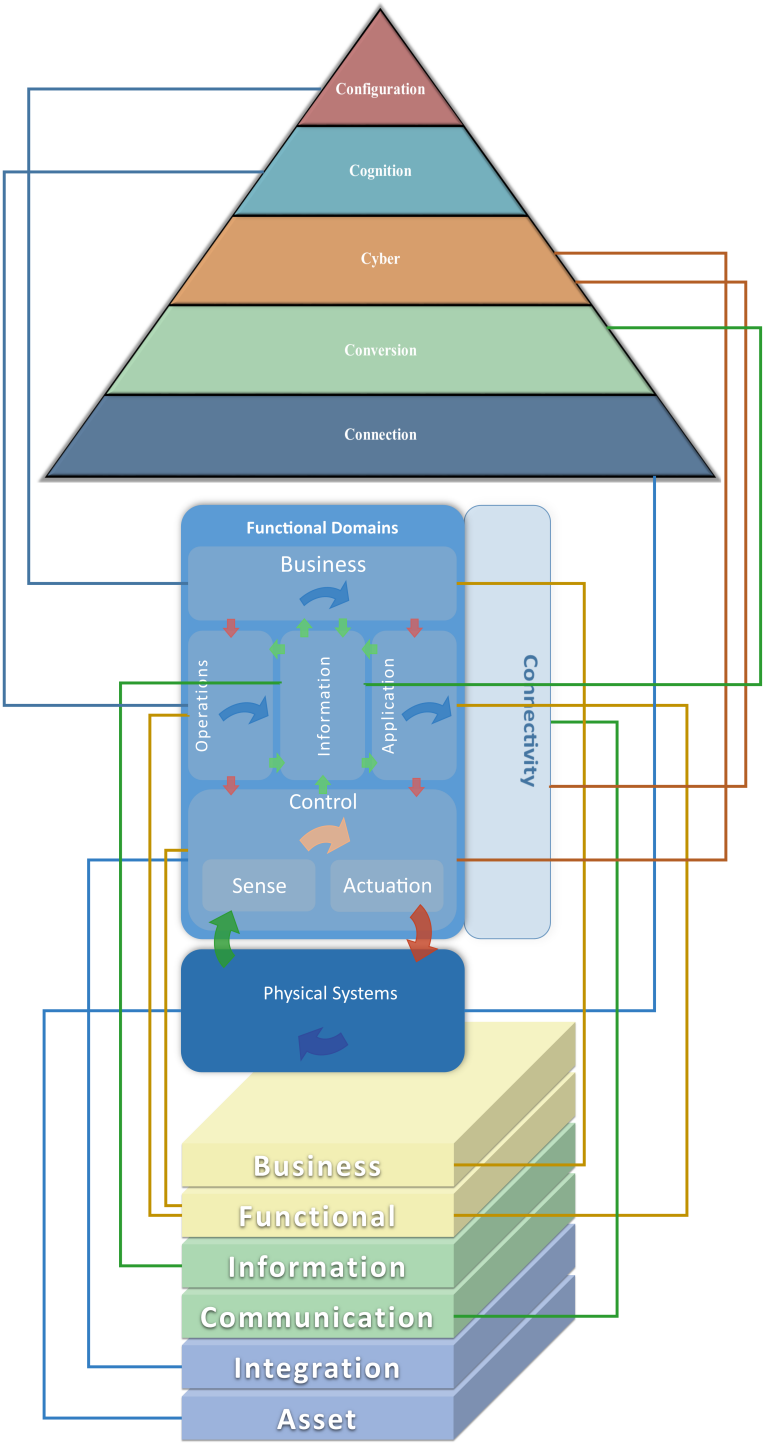

Figure 3.4: Functional mapping among 5C Architecture, IIRA and RAMI 4.0.

Chapter 4

System Implementation

4.1 Categorization of Industrial Variables

To comprehensively understand the plants' operations and subsequently comprehend it in the CPS, we categorize industrial variables into four key categories, Process, Transactional, Reliability-Centered Maintenance (RCM) and Auxiliary variables.

Process Variables form the bedrock of data acquisition in a manufacturing plant. They encompass a wide range of parameters directly linked to the production processes (product transformation). Understanding and monitoring these variables are crucial to optimizing production, ensuring product quality, and maintaining operational efficiency. The most important process variables include temperature, pressure, humidity, speed, material flow rates, etc.

Transactional Variables encompass data related to the plant's operational transactions, including order processing times, inventory management, and logistics data. These variables provide insights into daily operations and resource allocation.

Reliability-Centered Maintenance (RCM) Variables focus on equipments reliability and include data on machine wear and tear, failure rates, and maintenance history. These variables support predictive maintenance strategies to minimize downtime.

Auxiliary Variables encompass essential resources' consumption, such as water, gas, and electricity. Efficient management of these resources is crucial for sustainability and cost control.

In this way, to effectively manage and optimize manufacturing processes, the correct understanding of key industrial variables is crucial. Within the textile

manufacturing domain, these variables play a central role in ensuring the efficiency and quality of operations. Thereby, the critical variables are explored as specific to each of the most important stages of the textile production process in the company, weaving, finishing, cutting, and sewing. By categorizing these variables, we lay the basis for precise monitoring and management, ultimately enhancing production rates, product quality, and overall operational efficiency, and in that context, specific variables that are important could be mentioned.

Weaving Process Variables

- Machine Speed (Process Variable): Monitoring the speed of weaving machines is crucial for optimizing production rates and ensuring consistent fabric quality.
- Tension and Stress (Process Variable): Maintaining proper tension in the weaving process is vital to prevent fabric defects and ensure material flow rates are consistent.
- Material Flow Rates (Process Variable): Measuring the rate at which raw materials, such as different types of threads (cotton, elastane, polymers, etc.), are fed into the weaving machines is essential for quality control and production optimization.
- Temperature and Humidity (Process Variable): Monitoring environmental conditions can be crucial, especially for natural fibers like cotton. Temperature and humidity control can impact the quality of the woven fabric.
- Roll Identification (Transactional Variable): Assigning unique identifiers to woven rolls allows for easy tracking and tracing of materials throughout the manufacturing process.
- Loom Setting (Process Variable): The configuration and settings of the weaving machines, including parameters like harness and shuttle adjustments, directly influence fabric quality and production efficiency.
- Yarn Tension Variation (Process Variable): Monitoring variations in yarn tension within the weaving process to ensure even fabric construction.
- Weft Thread Density (Process Variable): Measuring the number of weft threads per inch to control fabric weight and structure.
- Loom Downtime (Process Variable): Tracking the time during which looms are not in operation due to maintenance or issues, which affects production efficiency.

- Fabric Width (Process Variable): Measuring the width of the woven fabric to ensure it conforms to specifications.
- Quality Inspection Points (Process Variable): Identifying key checkpoints for quality inspections during the weaving process to detect defects.
- Maintenance Schedule Compliance (RCM Variable): Ensuring machines maintenance schedules are adhered to for preventing breakdowns and downtime.

Finishing Process Variables

- Temperature and Humidity (Process Variable): Precise control of temperature and humidity is essential for processes like thermofixation, ensuring that fabric finishing is consistent and meets quality standards.
- Scouring Process Monitoring (Process Variable): Monitoring the scouring process is crucial to achieve the desired fabric properties during finishing.
- Dyeing Parameters (Process Variable): Measuring and controlling the dyeing process, including factors like dye concentration, temperature, and time, is vital for achieving the desired color and appearance.
- Tension and Stress (Process Variable): Maintaining proper tension during finishing processes ensures fabric quality and reduces defects.
- Roll Tracking (Transactional Variable): Continuing to track the fabric rolls to maintain traceability and monitor the quality of finished products.
- Drying Time (Process Variable): Monitoring the time taken for drying processes, which impacts overall production time.
- Chemical Concentrations (Process Variable): Measuring the concentration of chemicals used in processes like dyeing and finishing to control color consistency and fabric properties.
- pH Levels (Process Variable): Monitoring pH levels during chemical treatments, as variations can affect fabric properties.
- Energy Consumption (Auxiliary Variable): Measuring the energy consumed during finishing processes for cost control and environmental sustainability.
- Roller Pressure (Process Variable): Controlling the pressure applied by rollers during processes like calendering, which impacts fabric smoothness.
- Waste Generation (Auxiliary Variable): Tracking the generation of waste materials during finishing processes for sustainability measures.
- Maintenance Schedule Compliance (RCM Variable)

Cutting Process Variables

- Material Utilization (Process Variable): Measuring the efficiency of material usage during the cutting process helps minimize waste and optimize material consumption.
- Cutting Speed and Accuracy (Process Variable): Monitoring the speed and precision of cutting machines is essential to ensure that fabric pieces are accurately cut to the required dimensions.
- Inventory Management (Transactional Variable): Efficient management of cut fabric pieces in inventory, including tracking stock levels, reduces carrying costs and helps with resource allocation.
- Pattern Layout Efficiency (Process Variable): Assessing the efficiency of pattern layouts to maximize fabric utilization and minimize waste.
- Cutting Blade Sharpness (Process Variable): Monitoring the sharpness of cutting blades, as dull blades can lead to uneven cuts and fabric defects.
- Cutting Table Surface Quality (Process Variable): Ensuring the quality and condition of the cutting table surface, which affects the precision of cutting.
- Pattern Alignment (Process Variable): Verifying the alignment of fabric patterns with cutting templates to avoid inaccuracies.
- Maintenance Schedule Compliance (RCM Variable)

Sewing Process Variables

- Sewing Machine Efficiency (Process Variable): Measuring the efficiency of sewing machines in terms of speed, downtime, and quality of stitching is crucial for optimizing production.
- Quality Control (Process Variable): Implementing quality control measures, such as checking stitch consistency and product dimensions, ensures the final product meets quality standards.
- Order Processing Times (Transactional Variable): Tracking the time it takes to process and complete orders is important for operational efficiency and customer satisfaction.
- Logistics Data (Transactional Variable): Monitoring logistics data, such as shipping times and transportation costs, ensures timely deliveries and cost-effective transportation.

- Thread Tension (Process Variable): Monitoring the tension of sewing machine threads to prevent thread breakage and ensure stitch quality.
- Operator Skill Level (Process Variable): Evaluating the skill level of machine operators, as it influences stitching quality and production speed.
- Maintenance Schedule Compliance (RCM Variable)
- Production Output Variability (Process Variable): Measuring variations in daily production outputs to optimize resource allocation.
- Operator Workload (Process Variable): Monitoring the workload of sewing machine operators to ensure a balanced and efficient work environment.
- Defects per Unit (Process Variable): Calculating the number of defects per unit produced to maintain quality standards.

These variables among many more, when measured and monitored effectively, enable the company to optimize its processes, reduce defects, minimize waste, and enhance overall operational efficiency, leading to the production of high-quality textile products.

4.2 Data Acquisition and Preparation

Data acquisition and preparation, serves as the cornerstone of implementing a robust CPS for an industrial plant. Data acquisition involves the systematic collection of relevant data from the industrial variables, while data preparation ensures the quality, consistency, and readiness of this data for its consumption.

4.2.1 System Structure

Before the implementation of our enhanced system, the company had an existing structure, as depicted in Figure 4.1, for data acquisition from machines. However, this previous system had limitations, notably in terms of connectivity, as it did not have the capability to access the data from PLCs and certain machines, leaving these critical data sources unmonitored and unmanaged. On the other hand, as shown in Figure 4.2, the system architecture deployed by the company today for data acquisition and management embodies a framework which is evolving based on the good practices proposed in this work. This structure is designed to capture, store, and visualize industrial data, mostly driven by the following core components:

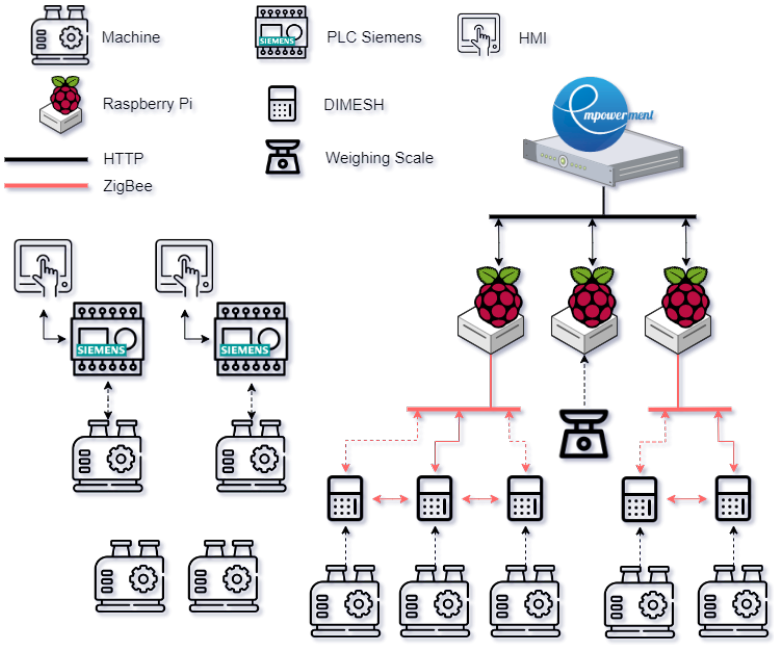

Figure 4.1: Old system structure.

Empowerment Application: At the heart of the system resides the Empowerment application, hosted on a dedicated server. This in-house made application functions as the main actor for data storage and visualization of the industrial processes. It primarily accumulates and manages data belonging to Process and Transactional variables originating from diverse machinery and industrial processes. Raspberry Pi devices play a the main role in transmitting this data to the Empowerment database. Empowerment is instrumental in data analysis and visualization, allowing the company to deploy insights derived from the acquired data.

Cybernetics Server: A notable enhancement to the system structure is the introduction of the Cybernetics server (Ubuntu). This server assumes a central role in data acquisition and management for IIOT applications. It directly interfaces with Programmable Logic Controllers (PLCs) using Node-RED through HTTP, facilitating the real-time extraction of data concerning Process and Transactional variables from machines under PLC control, among other sources of information like in site PCs and Raspberry Pis through SSH connections. This direct connections strengthen the system's ability to promptly access critical data.

Raspberry Pi devices: Integral components within the system's architecture are the Raspberry Pi devices, which function as edge computing devices.

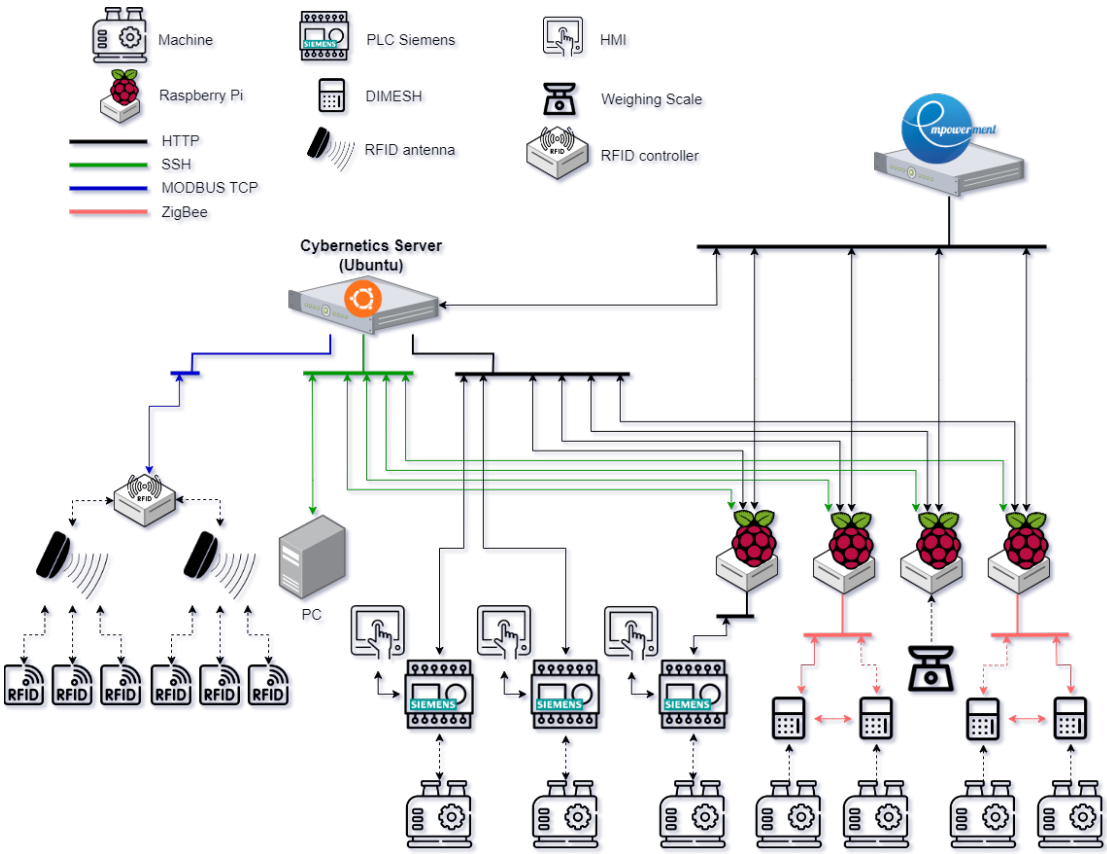

Figure 4.2: Current system structure.

These compact computers are strategically deployed throughout the industrial environment, enabling real-time data acquisition, processing, and transmission at the network's edge. They are strategically positioned to interface with a wide array of machinery, devices, and sensors, including those that were previously unmonitored. This strategic deployment enhances the capabilities of the Cybernetics server by extending its reach and data acquisition capabilities.

Prometheus Node Exporter: To complement the data acquisition process from Raspberry Pi devices, the Cybernetics server leverages Prometheus Node Exporter, a tool that communicates via HTTP. This element empowers the server with the capability to retrieve health-related data. It offers insights into the well-being of not only the Raspberry Pi devices but also the server itself. This health monitoring feature is indispensable, guaranteeing the seamless operation of the devices and enabling rapid responses to any potential issues.

Ansible Automation: The Cybernetics server sets new standards in automation

of Infrastructure as Code (IaC) by incorporating Ansible. This automation solution establishes SSH connections with Raspberry Pi devices, ushering in a streamlined approach to device management. It encompasses essential tasks, such as automated software updates, installations, and device configurations as needed. The automation element optimizes operational efficiency, minimizing the need for manual interventions.

RFID Reader Controller: Beyond its existing functions, the Cybernetics server also performs data acquisition through interfacing with an RFID reader controller that utilizes the Modbus TCP protocol. This integration allows the ability to track RFID tags within textile rolls. This supplementary layer of data enriches the system's capability to gather comprehensive information.

DIMESH Integration: In a major step towards data enrichment and performance optimization, the company has developed the so called "DIMESH" devices. These devices, rooted in a microcontroller and Zigbee communication, were created in-house and they serve the function of capturing machine state data, Process and Transactional variables, directly from each machine, and some input codes given by the machine operator. DIMESH devices utilize an array of sensors and electronics to extract this information. By dedicating a DIMESH device to each machine, one or more Zigbee mesh networks are formed in each plant, with each device playing the role in gathering insights of the machines that directly impact product quality and industrial process efficiency. Raspberry Pi devices take on the role of gateways within this network (Zigbee master), ensuring that data from DIMESH devices is effectively transmitted to the Empowerment database.

4.2.2 Methods and Devices

Node-RED Services for PLC Data Capture: To capture data from variables stored in the PLCs' databases, notably Siemens S7, Node-RED services are employed. These services leverage the 'node-red-contrib-s7' node to perform HTTP REST requests for reading and writing variables in the PLC databases. It's worth noting that each PLC may encompass one or more databases, each housing critical variables such as temperature, humidity, velocity, RPM, roll number, operator IDs, and various others.

For enhanced data reliability and to prevent conflicts during the data sending process, a queuing mechanism is implemented based on the 'node-red-contrib-queue-gate' node. This queue ensures that data is sent in a systematic order, avoiding issues where the server might reject packages sent before processing

the latest data. In this process, as depicted in the Figure 4.3, which shows the process for only one PLC and one or multiple databases, the queue awaits the confirmation response from the server for the last data received before transmitting the next data package. This queuing strategy optimizes data transmission, maintaining a smooth and error-free flow of information.

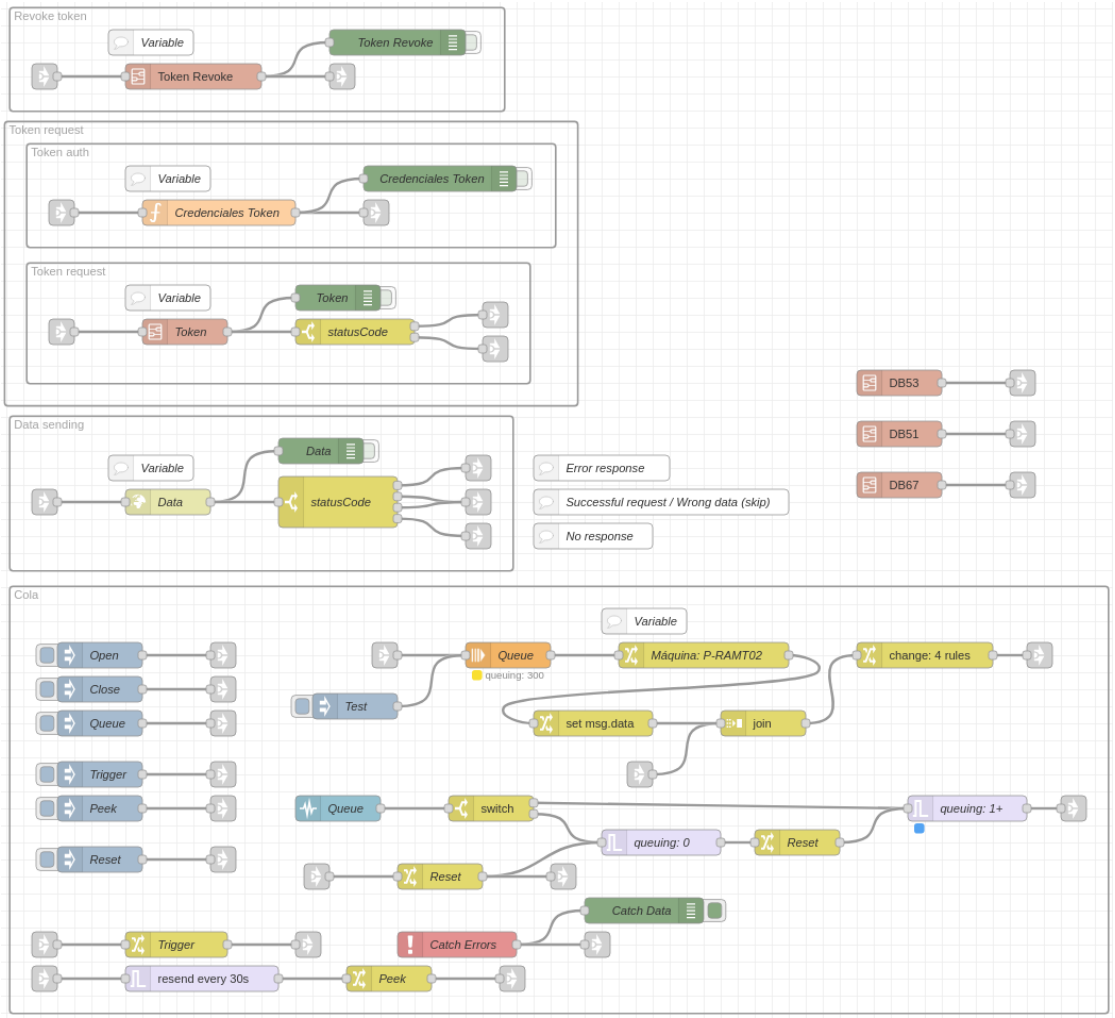

Figure 4.3: Node-RED data sending method.

In Figure 4.4, we can observe the queuing of data from the databases within the same PLC. This queuing mechanism ensures the orderly transfer of information. On the other hand, Figure 4.5 illustrates the operation of the 'S7 in' node in the 'node-red-contrib-s7' library. This node efficiently collects data from various variables when provided with the respective address configurations. For example, 'DB51,WORD4' corresponds to the 'AJUST_VELOCIDAD_MAQ'

variable.

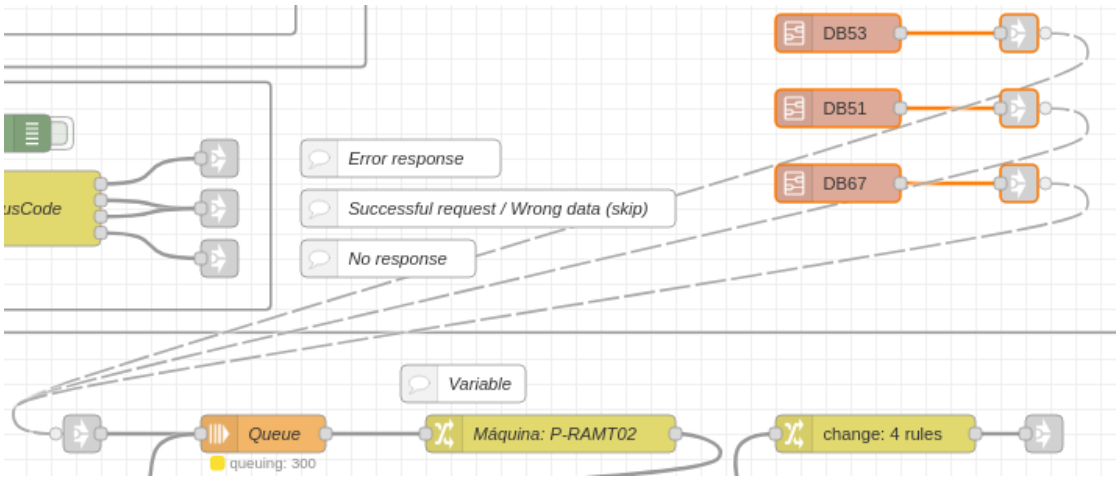

Figure 4.4: Node-RED data queuing from PLC databases.

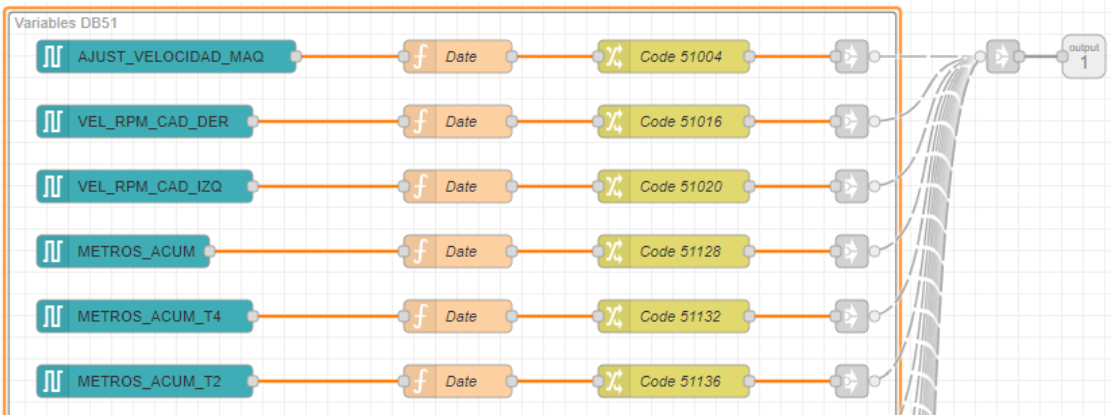

Figure 4.5: Node-RED database subflow inputs.

For instance, Node-RED is utilized to acquire data from multiple databases within any PLC with API REST activated, for example to gather the data given by the PLC controlling the Rama machines, some of the mos important components in the company's operations, among other machines PLCs.

Python Applications for Data Capture: Additional data capture methodologies involve Python applications. These applications, executed on either Raspberry Pi devices or the Cybernetics server, operate as services that send the data using HTTP requests. They serve diverse purposes, including:

- Capturing Weight from Weighing Scales using Raspberry Pis.

- Capturing the Width of Textile Rolls, obtaining measurements from a fabric inspection machine's PLC via a Raspberry Pi.
- Sending Fabric Dyeing Data, which are critical quality parameters, for example "color fastness" which refers to the ability of a dye to retain its original color without fading when it is exposed to conditions such as moisture, washing, or light. It is one of the most critical issues in the textile industry and this data is gathered from the chemical laboratory PCs to Empowerment for use in the fabric finishing process.

The `HTTPSCommunication` class in this work implemented in Python serves as a crucial component in these applications, enabling secure communication with the Empowerment data repository. Here's an overview of the key points and functionalities of this class:

1. **Initialization**: The class is initialized with essential parameters, including a username, password, and entity name for authentication and data retrieval. These parameters are crucial for establishing a connection with the external data source.
2. **Token Management**: One of the primary functions is to handle access tokens. It requests and retrieves an access token for authentication, which is required for making secure HTTP requests to the data source.
3. **Data Transmission**: The class provides methods for sending, reading, and modifying data at the specified URL. It allows for the exchange of data with the external source, facilitating data updates and retrievals.
4. **Token Revocation**: In cases where it's necessary to close a session, the class handles token revocation. It ensures that the session is closed and the token file is deleted, enhancing security and access control.
5. **Headers Management**: The class efficiently manages HTTP headers, including bearer tokens, to ensure secure and authenticated communication with the data source.
6. **Exception Handling**: Error handling and response validation are integrated into the class to provide robust communication with the external data repository.

The code of the `HTTPSCommunication` class to send the data is provided in Appendix Listing A.1 (page 63) to illustrate the Python script used in this processes.

RFID Tracking Application: A unique application focuses on RFID tracking of tags in fabric rolls. This system employs multiple Pepperl+Fuchs RFID

antennas strategically positioned at various locations. A controller/gateway, communicating via Modbus TCP with the Cybernetics server, manages these antennas. The server issues commands to the gateway for reading operations, configuration adjustments of the antennas, and collection of data from the RFID tags. The information retrieved from the antennas is transmitted as a response to the server, subsequently forwarded to Empowerment for data integration.

As an overview of the key points and functionalities of this application code provided in Appendix Listing A.2 (page 67) we have:

1. **Modbus TCP Configuration**: The code starts by configuring the Modbus TCP server details, including the server's IP address (SERVER_IP) and port number (SERVER_PORT).
2. **Modbus TCP Client Setup**: A Modbus TCP client instance is created using the configuration details. This client will establish a connection with the RFID equipment and manage data exchange.
3. **Extract Tag Information Function**: The code defines a function named extract_tag_information that is responsible for processing the data obtained from RFID tags. It extracts information such as UII (Unique Item Identifier) and TID (Tag Identifier). The extracted information is then formatted for further use. The extraction parameters and the codes sent to the gateway are specific instructions and registers to write and read referenced by the manufacturer.
4. **Main Loop**: The code enters a main loop where it performs the following actions:
   (a) Connects to the Modbus TCP server.
   (b) Initializes a list called known_tags to keep track of RFID tags that have been read.
   (c) Within the loop, it writes specific values to Modbus registers to request tag data. This is done using the client.write_registers function.
   (d) It continuously reads data from the Modbus registers using the client.read_holding_registers function and checks for any new RFID tags.
   (e) If a new tag is detected and it's not already in the known_tags list, it's added to the list, and information about the tag is printed.
   (f) The code keeps track of the total number of tags read and the number of unique tags in the known_tags list.
5. **Modbus Error Handling**: The code includes error handling for Modbus-related errors, ensuring that any issues during communication are properly handled.

6. **Client Cleanup**: Finally, Modbus client is closed using the client.close() method to release any resources and terminate the connection.

Turbidity Measuring Application: The system incorporates a specific Python application for cost-effective water turbidity monitoring. Operating on a Raspberry Pi, this application leverages the OpenCV library for visual monitoring. A camera observes the color of incoming water within a white light chamber, as depicted in Figure 4.6 and Figure 4.7. By comparing the brightness of captured frames to a reference frame representing ideal water transparency, the application calculates turbidity. Importantly, the turbidity data is not transmitted to Empowerment but is directly relayed to the respective PLC via Modbus TCP. The PLC controls valves regulating water entry.

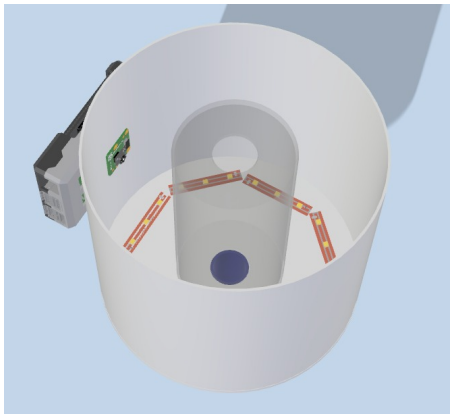

Figure 4.6: White light chamber inside, LED, Raspberry Pi and camera placing.

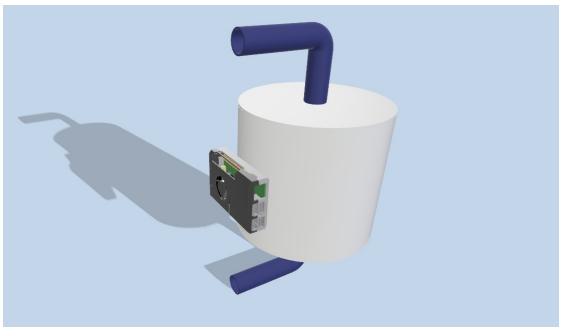

Figure 4.7: White light chamber outside and mounting position.

To provide an overview of the application code found in Appendix Listing A.3 (page 69), we present the following:

1. **Importing Necessary Modules**: The code starts by importing essential

Python modules, including those for Modbus communication, computer vision (OpenCV), Raspberry Pi camera control, and date/time handling.

2. **Modbus Configuration**: It establishes a connection with a Modbus PLC (Programmable Logic Controller) for data exchange. The IP address and port number of the PLC are configured. This connection is crucial for sending turbidity data to the PLC.
3. **Camera Setup**: The code sets up the Raspberry Pi camera, specifying its resolution and exposure settings. Additionally, variables are initialized for managing image frames.
4. **Frame Capture Loop**: The primary loop captures video frames from the camera. It calculates turbidity in real-time by comparing the current frame with a reference frame. The brightness differences between frames are used to calculate turbidity.
5. **Turbidity Calculation**: Turbidity is calculated by measuring pixel-level differences between frames. These differences are normalized to determine turbidity values, which are then sent to the PLC for control.
6. **Image Storage**: The code periodically captures and stores images for logging purposes. It saves images in directories, creating subdirectories for each day, and removes older images to manage storage efficiently.

Additionally, it's important to highlight images from practical tests, showcasing how the solution works and presenting the associated measurements shown in Appendix C (page 79). These tests involved the utilization of four testing tubes, each containing different water samples with varying turbidity levels as shown in Figure C.1. The objective of these tests was to ensure the precision and reliability of the system in monitoring water turbidity. For each test tube, the system's camera and computation were used to capture the images and compute the corresponding turbidity values shown in Figures C.2, C.3, C.4 and C.5.

In the company the Siemens WinCC Supervisory Control and Data Acquisition (SCADA) application is used to depict the information taken from various specific and crucial PLCs in the plants, and among them the PLC that controlls the inflow of water to the production system. Figure 4.8 illustrates the seamless flow of information from the Raspberry Pi-based turbidity measurement system to the WinCC application, making it available for monitoring and decision-making.

The data transmitted to the PLC is accessible for viewing through the WinCC system, as shown in Figure 4.9. This visualization clearly displays the measurement of water turbidity in the incoming flow.

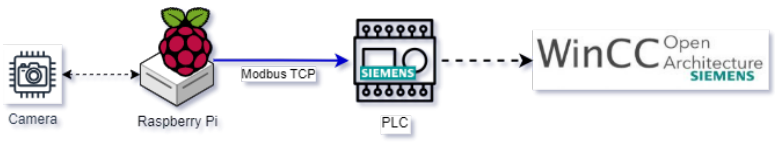

Figure 4.8: Flow of information from the Raspberry Pi measuring turbidity to the WinCC application.

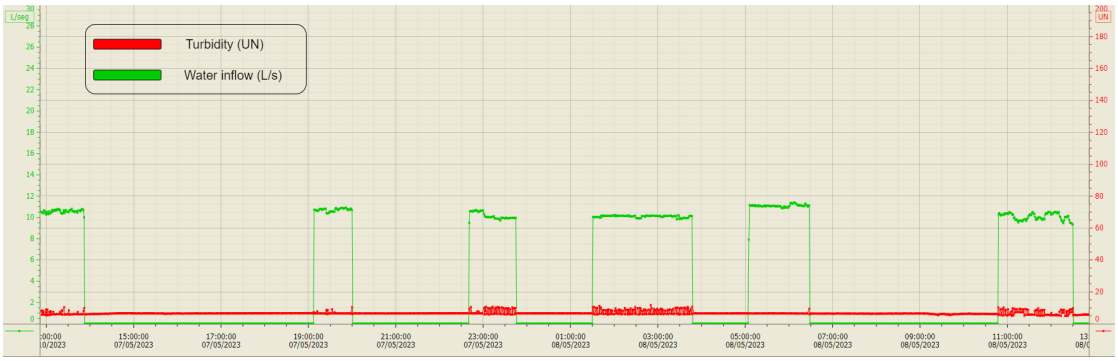

Figure 4.9: WinCC graph with water inflow to the production system in green (L/s) and the measured turbidity in red (units).

4.3 DevOps Approach

In the implementation of the system, a DevOps approach is adopted to enhance efficiency and maintain a seamless workflow. DevOps principles emphasize collaboration, automation, and monitoring throughout the software development and deployment lifecycle. This section discusses the implementation of DevOps practices, specifically focusing on the use of Ansible, Prometheus (with Node Exporter), and Grafana as shown in Figure 4.10, to ensure the reliability and scalability of the system.

4.3.1 Version Control

Version control is an essential aspect of the DevOps strategy. Git, a distributed version control system, is used to manage source code and keep track of changes in a structured manner. Git enables:

1. **Track Code Changes:** The system maintains a central repository that houses the project's source code. Developers create branches for specific features or fixes, and changes are tracked in a systematic way.

2. **Collaboration:** Git supports collaborative development, allowing multiple

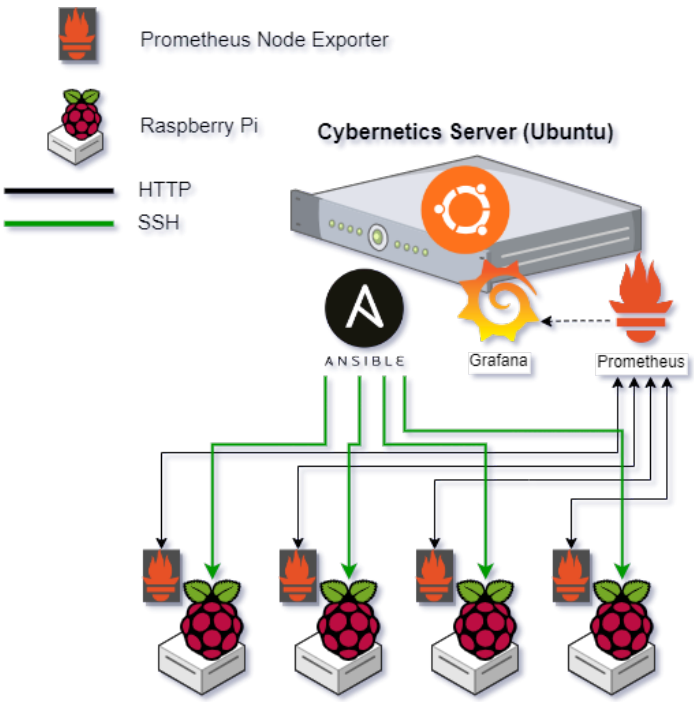

Figure 4.10: Cybernetics server's Ansible, Prometheus, and Grafana implementation.

team members to work on the same project simultaneously, merge their changes, and resolve conflicts.

3. **Change History:** Git keeps a comprehensive history of code changes, making it easy to trace when and why specific alterations were made.

The use of Git extends beyond programming languages and deployment stages, maintaining a coherent and organized history of changes in both hardware and software components of the system, ensuring transparency and traceability.

4.3.2 Infrastructure as Code (IaC)

The Infrastructure as Code (IaC) paradigm is fundamental to the DevOps strategy. As explained in subsection 4.2.1, Ansible, an open-source automation tool, is employed to define and manage infrastructure using code. IaC practices are essential for ensuring the reliability, scalability, and efficiency of the system. The detailed Ansible YAML files used in this process are presented in the Appendix B (page 73) for reference and further examination. The process includes several key steps, as outlined below:

1. **Automated Deployment:** Ansible automates the provisioning and configuration of servers and devices, including Raspberry Pis, within the system. It takes charge of deploying code, configurations, and executing tasks like updates and software installations.

2. **Configuration Management:** Infrastructure components, along with their configurations and desired states, are defined in Ansible playbooks. This approach ensures uniformity and the ability to reproduce the system's environment reliably.

3. **Scalability:** As the system expands, Ansible facilitates smooth scalability by accommodating the addition of new devices and servers. This is managed through an inventory '.ini' file, as illustrated in Listing B.1 (page 73), guaranteeing a consistent and dependable environment across all system components.

4. **SSH Communication:** Ansible employs SSH (Secure Shell) for establishing secure and efficient communication with the target devices, as shown in Listings B.2 and B.3 (page 74). This method ensures the management of remote systems is both reliable and robust.

5. **Periodic Updates and Maintenance:** To maintain the system's smooth operation, Ansible is employed to periodically update and upgrade components, as demonstrated in Listing B.4 (page 75). Additionally, Ansible manages any necessary reboots.

6. **Prometheus Node Exporter Installation:** To monitor the system's health and performance, Ansible is applied to install Prometheus Node Exporter on all relevant devices after creating the required Prometheus user and group as shown in Listings B.5 and B.6 (page 76).

The integration of Ansible in the DevOps approach is an essential element in the system's development and operation. These automation practices reduce manual intervention, enhance system reliability, and minimize the risk of errors.

4.3.3 Monitoring, Alerting, and Visualization

Ensuring the health and performance of the system is of prime importance, and for this purpose, a combination of monitoring, alerting, and visualization tools is employed, with a primary focus on Prometheus Node Exporter and Grafana.

1. **Prometheus Node Exporter:** The Prometheus Node Exporter serves as a pivotal component in the system's monitoring infrastructure. It constantly collects (scrapes) a diverse array of system metrics from nodes within the

network. These metrics encompass a wide spectrum of critical data, including but not limited to CPU usage, RAM utilization, system temperature, disk I/O, and network statistics. By aggregating this information, Prometheus Node Exporter provides a comprehensive and real-time view of the system's health. This real-time monitoring capability ensures that deviations from expected performance are detected promptly.

2. **Grafana:** Grafana plays a crucial role in visualizing the wealth of data collected by Prometheus Node Exporter. It offers a versatile and user-friendly platform for creating custom dashboards and interactive charts, enabling a detailed analysis of system performance. The visual representation of data in Grafana enhances the understanding of the system's operational status and enables swift identification of performance trends, bottlenecks, or anomalies.

3. **Alerting:** Beyond mere observation, the system's monitoring goes a step further by implementing proactive alerting mechanisms. Prometheus, the underlying engine for Node Exporter, is meticulously configured to trigger alerts when specific performance thresholds are breached. These alerts are designed to be highly customizable, ensuring that they align with the system's operational requirements and the specific parameters deemed vital. This approach enables a proactive response to issues, minimizing downtime, and preserving system stability. Timely notifications are dispatched to designated personnel, ensuring that potential problems are addressed promptly and effectively.

The integration of Prometheus Node Exporter and Grafana not only facilitates comprehensive monitoring but also empowers the system with the ability to visualize and analyze performance data. This insight into the operational status of devices and the broader system architecture enables timely actions, resulting in the efficient maintenance of system reliability. As demonstrated in the sample dashboard provided in Figures D.1 and D.2 in the Appendix D, Grafana allows for the creation of custom dashboards and interactive charts, providing a clear and user-friendly way to visualize critical system metrics utilizing the relevant queries to retrieve the data.

The incorporation of DevOps principles plays a crucial role in the progress of the CPS. It ensures seamless collaboration, automates processes, and enables effective monitoring, thereby enhancing the efficient administration of complex systems and swift resolution of operational issues. Within an industry that prioritizes precision and reliability, the DevOps approach significantly enhances the development, deployment, and maintenance of such systems.

Chapter 5

LoA Measurement Assessment

With the introduction of technological advancements in industrial processes, there is a critical need to meticulously assess the different Levels of Automation (LoA) to ensure effective and efficient operations. This chapter presents a comprehensive assessment of the LoA measurements, elucidating its impact on various stages of the data acquisition and preparation framework. Each implemented method (Node-RED, Python, RFID tracking, and Turbidity measurement applications) undergoes a scrutiny within the paradigm of LoA, clarifying the evolution from pre-implementation to the current state of the application. Different stages of each process, as presented in Figure 5.1, were taken into account, and each of them has its own LoA assessment to subsequently calculate the general LoA in the information dimension of each development.

Figure 5.1: Implemented stages of data management

Noteworthy, in this research we are using different LoA states and only in the information field as presented in Table 5.1, given that the processes we want to automate are data driven developments.

5.1 Node-RED Applications LoA Assessment

The integration of Node-RED developments into this data acquisition ecosystem marks a transformative situation in the automation panorama. The LoA assessment

Levels	Information
1	Manual Operation
2	Assisted Manual Operation
3	Partial Automation
4	Conditional Automation
5	High-Level Automation
6	Full Automation with Oversight
7	Fully Autonomous Operation

Table 5.1: Levels of Automation for this work information applications.

of these Node-RED applications takes part in the different stages or tasks in the implementation as described below and in Figure 5.2.

System Configuration: The integration of Node-RED developments expedites system configurations, fostering an enhanced LoA in system adaptability. The system's agility to seamlessly interface with diverse PLCs, implement efficient queuing strategies, and streamline systematic data flows signifies an elevated LoA in configuring the broader industrial framework. This advancement allows for the automated management of data and configuration tasks, marking a significant improvement from previous implementations where only the PLC programming involved computational assistance, leaving the remaining system stages reliant on manual methods for data gathering.

As a result, the current LoA assigned to the System Configuration stage has risen to level 4, a substantial improvement from the previous implementation where it was situated at level 2.

Data Acquisition: Despite the seamless integration with Siemens S7 PLCs and the implementation of robust queuing mechanisms, the incorporation of Node-RED services for PLC data capture doesn't yield a significant improvement in the efficiency and comprehensiveness of data acquisition. As the PLCs are already responsible for the acquisition from sensors and actuators, so the LoA in data acquisition remains consistently at level 5.

Data Transformation: A remarkable improvement is observed in Data Transformation, with the LoA increasing significantly from 1 to 7. Node-RED has allowed to excel in the real-time transformation of raw data into meaningful information, showcasing substantial enhancements in processing capabilities. The flexibility to capture data from multiple databases within any PLC with activated REST APIs exemplifies its sophisticated transformation capability.

The systematic structuring of data from diverse sources further contributes to a higher LoA in preparing information for downstream processes.

Data Transmission: Node-RED's proficiency in data transmission significantly enhances the LoA in conveying critical information. The methodology's capability to execute HTTP REST requests for reading and writing variables in PLC databases and sending data to the Empowrment server ensures a swift and orderly data transmission process, replacing the previous SQL data transmission made with barcode-dependent Radio Frequency (RF) terminals or PC data entry.

Data Transmission exhibits a noteworthy advancement, with the LoA increasing from 5 to 7. Node-RED has strengthened its capabilities in transmitting data efficiently, contributing to a more responsive and agile data acquisition system. This enhancement in data transmission aligns with the goal of achieving real-time insights and responsiveness, fostering a more robust industrial data acquisition framework.

Error Handling: Robust error handling mechanisms embedded in Node-RED developments fortify the LoA in maintaining data integrity. The implementation of queuing mechanisms mitigates potential conflicts during data transmission, fostering a fault-tolerant architecture. This proactive approach in error handling ensures a resilient data flow, minimizing disruptions and optimizing operational continuity.

On this regard, there is a noteworthy improvement in Error Handling, with the LoA rising from 2, where there was no dedicated error handling beyond the ability to manually check databases or terminal messages, to a LoA of 6. In this enhanced level, errors in authentication, data transmission and transformation are handled automatically by the Node-RED development. This signifies a substantial advancement, showcasing a more robust and fault-tolerant design, enhancing the ability to handle errors effectively during data processing.

Maintenance and Updates: The Node-RED methodology introduces a new paradigm in maintenance and updates, elevating the LoA in system management. Leveraging the visual aids provided by Node-RED for developments, the LoA in maintenance and updates reflects an efficient and adaptive approach, aligning with the dynamic needs of industrial operations.

The Maintenance and Updates task improves the LoA from a level 2, where any maintenance and update was done manually on each PLC and stage of the implementation, to a level 3. In this improved state, assisted visualization

and configuration through Node-RED provide effectiveness in maintaining and updating the systems. This indicates a better level of efficiency in managing and updating the system.

Monitoring and Reporting: The monitoring and reporting capabilities of Node-RED have undergone significant advancements in the digitalization tasks, resulting in a noteworthy increase in the LoA from 2 to 4. This heightened automation facilitates more effective oversight and reporting, providing comprehensive insights and improved accuracy.

Node-RED's real-time insights derived from PLCs, coupled with improved queuing, error handling, and enhanced messaging and debugging interfaces, contribute to the creation of a dynamic monitoring environment. This proactive approach ensures the timely and accurate dissemination of critical information, further enhancing the overall monitoring and reporting capabilities within the industrial framework.

The average LoA for the Node-RED developments has increased from 3 to 5, as shown in Figure 5.2. Node-RED applications' substantial improvements in Data Transformation, Data Transmission, and Error Handling underscore its growing significance in supporting a more sophisticated and efficient data acquisition system. These advancements contribute to an overall elevation in the LoA, reflecting the enhanced capabilities and impact of Node-RED on various aspects of the industrial data acquisition framework.

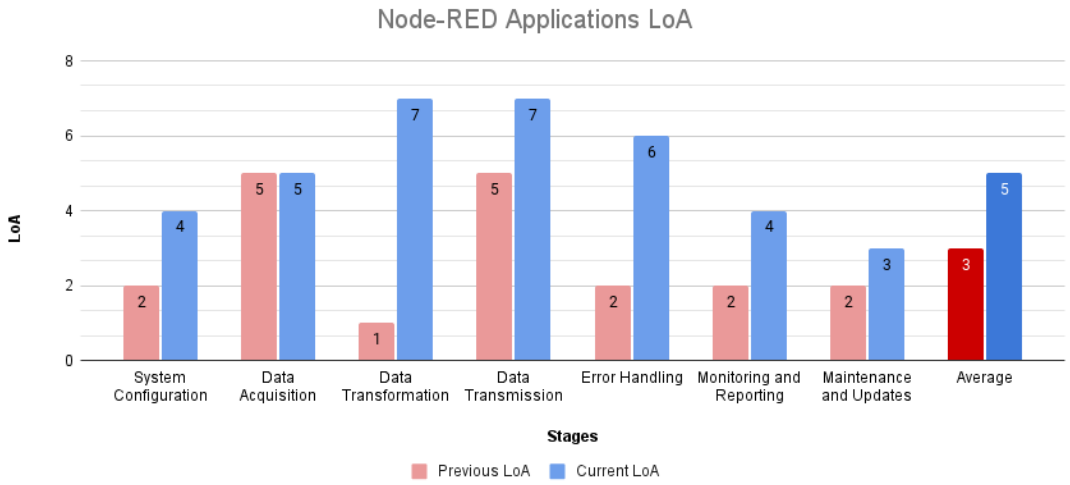

Figure 5.2: Node-RED Applications stages and average LoA Assessment

5.2 Python Applications LoA Assessment

It's evident that the integration of Python advancements into our data acquisition toolkit has introduced a diverse range of functionalities, collectively contributing to the comprehensive LoA, noted ahead in Figure 5.3. Upon closer examination of the distinct application tasks, it becomes clear that a transition from traditional to modern processes is an improvement. Notably, it is important to highlight that information acquisition implementations, preceding these specific developments, were predominantly manual in nature, with tools primarily present in the Data Acquisition stage and Data Transmission only.

System Configuration: Our Python developments contribute to an advanced LoA in system configuration, optimizing the industrial data acquisition landscape. The integration of the HTTPSCommunication class (Appendix A) facilitates secure communication with the Empowerment data repository, streamlining system configurations for both Raspberry Pi devices and the Cybernetics server. The LoA in system configuration reflects an adaptive and efficient approach, ensuring the seamless orchestration of diverse data sources.

Furthermore, system configuration shows a gradual improvement, with LoA rising from 1 (entirely manual) to 3. This transition involves leveraging programming tools, Ansible automations and computation infrastructure to develop the application logic.

Data Acquisition: Python developments slightly elevate the LoA in data acquisition, introducing specialized functionalities tailored to manual inputs or file reading. Ranging from capturing weight from weighing scales to obtaining fabric dyeing data and measuring the width of textile rolls, these advancements showcase versatility and precision, ensuring a comprehensive understanding of multifaceted industrial parameters.

A modest improvement is observed in Data Acquisition, with the LoA increasing from 3 to 4. This improvement signifies the complementation of using of tools for pure data acquisition, with further enhancements using tools like barcode reading handhelds, providing a more advanced and automated approach to data collection.

Data Transformation: Data transformation from these Python developments stands as one of the most important tasks, addressing diverse data types and formats with specialized functionality. The HTTPSCommunication class, serving as the core in these implementations, facilitates secure communication with

the Empowerment data repository. The elevated LoA in data transformation ensures seamless integration into the global data preparation pipeline.

Much like Node-RED developments, Python's demonstrate a significant improvement in Data Transformation, with the LoA soaring from 1 to 7. This enhancement highlights the advanced capabilities of these Python applications in processing and structuring data effectively. The improvement in this stage is particularly driven by the need for readable and transmittable data to the Empowerment databases, showcasing Python's prowess in preparing data for downstream processes in a comprehensive and efficient manner.

Data Transmission: The data transmission capabilities embedded in our Python developments significantly elevate the LoA in conveying critical information. These applications leverage HTTP requests for swift and secure data transmission. In this way, the LoA in data transmission reaches a sophisticated state, ensuring the prompt delivery of crucial insights.

Data Transmission undergoes a substantial improvement, with the LoA surging from 3 to 7. Python's enhanced capabilities in automatically transmitting data represent a noteworthy shift from previous manual methods, which involved using tools like computers or tablets.

Error Handling: Python developments showcase a remarkable improvement in the LoA for error handling, escalating from 1 to 6. The HTTPSCommunication class plays a pivotal role in managing access tokens, addressing data transmission errors, and ensuring secure communication. This substantial increase in LoA signifies Python's enhanced ability to automatically handle errors, providing effective solutions for communication and authentication failures. This advancement strengthens the reliability of data exchanges, fortifying the foundation of the industrial data flow and contributing to a more resilient data acquisition system in the face of potential challenges.

Monitoring and Reporting: Python developments demonstrate a potential improvement of LoA in monitoring and reporting. The real-time insights and logs derived from capturing weight, textile roll width, and fabric dyeing parameters, combined with comprehensive error handling, contribute to a dynamic monitoring environment. The elevated LoA in monitoring and reporting ensures the timely and accurate dissemination of critical information, facilitating informed decision-making within the production environment.

There is a moderate improvement in Monitoring and Reporting, with the LoA increasing from 1 to 3. These Python application's contributions amplify

the system's ability to provide more comprehensive insights into industrial processes.

Maintenance and Updates: Python developments introduce an improved LoA in maintenance and updates, streamlining essential tasks for both Raspberry Pi devices and the Cybernetics server. The automation of tasks through Ansible ensures accurate device management, minimizing manual interventions and automating software updates, installations, and device configurations.

Consequently, there is a notable improvement in Maintenance and Updates, with the LoA increasing from 1 to 4. Python's contributions, especially through tools like Ansible, enhance the efficiency of managing and updating the system, marking a significant advancement in automation.

The average LoA for the Python implementations carried out in this thesis has experienced a notable improvement, progressing from an initial level of 2 to a more advanced level of 5 as depicted in Figure 5.3. The most significant contributions to this improvement came from tasks such as Data Transmission, Data Transformation, and Error Handling, showcasing Python's pivotal role in automating critical processes within the industrial data acquisition system. The transition from manual processes to a more automated and sophisticated approach has marked a substantial leap in the efficiency and reliability of the overall system.

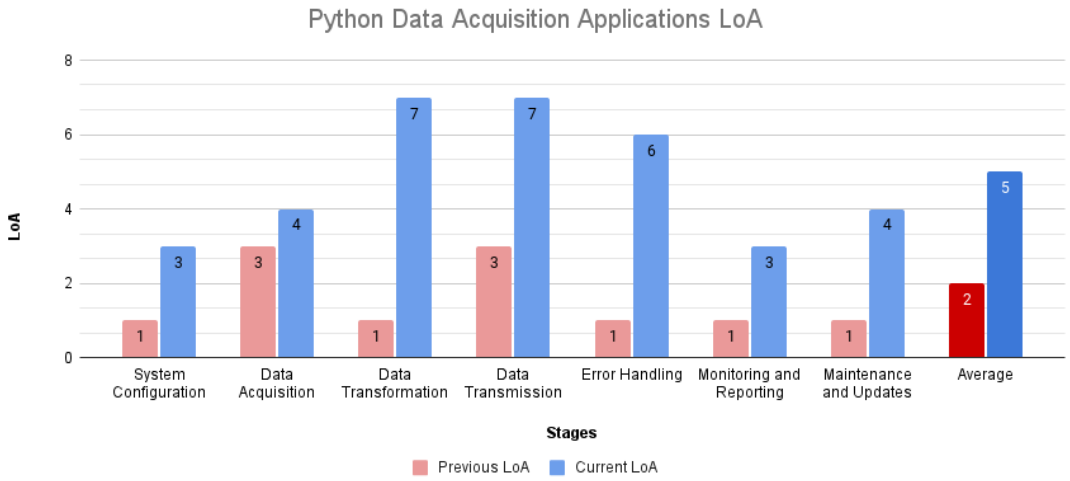

Figure 5.3: Python Application stages and average LoA Assessment

5.3 RFID Tracking LoA Assessment

RFID tracking plays a crucial role in enhancing the efficiency of our information gathering processes, particularly in the inventory tracking of production materials, machines, devices, and more. Although Python data transmission is used in this implementation, we treat RFID tracking as a distinct application for assessment, given that this innovative technology represents a new and integral component of our automation strategy, aiming to significantly improve the effectiveness of information acquisition within the company. Through the systematic evaluation of its LoA shown in Figure 5.4, we gain a comprehensive understanding of how RFID tracking contributes to the overall automation landscape, optimizing data acquisition and fostering operational excellence.

System Configuration: RFID tracking contributes to a modest LoA in system configuration by seamlessly integrating Modbus TCP communication for efficient data exchange and the Python HTTPSCommunication class. The configuration of RFID antennas and the controller/gateway establishes a robust foundation for real-time tracking, showcasing the advanced capabilities of RFID technology in coordinating with our data acquisition framework. However, there is no substantial improvement in LoA for system configuration, maintaining a level of 2. The complexity of the tracking system's configuration involving manual planning persists, although the programming aspect receives assisted support. It's essential to note that the traditional tracking system also undergoes a similar configuration process, leading to a consistent LoA level.

Data Acquisition: The RFID tracking development introduces a paradigm shift in data acquisition, specifically in tracking RFID tags within fabric rolls and boxes. Through its interface with RFID antennas and the controller/gateway, this development achieves a heightened LoA in acquiring real-time data. The capacity to track tags significantly enhances the system's capability to gather comprehensive information.

Data Acquisition witnesses a substantial improvement, with the LoA increasing from 2 to 6. The RFID implementation shows improved capabilities in acquiring data from RFID tags, marking a significant advancement in the data acquisition framework. This improvement is particularly noteworthy, given that the previous tracking implementation relied on manual tasks aided only by Radio Frequency (RF) handheld terminals, resulting in inconsistent data acquisition times compared to the actual moment elements changed locations in the production line.

Data Transformation: The RFID tracking development demonstrates a commendable LoA in data transformation. Through the extraction of information such as UII (Unique Item Identifier) and TID (Tag Identifier) from RFID tags, along with the formatting of the extracted data, the system showcases a sophisticated transformation capability. The development ensures that data from RFID tags is structured and ready for seamless integration into the broader data preparation pipeline, thereby enhancing the overall LoA in data transformation.

While Data Transformation maintains a high LoA of 7, there is a slight improvement. This RFID development continues to excel in transforming raw RFID tag data into structured and meaningful information. This improvement is notable, considering the previous implementation's high LoA of 6, which already benefited from the aid of data interpretation from RF terminals. However, the current RFID tracking system achieves a Level 7 due to the complete automation of data acquisition, eliminating the need for constant logins and authorizations required in the previous method.

Data Transmission: The RFID tracking development exhibits a sophisticated LoA in data transmission. Leveraging Modbus TCP communication, the development seamlessly conveys information about RFID tag movements to the Cybernetics server and then to the Empowerment server using the Python HTTPSCommunication class. This real-time data transmission enhances the overall responsiveness and agility of the data acquisition framework. The LoA in data transmission attains a heightened state, facilitating prompt decision-making based on the dynamic movements of fabric rolls.

Data Transmission maintains a high LoA of 7, indicating consistent efficiency in transmitting RFID tag data. The implementation ensures prompt and reliable transmission of critical information. It's worth noting that the previous implementation also had a Data Transmission LoA of 7, given that the transmission itself was already parametrized and automatic.

Error Handling: The RFID tracking development exhibits a robust LoA in error handling. The implementation includes error handling for Modbus-related errors, ensuring that any issues during communication with RFID antennas are properly addressed. The development's fault-tolerant design contributes to a higher LoA in error handling, maintaining the reliability of RFID data acquisition even in challenging conditions.

Error Handling sees a notable improvement, with the LoA increasing from 2 to 5. RFID demonstrates a more robust design, enhancing its ability to handle

errors effectively during data acquisition. It's important to note that while there is significant progress, considering the complex nature of tracking multiple tags in the same location, the possibility of oversight remains, necessitating occasional manual checks to ensure accuracy, especially in scenarios involving numerous fabric rolls on transport carts.

Monitoring and Reporting: RFID tracking development introduces a commendable LoA in monitoring and reporting. By strategically positioning RFID antennas and interfacing with the controller/gateway through Modbus TCP, this development provides real-time insights into the movement of fabric rolls, boxes, machines, and devices. The LoA in monitoring and reporting ensures the timely availability of critical information, contributing to informed decision-making within the industrial environment.

Monitoring and Reporting witness a moderate improvement, with the LoA increasing from 2 to 4. RFID contributes to more comprehensive insights into the movement of fabric rolls. It's essential to note that the improvement, while significant, reflects the transition from manual monitoring and reporting in the previous implementation, aided by computers and SQL querying, to a more visually accessible and comprehensive monitoring through the Empowerment platform.

Maintenance and Updates: The RFID tracking development extends the LoA in maintenance and updates through efficient communication with the controller/gateway using Modbus TCP. This automated approach optimizes device management, enhancing operational efficiency.

Maintenance and Updates maintain a consistent LoA of 3. RFID demonstrates continued efficiency in managing and updating its components. It's important to note that the maintenance and updates in this context primarily involve adjustments related to IP address changes or network configuration modifications rather than routine updates to the devices. This is similar from the previous implementation, where maintenance and updates were primarily focused on RF handheld terminals' OS and the SQL database of the company, if necessary.

The average LoA for RFID has increased from 3 to 5, as illustrated in Figure 5.4. This notable improvement is particularly evident in the areas of Data Acquisition and Error Handling, highlighting RFID's growing significance in reinforcing a comprehensive data acquisition system, with a specific focus on tracking inventory movements. The enhanced capabilities in these crucial aspects signify the successful

integration of RFID technology, contributing to the overall efficiency and reliability of the information gathering processes within the company.

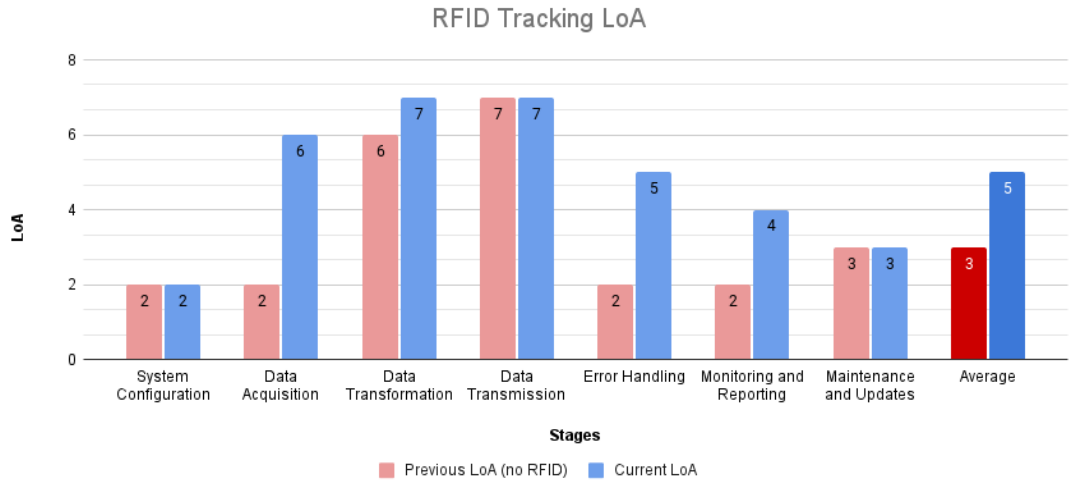

Figure 5.4: RFID Tracking Application stages and average LoA Assessment

5.4 Turbidity Measurement LoA Assessment

The integration of the Turbidity Measurement development marks a crucial milestone in our data acquisition initiatives. In this context, despite being a Python-based development, it is presented as a distinct LoA-measured implementation in this thesis given the unique nature of the data it collects, which was not previously available. The significance of this in-house project lies in its ability to monitor water turbidity, especially crucial when elevated turbidity levels could potentially lead to production losses. The LoA assessment comprehensively explores the various dimensions of this development, lighting on its substantial impact on water quality monitoring within the company's operational framework.

It's important to note that due to the absence of previous turbidity measurements, LoA for the tasks undertaken in the current implementation is being compared to a scenario of completely manual measurements, which represents an initial LoA of 1.

System Configuration: The Turbidity Measurement development contributes to a slight improved LoA in system configuration by seamlessly integrating with Modbus TCP for direct communication with the PLC. The configuration of the Raspberry Pi camera and the parameters for turbidity calculation establishes a robust framework for real-time monitoring.

System Configuration sees a modest improvement, with the LoA increasing from 1 to 2. However, it's crucial to note that the system configuration still relies on manual operations, assisted by tools for coding and machines for constructing components like the white camera and tube bypasses where the system is placed.

Data Acquisition: The Turbidity Measurement development introduces an elevated LoA in water quality data acquisition. Leveraging computer vision through the OpenCV library on a Raspberry Pi, this development monitors water turbidity in real-time. The ability to capture and transmit turbidity data directly to the respective PLC via Modbus TCP showcases a heightened LoA in acquiring critical information for water quality control.

Data Acquisition witnesses a substantial improvement, with the LoA increasing from 1 to 5. Turbidity Measurement enhances its capabilities in acquiring real-time data on water turbidity, contributing to a more comprehensive water quality monitoring system. This improvement is noteworthy, especially considering that the previous method relied on manual processes for water quality assessment, and the current implementation provides a more automated and efficient approach.

Data Transformation: The Turbidity Measurement development introduces a sophisticated LoA in data transformation. The comparison of brightness differences between frames captured by the Raspberry Pi camera, coupled with the calculation of turbidity values, exemplifies a robust transformation capability. The development ensures that raw visual data is transformed into meaningful turbidity measurements, contributing to an advanced LoA in data transformation.

Data Transformation maintains a high LoA of 7, emphasizing Turbidity Measurement's continued effectiveness in transforming raw visual data into meaningful turbidity measurements. This level of automation signifies a completely automatic and efficient process in converting raw sensor data into valuable information for water quality assessment.

Data Transmission: The Turbidity Measurement development showcases a high LoA in data transmission. Through direct communication with a PLC via Modbus TCP, turbidity data is efficiently transmitted for control and decision-making. This real-time data transmission enhances the overall responsiveness and agility of the water quality monitoring system. The LoA in data transmission achieves a advanced state, ensuring prompt actions based on dynamic changes in water turbidity.

Data Transmission maintains a high LoA of 7, indicating consistent efficiency in transmitting turbidity data. The implementation ensures prompt and reliable transmission of important water quality information, contributing to the overall effectiveness of the water quality monitoring system.

Error Handling: The Turbidity Measurement development showcases a commendable LoA in error handling. The implementation includes error handling for Modbus-related errors and potential issues during the video frame capture process. The development's robust design contributes to a decent LoA in error handling, ensuring the reliability of turbidity data even in dynamic and challenging environmental conditions.

Error Handling sees a moderate improvement, with the LoA increasing from 1 to 3. Turbidity Measurement demonstrates a more robust design, enhancing its ability to handle errors during data acquisition. It's important to note that while significant progress has been made, certain errors related to non-laminar flow of water, influenced by external factors such as temperature affecting bubble formation, are not fully automated and may require manual intervention. These specific conditions are considered in the ongoing efforts to enhance the system's resilience.

Monitoring and Reporting: The Turbidity Measurement development exhibits an acceptable LoA in monitoring and reporting water quality. By capturing real-time turbidity data and directly relaying it to the PLC, this development ensures prompt responsiveness to changes in water quality. Additionally, the real-time connection status of the Raspberry Pi is monitored, providing an additional layer of control.

As a result, Monitoring and Reporting witness a moderate improvement, with the LoA placing itself at a level 4. Turbidity Measurement contributes to more comprehensive insights into water quality parameters. Moreover, the data is depicted in a graphical interface of the Siemens WinCC system, enhancing the visualization and accessibility of crucial water quality information.

Maintenance and Updates: Turbidity Measurement enhances the LoA in maintenance and updates by implementing Ansible for continuous and automated OS updates of the Raspberry Pi. This streamlined approach optimizes device management, minimizing manual interventions and ensuring the continued reliability of water quality measurements. The implementation of Ansible for automated updates reflects an adaptive and efficient strategy, aligning with the dynamic needs of water quality control. Maintenance and Updates achieve a LoA of 4, showcasing Turbidity Measurement's continued efficiency

in managing and updating its components, particularly with the introduction of Ansible for seamless and automated Raspberry Pi OS updates.

The average LoA for our Turbidity Measurement implementation is placed at a level 5 as shown in Figure 5.5. The noteworthy improvements in Data Transformation and Data Transmission support Turbidity Measurement's growing importance in supporting a reliable water quality monitoring system, demonstrating its transformative impact on enhancing the automation and effectiveness of water quality assessments within the company.

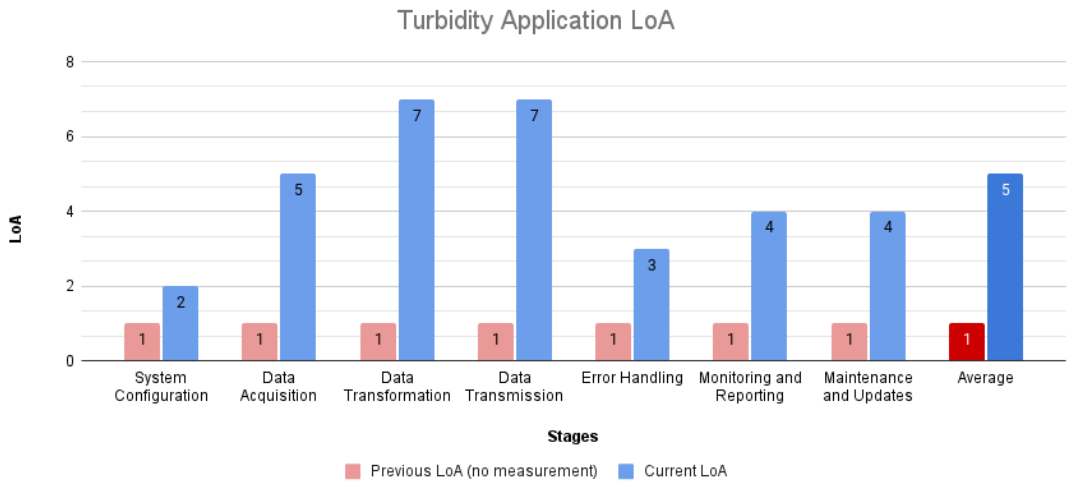

Figure 5.5: Turbidity Measuring Application stages and average LoA Assessment

5.5 LoA Assessment vs Cost of Implementation vs Implementation Difficulty

The 3D plot in Figure 5.6, provides a visual representation of the LoA for each of the implementations, comparing their previous and current states. Each bubble represents a specific task, and the position of the bubble in the plot indicates the LoA (x axis), Implementation Difficulty (y axis scale from 0 to 10), and Cost of Implementation (z axis scale from 0 to 10).

Node-RED Applications: The Node-RED developments exhibit a significant improvement in LoA, symbolized by the upward movement on the LoA axis in the plot. This positive shift indicates a successful transition towards a more

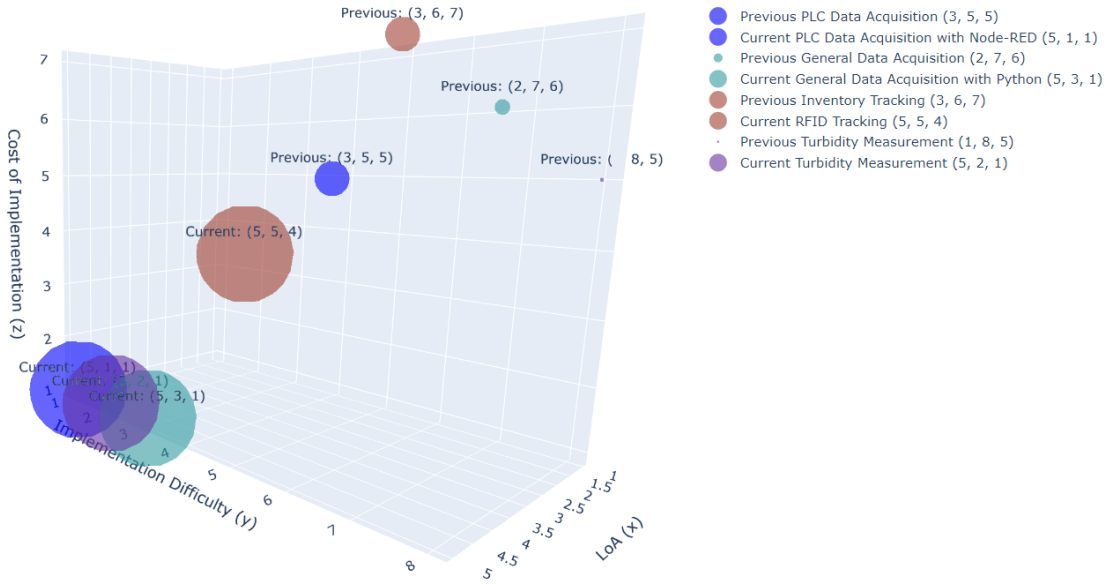

Figure 5.6: Previous and Current developments LoA vs Implementation Difficulty vs Cost of Implementation

automated process. Importantly, the implementation difficulty has decreased, signifying that the incorporation of Node-RED has streamlined the data capture process, reducing overall complexity. The cost has also decreased, pointing towards notable savings in data acquisition, processing costs, and a reduction in the time required for human labor. Consequently, the benefits derived from increased automation and operational efficiency not only justify the reduced costs but also make this implementation highly suitable for the production environment, offering a successful and cost-effective solution.

Python Applications: The Python Applications display a positive trend, marked by a notable increase in LoA, signifying enhanced automation capabilities. The implementation difficulty has seen a decrease, showcasing that the transition to a higher level of automation was achieved with reduced complexity. Importantly, the costs associated with Python-based developments have decreased, representing significant savings in the implementation of these developments. This reduction in both implementation difficulty and costs highlights the efficiency gains and improved logistics introduced by Python Applications in the production environment.

RFID Tracking: RFID tracking tasks demonstrate a significant improvement in LoA, as depicted by the upward movement of the bubble on the LoA axis of the plot. The implementation difficulty has noticeably decreased, signifying that

the incorporation of RFID technology is less complex than manual tracking methods. Moreover, there is a reduction in costs associated with RFID tracking, suggesting that the initial investment in RFID technology is recovered within a short period of time. This cost reduction aligns with the efficiency gains and enhanced capabilities introduced by RFID tracking, making it a financially viable and operationally efficient solution for the production environment.

Turbidity Measurement: The introduction of turbidity measurement showcases a notable improvement in LoA, as illustrated by the bubble's position on the plot. The implementation difficulty has significantly decreased, indicating that the challenges associated with implementing the Turbidity Measurement system were not as high as for a manual measurement. Additionally, the costs are not as high as they might be with manual labor to measure turbidity, suggesting that the investment in developing and implementing the Turbidity Measurement system would result in savings and improved logistics in the production environment. The benefits of real-time monitoring and data collection not only justify a high LoA but also contribute to overall savings and enhanced operational efficiency.

This three-dimensional analysis visually represents the upward movement on the LoA axis for each implementation, which indicates a positive progression towards higher automation. Contrary to a traditional trade-off scenario, the associated increase in implementation difficulty and cost does not follow the conventional trend. Instead, there is a notable decrease in both implementation difficulty and cost, aligning with the improved automation capabilities. This unique dynamic underscores the efficiency gains achieved without sacrificing significant resources. It's essential to recognize that the enhanced automation capabilities bring great benefits in terms of efficiency and reliability, and the overall decrease in implementation difficulty and cost further emphasizes the success and suitability of these implementations in the production environment.

Chapter 6

Conclusions and Future Work

The aim of this research project was to investigate the transformative potential of digital twins and Cyber-Physical Systems (CPS) in the industrial context. As detailed in the Introduction chapter, digital twins have the capacity to revolutionize various industries, including textile manufacturing, by optimizing production schedules, minimizing waste, reducing energy consumption, enabling real-time equipment health monitoring, and supporting predictive maintenance to prevent unplanned downtime. The study aimed to enhance the efficiency and reliability of industrial operations through the application of CPS principles, including the adoption of a DevOps approach.

An effective system for data acquisition was successfully developed to address the systematic collection of data from PLCs and other machinery within the industrial environment. Multiple data capture methods were explored, including the utilization of Node-RED services for PLC data capture, employing Python applications for extracting critical data from various sources, tracking RFID tags, and measuring water turbidity. Leonisa provided the industrial context for this research, focusing on the vertical integration of the system to contribute to the company's future growth and success. The principles and methodologies developed hold substantial value for textile plant managers, engineers, and stakeholders interested in adopting digital twin technology.

The integration of the DevOps approach within the CPS framework is a significant achievement. DevOps principles, emphasizing collaboration, automation, and monitoring, have been effectively applied. The utilization of Ansible, an Infrastructure as Code (IaC) tool, played a crucial role in automating server provisioning, configuration management, and periodic updates. This comprehensive automation approach reduces the need for manual intervention, enhances system reliability,

and minimizes the risk of errors, ensuring uniformity and reproducibility of the system's environment. Periodic updates and maintenance tasks were streamlined through Ansible, resulting in improved system reliability.

In addition, the study highlighted the significance of monitoring and visualization within the CPS framework. The integration of Prometheus Node Exporter and Grafana allowed for comprehensive and real-time system monitoring, providing insights into system health and performance. The system was equipped with proactive alerting mechanisms, ensuring prompt responses to deviations from expected performance.

Moreover, the research takes into account the importance of incorporating well-established CPS architecture models into the implementation process. Three major reference architectures, the 5C architecture, the Reference Architecture Model Industry 4.0 (RAMI 4.0), and the Industrial Internet Reference Architecture (IIRA), have been explored, and their roles in guiding the design and implementation of Industry 4.0 systems have been examined. These reference architectures provide valuable guidelines and standards for the development of CPS, promoting the integration of various technologies such as digital twins, IIoT, and cloud computing, while addressing challenges in interoperability and security.

In the same vein, the research recognizes the significance of comprehensively categorizing industrial variables, which are fundamental to the functioning of a manufacturing plant. Industrial variables have been categorized into four key groups: Process, Transactional, Reliability-Centered Maintenance (RCM), and Auxiliary variables. These categories encompass a wide range of parameters and data types crucial for optimizing production, ensuring product quality, and maintaining operational efficiency.

The research carries significant implications as it provides a practical framework for optimizing industrial processes in manufacturing companies through CPS implementation. The principles and methodologies developed in this project have the versatility to be applied across various industrial domains, offering valuable insights into integrating DevOps practices within CPS, thereby enhancing operational efficiency and reliability.

Nonetheless, it is essential to acknowledge the inherent limitations within these systems. Challenges include adapting CPS to diverse industrial settings and the ongoing need for refinement in data acquisition techniques and cybersecurity measures. Continuous attention to these areas is paramount to ensure a comprehensive and successful CPS implementation within the company.

Moving forward, future work in this field should emphasize the expansion of CPS applications into diverse industrial contexts, with particular attention to addressing unique challenges presented by each setting. This expansion must also prioritize

the strengthening of security measures to protect the network, particularly as the system becomes more susceptible to potential cyberattacks with the increasing number of IoT devices. Furthermore, there is a need for ongoing refinement in data acquisition techniques, with an unwavering commitment to optimizing both efficiency and data accuracy. The exploration of additional DevOps practices and tools for CPS implementation can further enhance automation and streamline system maintenance.

Additionally, research into advanced analytics and machine learning algorithms for predictive maintenance and process optimization within CPS can unlock new levels in industrial performance and sustainability. The comprehensive understanding of industrial variables, as categorized in this research, should serve as a foundation for future studies aimed at fine-tuning CPS implementations across different domains.

In this vision, the system in the company will undergo significant improvements, streamlining information flow and edge device management as shown in Figure 6.1. The Cybernetics server will serve as the central hub for data acquisition, integrating more closely into the industrial environment. Key enhancements include better connectivity between the Cybernetics server and Raspberry Pi devices, optimizing their deployment for more efficient data transmission. These Raspberry Pi devices play a vital role in real-time data acquisition at the network's edge.

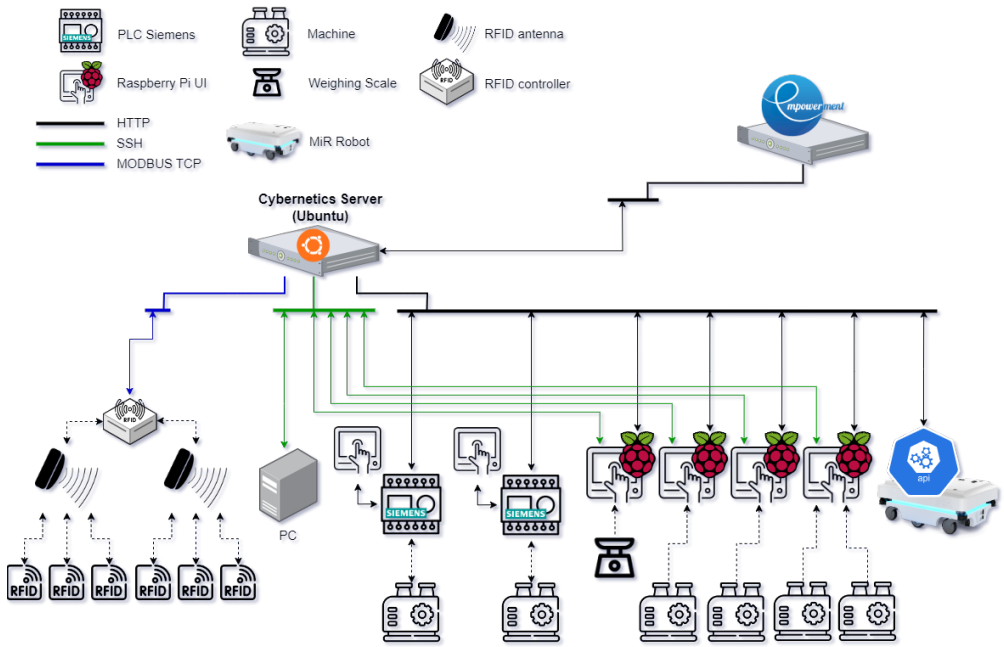

Figure 6.1: Future system structure scheme.

As industries progress towards advanced technological landscapes, the incorporation of Kubernetes emerges as a pivotal aspect in optimizing the utilization of servers, and edge computing devices. In this way, the envisioned future implementations involve leveraging Kubernetes to create one or multiple clusters that seamlessly integrates servers and edge computing devices, such as Raspberry Pis. This cluster structure integrated with the different management and monitoring software like Snipe-IT for device management, Rancher for Kubernetes coordination, and ensuring High Availability (HA) of services in the cluster, as shown in Figure 6.2 and their respective agents and monitoring information flow presented in Figure 6.3, holds tremendous potential for enhancing the efficiency, scalability, and resource utilization within the industrial environment.

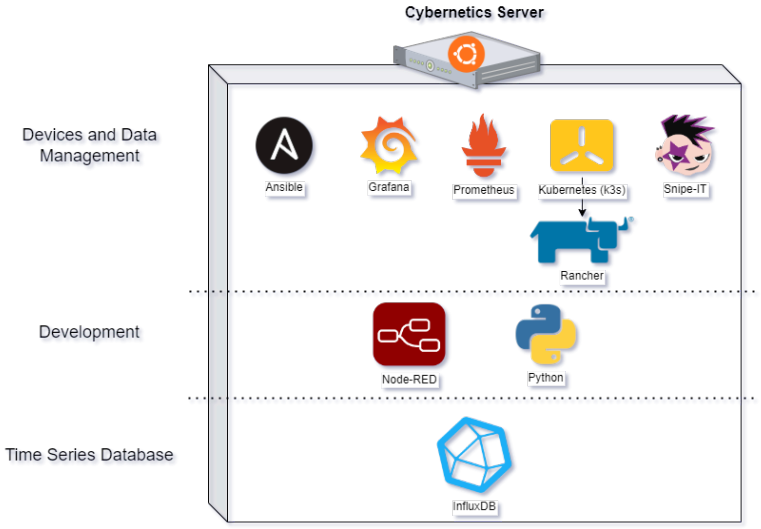

Figure 6.2: Cybernetics server software stack.

The system is also set to integrate with robots and machines like the MiR robot, enhancing automation and monitoring capabilities. Notably, DIMESH devices will be replaced by Raspberry Pis with touch screens, simplifying data input and enabling remote management, improving information quality and error resolution. This future system structure promises greater integration and efficiency, leveraging technology and DevOps practices for enhanced productivity. Specifically, as depicted in Figure 6.4 as an example, the use of a microservice structure paradigm will delineate data acquisition from various sources like weighing scales (Raspberry Pi), Datacolor spectrophotometers, process and transactional variables (using Raspberry Pis), Siemens PLC, and RFID data acquisition and many more. These microservices, deployed in both the Raspberry Pis and the Cybernetics

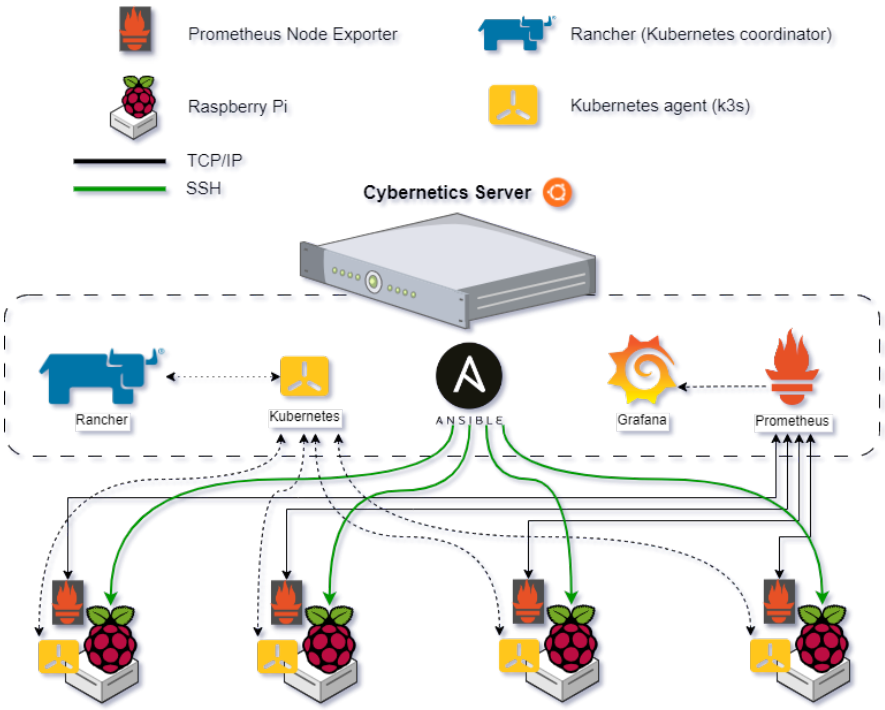

Figure 6.3: In development Cybernetics server agents and monitoring information flow.

server, collectively feed data into InfluxDB, with specialized data transmission microservices connecting it to the Empowerment server.

On this regard, the project has laid the foundation for a promising future in the integration of digital twins and CPS in the industry, specifically in Leonisa. There are many opportunities and challenges and by leveraging the power of digital twins, CPS, and emerging technologies, industries can unlock new levels of efficiency, productivity, and sustainability, supported by the robust practices of DevOps.

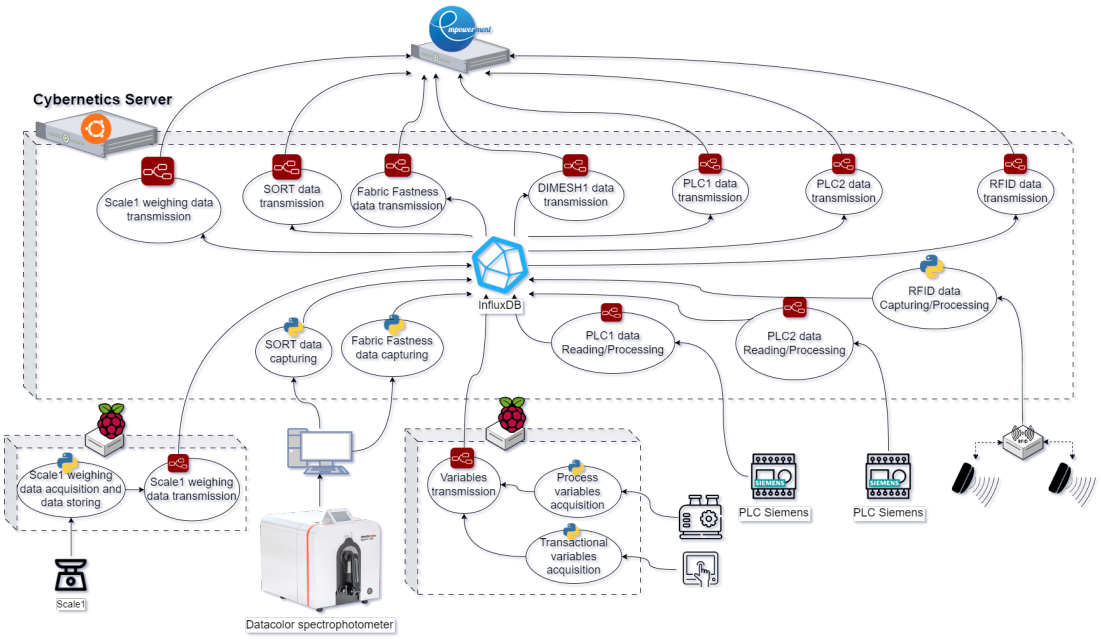

Figure 6.4: Cybernetics server future microservice structure.

Appendix A

Python Applications for Data Capture

Listing A.1: HTTPSCommunication Class in Python

```
import json
import time
import requests

class HTTPSCommunication(object):
    def __init__(self, username, password, entity_name):
        """
        Initializes the HTTPSCommunication object.

        Args:
            username (str): The username for authentication.
            password (str): The password for authentication.
            entity_name (str): The entity name for the URL.
        """
        self.username = username
        self.password = password
        self.name = entity_name
        self.Authorization = 'Basic ******************************=='
        self.url_Data = "https://empowerment.../entities/" +
    entity_name
        self.url_Token = "https://empowerment.../oauth/token"
        self.url_Revoke = "https://empowerment.../oauth/revoke?token=
    "
        self.pathToken = 'token.txt'

        self.headersT = {
            'Content-Type': 'application/x-www-form-urlencoded',
            'Accept': 'application/json',
```

```
            'Authorization': str(self.Authorization),
            'Connection': 'keep-alive',
            'User-Agent': 'otsrcibernetica',
            'Cache-Control': 'no-cache',
            'Accept-Encoding': 'gzip, deflatee'
        }

    def headersB(self, Token: str):
        """
        Creates and returns headers with the bearer token for HTTP requests.

        Args:
            Token (str): The bearer token.

        Returns:
            dict: The headers with the bearer token.
        """
        bearer = 'Bearer '
        Bearer = bearer + str(Token).strip('"')
        headersB = {
            'Authorization': Bearer,
            'Content-Type': 'text/plain',
            'User-Agent': 'otsrcibernetica',
            'Accept': '*/*',
            'Cache-Control': 'no-cache',
            'Accept-Encoding': 'gzip, deflatee',
            'Connection': 'keep-alive',
        }
        return headersB

    def RequestToken(self, path: str) -> str:
        """
        Requests and retrieves the access token for authentication.

        Args:
            path (str): The path for saving the token.

        Returns:
            str: The access token.
        """
        payload = "grant_type=password&username=" + \
            str(self.username) + "&password=" + str(self.password)
        response = requests.post(
            self.url_Token, headers=self.headersT, data=payload)
        data = json.loads(response.text)
        Token = data['access_token']
        with open(path + self.pathToken, 'w') as Token_txt:
            Token_txt.write(str(Token))
```

```
75          time.sleep(0.1)
76          return Token
77
78      def SendData(self, Token: str, Data: dict):
79          """
80          Sends data to the specified URL.
81
82          Args:
83              Token (str): The access token.
84              Data (dict): The data to be sent.
85
86          Returns:
87              requests.Response: The response of the HTTP request.
88          """
89          payload = json.dumps(Data)
90          response = requests.post(
91              self.url_Data, headers=self.headersB(Token), data=payload
    , timeout=15)
92          return response
93
94      def ReadData(self, Token: str) -> list[dict]:
95          """
96          Reads data from the specified URL.
97
98          Args:
99              Token (str): The access token.
100
101         Returns:
102             list[dict]: The list of dictionaries containing the data.
103         """
104         response = requests.get(
105             self.url_Data, headers=self.headersB(Token))
106         return json.loads(response.text)
107
108     def ModifyData(self, Token: str, Data: dict):
109         """
110         Modifies data at the specified URL.
111
112         Args:
113             Token (str): The access token.
114             Data (dict): The data to be modified.
115
116         Returns:
117             requests.Response: The response of the HTTP request.
118         """
119         payload = json.dumps(Data)
120         response = requests.put(
121             self.url_Data, headers=self.headersB(Token), data=payload
    )
```

```
122         return response
123
124     def CloseSession(self, path: str) -> bool:
125         """
126         Closes the session and deletes the token file.
127
128         Args:
129             path (str): The path to the token file.
130
131         Returns:
132             bool: True if the session is closed successfully, else
    False.
133         """
134         with open(path + self.pathToken, 'r') as Token_txt:
135             Token = Token_txt.read()
136         time.sleep(0.1)
137         url_revoke_Token = self.url_Revoke + str(Token).strip('"')
138         response = requests.post(url_revoke_Token, headers=self.
    headersT)
139         if '200' in str(response) or '201' in str(response) or '400'
    in str(response) or '401' in str(response):
140             with open(path + self.pathToken, 'w') as Token_txt:
141                 Token_txt.write(str(''))
142             time.sleep(0.1)
143             return True
144         else:
145             return False
146
147 if __name__ == "__main__":
148     """
149     Example of modifying a variable in "emp_EmpAcaSort" database
    endpoint.
150     """
151     username = "username"
152     password = "password"
153     emp_endpoint = "emp_EmpAcaSort"
154
155     emp_Estados = HTTPSCommunication(
156         username=username, password=password, entity_name=
    emp_endpoint)
157
158     Token = emp_Estados.RequestToken()
159
160     data = {
161         "code": '03'
162     }
163     # emp_Estados.ModifyData(Token, data)
164     print(emp_Estados.ReadData(Token))
165
```

```
166     CloseSession = emp_Estados.CloseSession(Token)
```

Listing A.2: Modbus TCP Communication Code in Python to the RFID Gateway

```
1  import time
2  from pymodbus.client import ModbusTcpClient
3  import sys
4
5  # Define the Modbus TCP server details
6  SERVER_IP = '172.*.*.*'  # Pepperl+Fuchs gateway antenna controller
7  SERVER_PORT = 502
8
9  # Create a Modbus TCP client instance
10 client = ModbusTcpClient(SERVER_IP, port=SERVER_PORT)
11
12 # Function to extract tag information from Modbus registers
13 def extract_tag_information(registers):
14     try:
15         # Extract UII and TID information from the registers
16         uii_length = registers[4]
17         tid_length = registers[5 + int(uii_length/2)]
18
19         uii_section = registers[5:5 + int(uii_length/2)]
20         tid_section = registers[6 + int(uii_length/2):6 + int((
       uii_length + tid_length)/2)]
21
22         formatted_uii = ''.join(format(byte, '04X') for byte in
       uii_section)
23         formatted_tid = ''.join(format(byte, '04X') for byte in
       tid_section)
24
25         # Create a formatted tag information string
26         tag_info = f"UII: {' '.join(formatted_uii[i:i+2] for i in
       range(0, len(formatted_uii), 2))} TID: {' '.join(formatted_tid[i:i
       +2] for i in range(0, len(formatted_tid), 2))}"
27         return tag_info
28     except:
29         pass
30
31 try:
32     # Connect to the Modbus TCP server
33     if client.connect():
34         known_tags = []  # List to store known tags
35         while True:
36             # Write specific values to Modbus registers
37             combined_value_0 = (0x00 << 8) | 0x00
38             combined_value_1 = (0x00 << 8) | 0x06
39             combined_value_2 = (0x04 << 8) | 0x08
```

```
40              combined_value_3 = (0x39 << 8) | 0x39  # ASCII '9' and
    '9'
41              values = [combined_value_0, combined_value_1,
42                        combined_value_2, combined_value_3]
43
44              register_address = 3000
45
46              response = client.write_registers(
47                  register_address, values, slave=1)
48
49              if response.isError():
50                  print("Modbus Error:", response)
51
52              combined_value_0 = (0x00 << 8) | 0x00
53              combined_value_1 = (0x00 << 8) | 0x06
54              combined_value_2 = (0x01 << 8) | 0x08  # Command to read
    all tags once
55              combined_value_3 = (0x00 << 8) | 0x00
56              values = [combined_value_0, combined_value_1,
57                        combined_value_2, combined_value_3]
58
59              response = client.write_registers(
60                  register_address, values, slave=1)
61
62              if response.isError():
63                  print("Modbus Error (2):", response)
64
65              # Continuously read data from Modbus registers
66              previous_registers = []
67              tags_i = 0
68              read_tags = []
69
70              current_registers = [1]
71              tags = 0
72              while any(current_registers):
73                  read_response = client.read_holding_registers(
74                      register_address, count=19, slave=1)
75                  if read_response.isError():
76                      print("Modbus Read Error:")
77                  else:
78                      current_registers = read_response.registers
79
80                      if current_registers[4] != 0:
81                          # Extract tag information
82                          tag_info = extract_tag_information(
    current_registers)
83
84                          if (tag_info is not None) and (tag_info not
    in known_tags):
```

```
                                known_tags.append(tag_info)
                                print(tag_info)
                                tags += 1
                            previous_registers = current_registers

            print(f"{tags} tags leídos")  # Print the number of tags read
            print(f"{len(known_tags)} tags conocidos")  # Print the number of known tags

            time.sleep(0.1)  # Sleep for a short interval

    else:
        print("Connection failed.")
finally:
    client.close()

"""
Example of the results:
UII: 30 00 E2 80 11 91 A5 02 00 60 1E 03 AA 4A TID: E2 80 11 91 20 00 4A 4A F0 1D 03 00
UII: 34 00 30 14 F7 33 7C 00 1F 00 00 00 81 BA TID: E2 80 11 05 20 00 7B 0C A9 10 09 3B
UII: 34 00 30 14 F7 33 7C 00 1F 00 00 00 81 B9 TID: E2 80 11 05 20 00 7A CC A9 10 09 3B
UII: 0C 00 00 02 TID: E2 80 11 05 20 00 73 8C A9 12 09 3B
UII: 0C 00 00 03 TID: E2 80 11 05 20 00 72 CC A9 12 09 3B
UII: 34 00 30 14 F7 33 7C 00 1F 00 00 00 81 B7 TID: E2 80 11 05 20 00 7A 4C A9 10 09 3B
UII: 30 00 E2 80 11 91 A5 02 00 60 1E 04 CE AF TID: E2 80 11 91 20 00 4E AF F0 26 03 00
UII: 30 00 E2 80 11 91 A5 02 00 60 1E 11 58 D2 TID: E2 80 11 91 20 00 58 D2 F0 8A 03 00
UII: 30 00 E2 80 11 91 A5 02 00 60 1E 11 57 BD TID: E2 80 11 91 20 00 57 BD F0 8A 03 00
UII: 30 00 E2 80 11 91 A5 02 00 60 1E 11 20 83 TID: E2 80 11 91 20 00 40 83 F0 89 03 00
UII: 34 00 30 14 F7 33 7C 00 1F 00 00 00 81 B6 TID: E2 80 11 05 20 00 79 8C A9 10 09 3B
UII: 34 00 30 14 F7 33 7C 00 1F 00 00 00 81 B8 TID: E2 80 11 05 20 00 7A 8C A9 10 09 3B
12 read tags
12 known tags
"""
```

Listing A.3: Turbidity monitoring Python code based on OpenCV

```
# Import necessary libraries
from Paquetes.MODBUS.Comunicacion_MODBUS_TCP import *
```

```
import cv2
from picamera.array import PiRGBArray
from picamera import PiCamera, Color
import numpy as np
import warnings
import os
import time
import datetime
from os import path
import shutil
warnings.filterwarnings("ignore")

# Initialize Modbus communication with the PLC
PLC = Comunicacion_MODBUS_TCP(IP_Equipo="172.*.*.*", reInicial=1,
    reFinal=8)
reg_PLC = 1

# Remove a file if it exists
if path.exists("/home/pi/Desktop/WCC/frame1.png"):
     os.remove("/home/pi/Desktop/WCC/frame1.png")

# Initialize the PiCamera and set its properties
camera = PiCamera()
camera.resolution = (640, 480)
rawCapture = PiRGBArray(camera, size=(640, 480))

time.sleep(5)

# Set camera exposure mode and other settings
camera.exposure_mode = 'auto'

# Initialize variables for image processing
frame1 = ""
cf1 = 0
minuto = ""
turbidez_ant = 0
f_espera = 1

# Capture frames from the camera
for frameim in camera.capture_continuous(rawCapture, format="bgr",
    use_video_port=True):
     frame = frameim.array
     cv2.imshow("Frame", frame)

     if cf1 > f_espera:
         # Calculate image difference
          diff = cv2.absdiff(frame1, frame)
          cv2.imshow('Diferencia', diff)

```

```
50          # Calculate turbidity based on the image difference
51          p_diff_v = np.sum(diff)
52          max = 921600 * 0.25 * 255 * 217.77 / 200
53          turbidez = p_diff_v * 200 / max
54
55          if cf1 % 20 == 0:
56              print(turbidez)
57
58          # Send turbidity data to a PLC
59          PLC.respuestaHTTPS(reg_PLC, int(round(turbidez * 100)))
60
61          now = datetime.datetime.now()
62          today = datetime.date.today()
63          last_month = today - datetime.timedelta(days=30)
64          muestras = [0, 10, 20, 30, 40, 50]
65
66          if (now.minute in muestras) and now.minute != minuto:
67              try:
68                  os.mkdir("/home/pi/Desktop/WCC/Log_images/%d-%d-%d" %
69                           (now.year, now.month, now.day))
70              except:
71                  pass
72
73              cv2.imwrite("/home/pi/Desktop/WCC/Log_images/%d-%d-%d/
    foto-%d-%d-%d_%d:%d.png" %
74                          (now.year, now.month, now.day, now.year, now.
    month, now.day, now.hour, now.minute), frame)
75              print("Foto foto-%d-%d-%d_%d:%d.png" %
76                    (now.year, now.month, now.day, now.hour, now.minute
    ))
77
78          minuto = now.minute
79
80          try:
81              # Remove images from the previous month
82              shutil.rmtree("/home/pi/Desktop/WCC/Log_images/%d-%d-%d"
    %
83                            (last_month.year, last_month.month,
    last_month.day))
84          except:
85              pass
86
87      if cf1 == f_espera:
88          if path.exists("/home/pi/Desktop/WCC/frame1.png"):
89              frame1 = cv2.imread("/home/pi/Desktop/WCC/frame1.png")
90          else:
91              frame1 = frame
92              cv2.imwrite("/home/pi/Desktop/WCC/frame1.png", frame1)
93          frame1 = cv2.imread("/home/pi/Desktop/WCC/frame1_flujo.png")
```

```
94          print('Funcionando')
95      elif cf1 == 0:
96          print('Iniciando')
97
98      cf1 += 1
99
100     key = cv2.waitKey(1) & 0xFF
101     rawCapture.truncate(0)
102
103     if key == ord("q"):
104         break
```

Appendix B

Ansible INI and YAML Codes

Listing B.1: Devices Ansible inventory INI file sample

```
[otsrvcibernetica]
172.24.43.9

[rpi_wlan_OT]
172.24.15.15 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD

[rpi_ssh_ok]
172.19.2.197 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
172.19.2.202 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
172.19.2.204 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
172.24.13.231 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
...

[rpi_manuf]
172.19.2.197 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
172.19.2.202 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
172.19.2.203 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
...

[rpi_DT]
172.24.13.25 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
172.24.13.82 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
172.24.13.85 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
172.24.13.218 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
172.24.13.219 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
172.24.13.220 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
172.24.13.221 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
...
```

```
29
30 [rpi_CD]
31 172.25.10.5 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
32 172.25.10.7 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
33 172.25.10.8 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
34 172.25.10.9 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
35 ...
36
37 [rpi_confec]
38 172.29.1.119 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
39 172.29.1.120 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
40 172.29.1.195 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
41 172.29.1.196 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
42 172.29.1.197 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
43 172.29.1.198 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
44 172.29.1.199 ansible_ssh_user=pi ansible_ssh_pass=PASSWORD
45 ...
```

Listing B.2: SSH connection to all hosts

```
1 ---
2 - name: SSH to All Hosts
3   hosts: all,!otsrvcibernetica  # Targets all hosts except the
      Cybernetics server
4   gather_facts: no  # Disable gathering of host facts
5
6   tasks:
7     - name: SSH to Remote Hosts
8       ansible.builtin.shell: sudo sshpass -p "{{ ansible_ssh_pass }}"
       ssh -o StrictHostKeyChecking=no -i /home/jbenjumeam/.ssh/id_rsa
      {{ ansible_ssh_user }}@{{ inventory_hostname }}  # Use the 'shell'
       module to run a shell command
9       delegate_to: localhost  # Delegate the task to run locally on
      the control machine
```

Listing B.3: Copy SSH key to all hosts

```
1 ---
2 - name: Copy SSH keys to hosts
3   hosts: all,!otsrvcibernetica  # Targets all hosts except the
      Cybernetics server
4   become: yes  # Elevate privileges to perform tasks as a superuser
5
6   tasks:
7     - name: Copy SSH key  # Task name or description
8       authorized_key:  # Use the 'authorized_key' module to manage
      SSH keys
9         user: "{{ ansible_ssh_user }}"  # The remote user for whom to
       copy the key
```

```
10          key: "{{ lookup('file', '/home/jbenjumeam/.ssh/id_rsa.pub')
    }}"  # The SSH key to be copied
11        ignore_errors: yes  # Ignore errors if the key is already
    copied
```

Listing B.4: Update and upgrade execution

```
1  ---
2  - name: Update and Upgrade Packages
3    hosts: all,!otsrvcibernetica  # Targets all hosts except the
       Cybernetics server
4    become: yes  # Run tasks with elevated privileges (sudo)
5    environment:
6      DEBIAN_FRONTEND: noninteractive  # Set the environment variable
       to noninteractive
7
8    tasks:
9      - name: Display Current Date
10       debug:  # Use the 'debug' module to print messages
11         msg: "Time: {{ ansible_date_time.date }} {{ ansible_date_time
       .time }}"  # Display the current date and time
12
13     - name: Update package lists  # Task to update the package lists
14       ansible.builtin.apt:  # Use the 'apt' module for package
       management
15         update_cache: yes  # Update the package cache
16       register: update_result  # Store the result in the '
       update_result' variable
17
18     - name: Execute 'sudo apt update -y' if update fails  # Task to
       execute 'apt update' if the previous update failed
19       ansible.builtin.shell: "sudo apt update -y"  # Run the 'apt
       update' command with '-y' for automatic confirmation
20       when: update_result.failed  # Condition: Execute only if the '
       update_result' task failed
21
22     - name: Upgrade all packages  # Task to upgrade all packages
23       ansible.builtin.apt:
24         upgrade: yes  # Upgrade packages
25       register: upgrade_result  # Store the result in the '
       upgrade_result' variable
26
27     - name: Execute 'sudo apt upgrade -y' if upgrade fails  # Task to
        execute 'apt upgrade' if the previous upgrade failed
28       ansible.builtin.shell: "sudo apt upgrade -y"  # Run the 'apt
       upgrade' command with '-y' for automatic confirmation
29       when: upgrade_result.failed  # Condition: Execute only if the '
       upgrade_result' task failed
30
```

```
31      - name: Autoremove  # Task to perform package autoremoval
32        ansible.builtin.apt:
33          autoremove: yes  # Remove packages that are no longer needed
34        register: autoremove_result  # Store the result in the '
    autoremove_result' variable
35
36      - name: Execute 'sudo apt autoremove -y' if autoremove fails  #
    Task to execute 'apt autoremove' if the autoremove task failed
37        ansible.builtin.shell: "sudo apt autoremove -y"  # Run the 'apt
     autoremove' command with '-y' for automatic confirmation
38        when: autoremove_result.failed  # Condition: Execute only if
    the 'autoremove_result' task failed
```

Listing B.5: Create Prometheus User and Group Ansible YAML File to run Prometheus services

```
1  ---
2  - name: Create Prometheus User and Group
3    hosts: all,!otsrvcibernetica  # Targets all hosts except the
       Cybernetics server
4    become: yes  # Run tasks with elevated privileges (sudo)
5
6    tasks:
7      - name: Check if prometheus group exists
8        ansible.builtin.group:
9          name: prometheus
10       register: prometheus_group_check
11       ignore_errors: yes  # Ignore errors if the group doesn't exist
12
13     - name: Create prometheus group if it doesn't exist
14       ansible.builtin.group:
15         name: prometheus
16         state: present
17       when: prometheus_group_check.failed  # Create group only if it
    doesn't exist
18
19     - name: Check if prometheus user exists
20       ansible.builtin.user:
21         name: prometheus
22       register: prometheus_user_check
23       ignore_errors: yes  # Ignore errors if the user doesn't exist
24
25     - name: Create prometheus user if it doesn't exist
26       ansible.builtin.user:
27         name: prometheus
28         group: prometheus
29         shell: /bin/bash
30         home: /home/prometheus
31         createhome: yes
```

```
32          state: present
33        when: prometheus_user_check.failed  # Create user only if it
      doesn't exist
34
35      - name: Ensure prometheus user is part of the prometheus group
36        ansible.builtin.user:
37          name: prometheus
38          groups: prometheus
39          append: yes
```

Listing B.6: Install Prometheus Node Exporter service

```
1 ---
2 - name: Install Prometheus Node Exporter
3   hosts: all
4   become: yes  # Run tasks with elevated privileges (sudo)
5
6   tasks:
7     - name: Check if Node Exporter is installed
8       ansible.builtin.stat:
9         path: /usr/local/bin/node_exporter
10      register: node_exporter_installed  # Register the result in the
      'node_exporter_installed' variable
11
12    - name: Download Node Exporter if not installed
13      ansible.builtin.get_url:
14        url: https://github.com/prometheus/node_exporter/releases/
      download/v1.6.1/node_exporter-1.6.1.linux-armv7.tar.gz  # Update
      URL for ARM architecture if needed
15        dest: /tmp/node_exporter.tar.gz
16      register: download_node_exporter  # Register the download
      result
17      when: not node_exporter_installed.stat.exists  # Execute only
      if Node Exporter is not installed
18      ignore_errors: true  # Ignore errors during download
19
20    - name: Check if Node Exporter is downloaded
21      ansible.builtin.stat:
22        path: /tmp/node_exporter.tar.gz
23      register: node_exporter_download  # Register the download check
       result
24
25    - name: Copy local tar.gz file if download fails
26      ansible.builtin.copy:
27        src: /etc/prometheus/templates/node_exporter-1.6.1.linux-
      armv7.tar.gz
28        dest: /tmp/node_exporter.tar.gz
29      when: not node_exporter_download.stat.exists  # Copy if
      download fails
```

```
30
31      - name: Extract Node Exporter
32        ansible.builtin.unarchive:
33          src: /tmp/node_exporter.tar.gz
34          dest: /tmp
35          remote_src: yes
36        when: not node_exporter_installed.stat.exists  # Extract if
    Node Exporter is not installed
37
38      - name: Rename extracted folder to "node_exporter"
39        ansible.builtin.shell: mv /tmp/node_exporter-*/node_exporter /
    usr/local/bin/node_exporter
40        when: not node_exporter_installed.stat.exists  # Rename if Node
     Exporter is not installed
41
42      - name: Set permissions for Node Exporter
43        ansible.builtin.file:
44          path: /usr/local/bin/node_exporter
45          owner: root
46          group: root
47          mode: 0755
48        when: not node_exporter_installed.stat.exists  # Set
    permissions if Node Exporter is not installed
49
50      - name: Check if service is installed
51        ansible.builtin.stat:
52          path: /etc/systemd/system/node_exporter.service
53        register: node_exporter_service_installed  # Register the
    service check result
54
55      - name: Install unit file to systemd
56        template:
57          src: /etc/prometheus/templates/servicios/node_exporter.
    service.j2
58          dest: /etc/systemd/system/node_exporter.service
59          owner: root
60          group: root
61          mode: 0600
62        when: not node_exporter_service_installed.stat.exists  #
    Install the unit file if service is not installed
63
64      - name: Configure Node Exporter systemd service
65        systemd:
66          name: node_exporter.service
67          enabled: yes
68          state: started
69          daemon_reload: yes
```

Appendix C

Turbidity Monitoring Implementation Tests

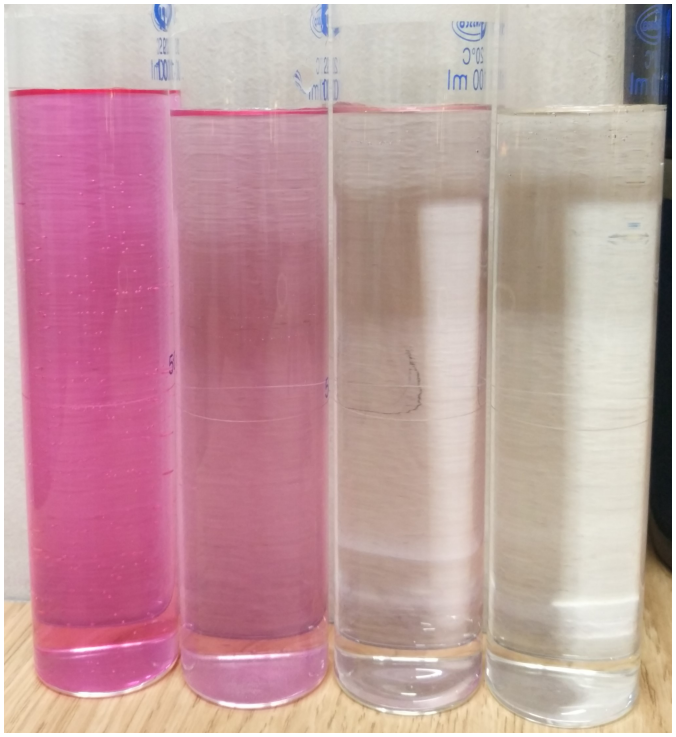

Figure C.1: Water turbidity tests.

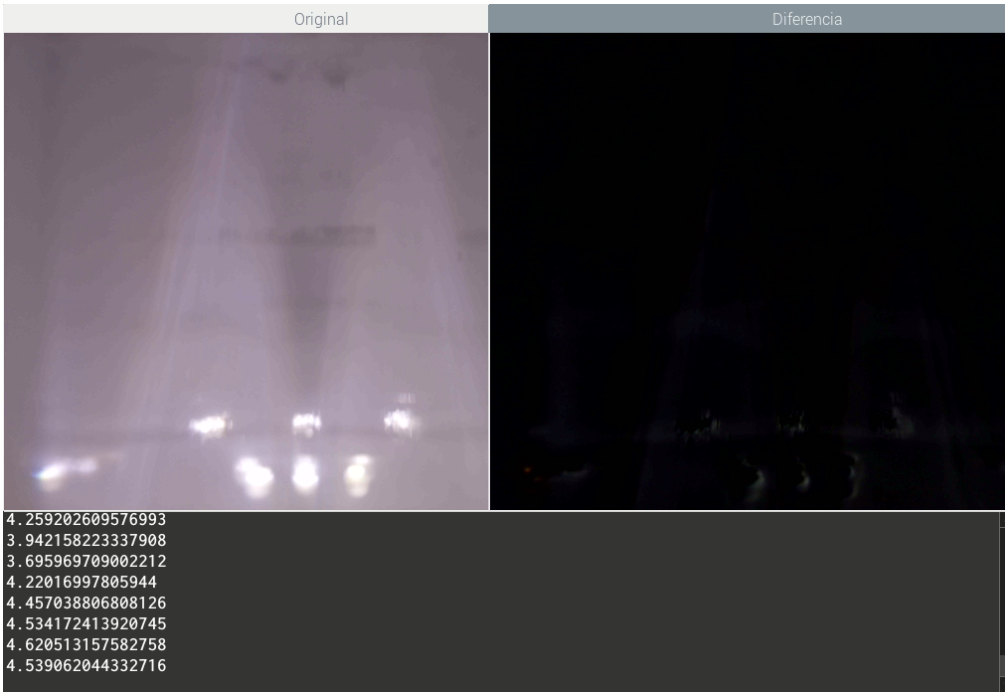

Figure C.2: Clean water turbidity test.

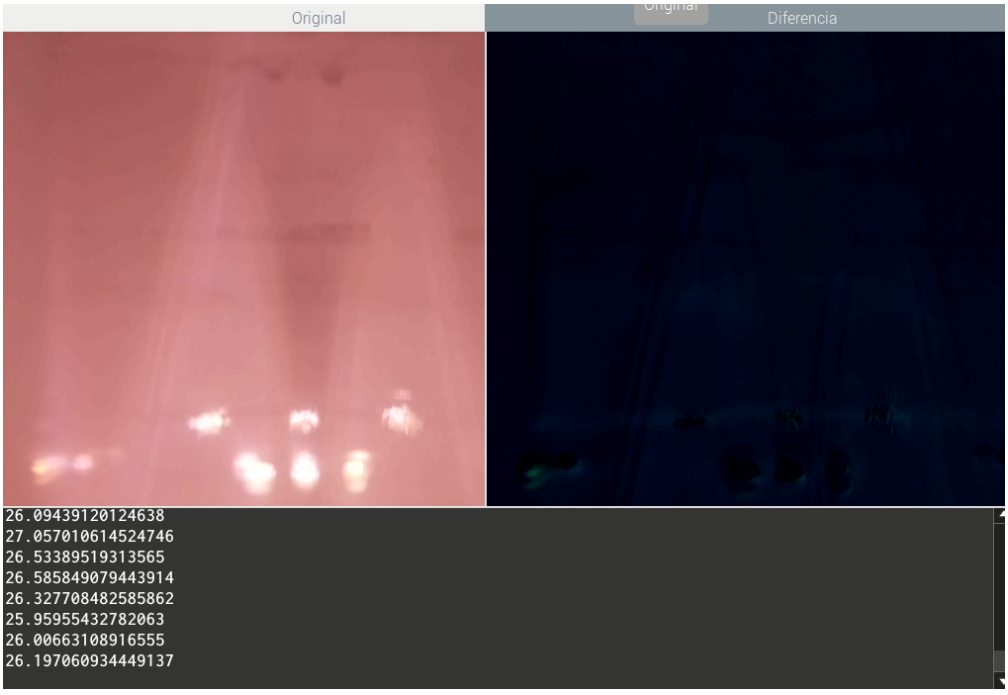

Figure C.3: First level water turbidity test.

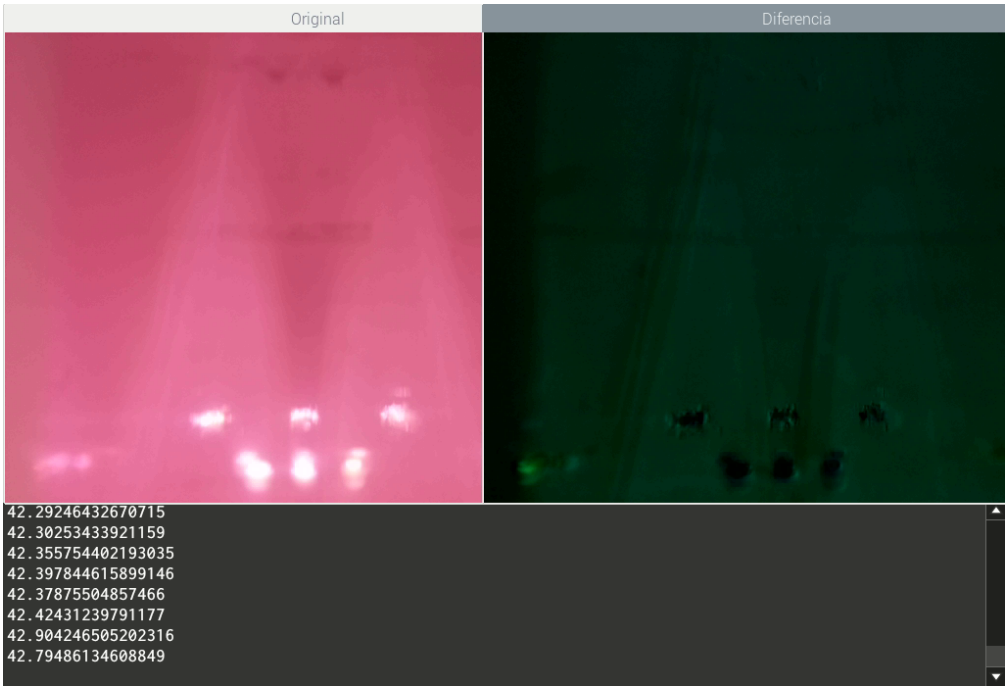

Figure C.4: Second level water turbidity test.

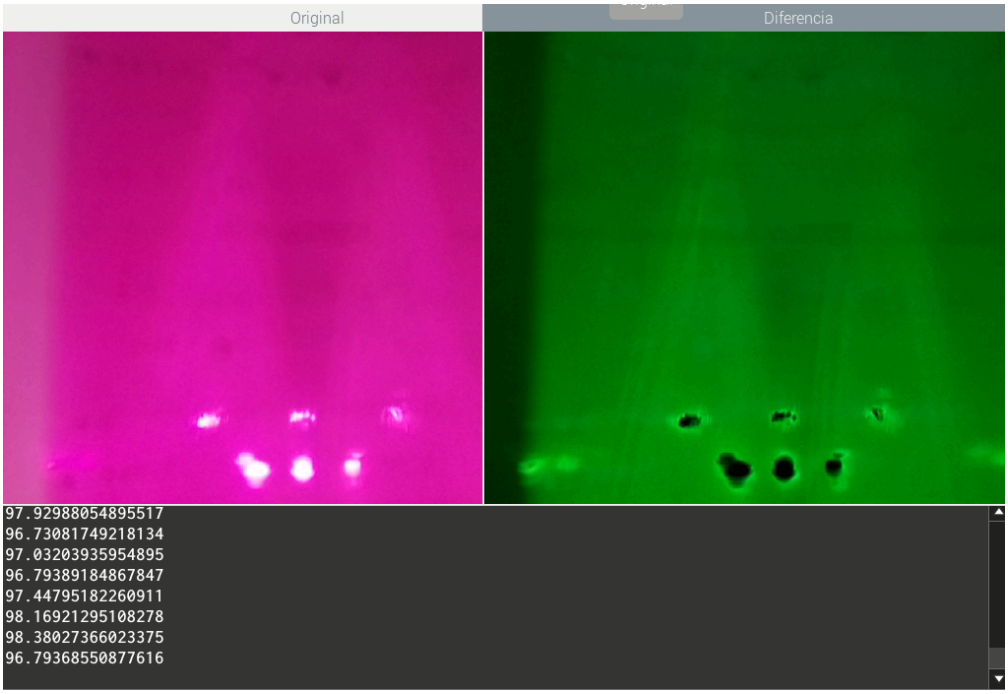

Figure C.5: Third level water turbidity test.

Appendix D

Grafana Dashboards Samples

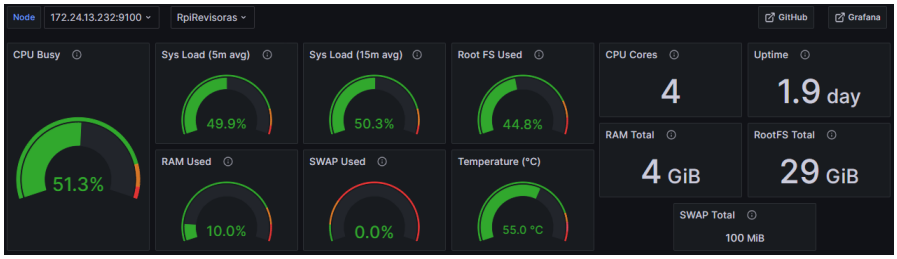

Figure D.1: Main Grafana dashboard gauges of the health information of one device.

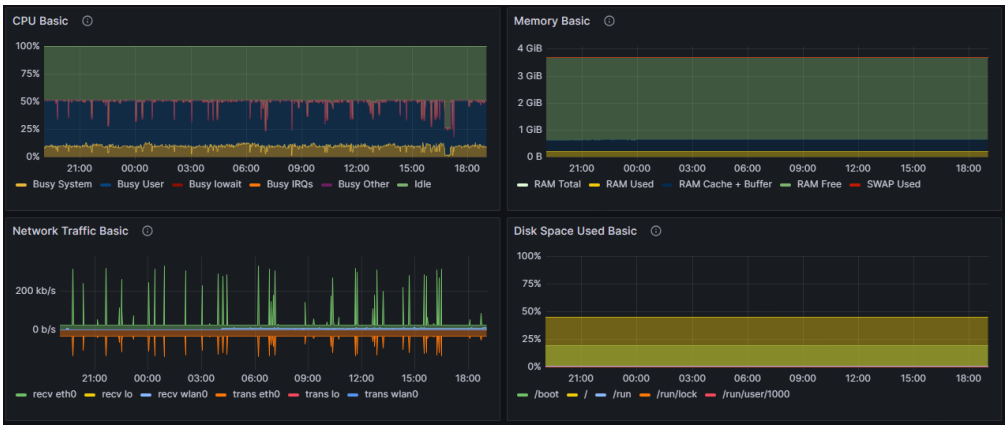

Figure D.2: Main Grafana dashboard CPU, Memory and Network information.

Bibliography

[1] Aidan Fuller, Zhong Fan, Charles Day, and Chris Barlow. «Digital Twin: Enabling Technologies, Challenges and Open Research». In: *IEEE Access* 8 (2020), pp. 108952–108971. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2998358. URL: https://ieeexplore.ieee.org/document/9103025/ (visited on 06/20/2023) (cit. on pp. 1, 2).

[2] Adil Rasheed, Omer San, and Trond Kvamsdal. «Digital Twin: Values, Challenges and Enablers From a Modeling Perspective». In: *IEEE Access* 8 (2020), pp. 21980–22012. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2970143. URL: https://ieeexplore.ieee.org/document/8972429/ (visited on 06/20/2023) (cit. on p. 1).

[3] Joseph Evans Agolla. «Human Capital in the Smart Manufacturing and Industry 4.0 Revolution». en. In: *Digital Transformation in Smart Manufacturing*. Ed. by Antonella Petrillo, Raffaele Cioffi, and Fabio De Felice. InTech, Feb. 2018. ISBN: 978-953-51-3841-9 978-953-51-3842-6. DOI: 10.5772/intechopen.73575. URL: http://www.intechopen.com/books/digital-transformation-in-smart-manufacturing/human-capital-in-the-smart-manufacturing-and-industry-4-0-revolution (visited on 09/05/2023) (cit. on pp. 1, 2).

[4] Wesam S Alaloul, Mohd Shahir Liew, Noor Amila Wan Abdullah Zawawi, and Bashar S Mohammed. «Industry Revolution IR 4.0: Future Opportunities and Challenges in Construction Industry». In: *MATEC Web of Conferences* 203 (2018). Ed. by M. Raza Ul Mustafa, I. Bin Othman, M. Latheef, D. Bayu Endrayana, and N. Zulaikha Bt Yusof, p. 02010. ISSN: 2261-236X. DOI: 10.1051/matecconf/201820302010. URL: https://www.matec-conferences.org/10.1051/matecconf/201820302010 (visited on 09/05/2023) (cit. on p. 2).

[5] Fei Tao, Fangyuan Sui, Ang Liu, Qinglin Qi, Meng Zhang, Boyang Song, Zirong Guo, Stephen C.-Y. Lu, and A. Y. C. Nee. «Digital twin-driven product design framework». en. In: *International Journal of Production Research* 57.12 (June 2018), pp. 3935–3953. ISSN: 0020-7543, 1366-588X. DOI:

10.1080/00207543.2018.1443229. URL: https://www.tandfonline.com/doi/full/10.1080/00207543.2018.1443229 (visited on 06/20/2023) (cit. on p. 2).

[6] Alperen Bal, Hilal Gevrek, and Sedefnur DemiR. «Kitlesel İmalat Sistemlerinde Dijital İkiz Kullanılarak Gerçek Zamanlı Üretim Çizelgeleme ve Tekstil Sektöründe Bir Uygulama». In: *International Journal of Advances in Engineering and Pure Sciences* 34.2 (June 2022), pp. 328–336. ISSN: 2636-8277. DOI: 10.7240/jeps.1068970. URL: http://dergipark.org.tr/tr/doi/10.7240/jeps.1068970 (visited on 06/20/2023) (cit. on p. 2).

[7] Kaznah Alshammari, Thomas Beach, and Yacine Rezgui. «Cybersecurity for digital twins in the built environment: current research and future directions». en. In: *Journal of Information Technology in Construction* 26 (Apr. 2021), pp. 159–173. ISSN: 1874-4753. DOI: 10.36680/j.itcon.2021.010. URL: https://www.itcon.org/paper/2021/10 (visited on 06/20/2023) (cit. on p. 2).

[8] S. Schweigert-Recksiek, J. Trauer, C. Engel, K. Spreitzer, and M. Zimmermann. «CONCEPTION OF A DIGITAL TWIN IN MECHANICAL ENGINEERING – A CASE STUDY IN TECHNICAL PRODUCT DEVELOPMENT». en. In: *Proceedings of the Design Society: DESIGN Conference* 1 (May 2020), pp. 383–392. ISSN: 2633-7762. DOI: 10.1017/dsd.2020.23. URL: https://www.cambridge.org/core/product/identifier/S2633776220000230/type/journal_article (visited on 06/20/2023) (cit. on p. 2).

[9] Jörgen Frohm. *Levels of automation in production systems*. eng. Doktorsavhandlingar vid Chalmers Tekniska Högskola N.S., 2736. Göteborg: Chalmers Univ. of Technology, 2008. ISBN: 978-91-7385-055-1 (cit. on p. 3).

[10] R. Parasuraman, T.B. Sheridan, and C.D. Wickens. «A model for types and levels of human interaction with automation». In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 30.3 (May 2000), pp. 286–297. ISSN: 10834427. DOI: 10.1109/3468.844354. URL: http://ieeexplore.ieee.org/document/844354/ (visited on 02/16/2024) (cit. on p. 4).

[11] Åsa Fasth, Åsa Fasth, Jörgen Frohm, Jörgen Frohm, Johan Stahre, and Johan Stahre. «RELATIONS BETWEEN PARAMETERS/PERFORMERS AND LEVELS OF AUTOMATION». In: *null* (2007). DOI: null (cit. on pp. 4, 5).

[12] Tava Lennon Olsen and Brian Tomlin. «Industry 4.0: Opportunities and Challenges for Operations Management». en. In: *Manufacturing & Service Operations Management* 22.1 (Jan. 2020), pp. 113–122. ISSN: 1523-4614, 1526-5498. DOI: 10.1287/msom.2019.0796. URL: https://pubsonline.informs.

org/doi/10.1287/msom.2019.0796 (visited on 06/16/2023) (cit. on pp. 7, 8).

[13] Kai Ding, Felix T.S. Chan, Xudong Zhang, Guanghui Zhou, and Fuqiang Zhang. «Defining a Digital Twin-based Cyber-Physical Production System for autonomous manufacturing in smart shop floors». en. In: *International Journal of Production Research* 57.20 (Oct. 2019), pp. 6315–6334. ISSN: 0020-7543, 1366-588X. DOI: 10.1080/00207543.2019.1566661. URL: https://www.tandfonline.com/doi/full/10.1080/00207543.2019.1566661 (visited on 06/16/2023) (cit. on pp. 7, 10).

[14] Jay Lee, Jay Lee, J. A. Lee, Behrad Bagheri, and Hung-An Kao. «A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems». In: *Manufacturing letters* 3 (Jan. 2015). MAG ID: 2029608738, pp. 18–23. DOI: 10.1016/j.mfglet.2014.12.001 (cit. on p. 7).

[15] Sameer Mittal, Muztoba Ahmad Khan, David Romero, and Thorsten Wuest. «Smart manufacturing: Characteristics, technologies and enabling factors». en. In: *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 233.5 (Apr. 2019), pp. 1342–1361. ISSN: 0954-4054, 2041-2975. DOI: 10.1177/0954405417736547. URL: http://journals.sagepub.com/doi/10.1177/0954405417736547 (visited on 06/16/2023) (cit. on pp. 7, 10).

[16] Moutaz Haddara and Ahmed Elragal. «The Readiness of ERP Systems for the Factory of the Future». en. In: *Procedia Computer Science* 64 (2015), pp. 721–728. ISSN: 18770509. DOI: 10.1016/j.procs.2015.08.598. URL: https://linkinghub.elsevier.com/retrieve/pii/S1877050915027337 (visited on 06/16/2023) (cit. on p. 7).

[17] Jay Lee, Jay Lee, and J. A. Lee. «Smart Factory Systems». In: *Informatik Spektrum* 38.3 (May 2015). MAG ID: 653982549, pp. 230–235. DOI: 10.1007/s00287-015-0891-z (cit. on p. 8).

[18] Davy Preuveneers and Elisabeth Ilie-Zudor. «The intelligent industry of the future: A survey on emerging trends, research challenges and opportunities in Industry 4.0». In: *Journal of Ambient Intelligence and Smart Environments* 9.3 (Jan. 2017). MAG ID: 2605859962, pp. 287–298. DOI: 10.3233/ais-170432 (cit. on pp. 8–10).

[19] Jay Lee, Moslem Azamfar, Jaskaran Singh, and Shahin Siahpour. «Integration of digital twin and deep learning in cyber-physical systems: towards smart manufacturing». en. In: *IET Collaborative Intelligent Manufacturing* 2.1 (Mar. 2020), pp. 34–36. ISSN: 2516-8398, 2516-8398. DOI: 10.1049/iet-cim.2020.0009. URL: https://onlinelibrary.wiley.com/doi/10.1049/iet-cim.2020.0009 (visited on 06/16/2023) (cit. on pp. 8, 9, 12, 13).

[20] Xiaokang Zhou, Xuesong Xu, Wei Liang, Zhi Zeng, Shohei Shimizu, Laurence T. Yang, and Qun Jin. «Intelligent Small Object Detection for Digital Twin in Smart Manufacturing With Industrial Cyber-Physical Systems». In: *IEEE Transactions on Industrial Informatics* 18.2 (Feb. 2022), pp. 1377–1386. ISSN: 1551-3203, 1941-0050. DOI: 10.1109/TII.2021.3061419. URL: https://ieeexplore.ieee.org/document/9363597/ (visited on 06/16/2023) (cit. on p. 8).

[21] Latif U. Khan, Ibrar Yaqoob, Nguyen H. Tran, S. M. Ahsan Kazmi, Tri Nguyen Dang, and Choong Seon Hong. «Edge-Computing-Enabled Smart Cities: A Comprehensive Survey». In: *IEEE Internet of Things Journal* 7.10 (Apr. 2020). MAG ID: 3016097478, pp. 10200–10232. DOI: 10.1109/jiot.2020.2987070 (cit. on p. 8).

[22] Qinglin Qi and Fei Tao. «A Smart Manufacturing Service System Based on Edge Computing, Fog Computing, and Cloud Computing». In: *IEEE Access* 7 (June 2019). MAG ID: 2953287140, pp. 86769–86777. DOI: 10.1109/access.2019.2923610 (cit. on p. 9).

[23] Rahul Rai, Manoj Kumar Tiwari, Dmitry Ivanov, and Alexandre Dolgui. «Machine learning in manufacturing and industry 4.0 applications». en. In: *International Journal of Production Research* 59.16 (Aug. 2021), pp. 4773–4778. ISSN: 0020-7543, 1366-588X. DOI: 10.1080/00207543.2021.1956675. URL: https://www.tandfonline.com/doi/full/10.1080/00207543.2021.1956675 (visited on 07/11/2023) (cit. on p. 9).

[24] Jiafu Wan, Xiaomin Li, Hong-Ning Dai, Andrew Kusiak, Miguel Martinez-Garcia, Di Li, and Di Li. «Artificial-Intelligence-Driven Customized Manufacturing Factory: Key Technologies, Applications, and Challenges». In: 109.4 (2020). MAG ID: 3110512320, pp. 377–398. DOI: 10.1109/jproc.2020.3034808 (cit. on p. 9).

[25] Pawani Porambage, Jude Okwuibe, Madhusanka Liyanage, Mika Ylianttila, and Tarik Taleb. «Survey on Multi-Access Edge Computing for Internet of Things Realization». In: *IEEE Communications Surveys & Tutorials* 20.4 (2018), pp. 2961–2991. ISSN: 1553-877X, 2373-745X. DOI: 10.1109/COMST.2018.2849509. URL: https://ieeexplore.ieee.org/document/8391395/ (visited on 07/11/2023) (cit. on p. 9).

[26] Falko Dressler et al. «V-Edge: Virtual Edge Computing as an Enabler for Novel Microservices and Cooperative Computing». In: *IEEE Network* 36.3 (May 2022), pp. 24–31. ISSN: 0890-8044, 1558-156X. DOI: 10.1109/MNET.001.2100491. URL: https://ieeexplore.ieee.org/document/9829315/ (visited on 06/20/2023) (cit. on p. 9).

[27] Chelsea Amanatides, Oana Ghita, Ken E Evans, and Genevieve Dion. «Characterizing and predicting the self-folding behavior of weft-knit fabrics». en. In: *Textile Research Journal* (May 2022), p. 004051752210996. ISSN: 0040-5175, 1746-7748. DOI: 10.1177/00405175221099670. URL: http://journals.sagepub.com/doi/10.1177/00405175221099670 (visited on 07/11/2023) (cit. on p. 9).

[28] Rafael A. Rojas and Erwin Rauch. «From a literature review to a conceptual framework of enablers for smart manufacturing control». en. In: *The International Journal of Advanced Manufacturing Technology* 104.1-4 (Sept. 2019), pp. 517–533. ISSN: 0268-3768, 1433-3015. DOI: 10.1007/s00170-019-03854-4. URL: http://link.springer.com/10.1007/s00170-019-03854-4 (visited on 06/16/2023) (cit. on p. 10).

[29] Bojana Bajic, Aleksandar Rikalovic, Nikola Suzic, and Vincenzo Piuri. «Industry 4.0 Implementation Challenges and Opportunities: A Managerial Perspective». In: *IEEE Systems Journal* 15.1 (Mar. 2021), pp. 546–559. ISSN: 1932-8184, 1937-9234, 2373-7816. DOI: 10.1109/JSYST.2020.3023041. URL: https://ieeexplore.ieee.org/document/9207825/ (visited on 06/16/2023) (cit. on p. 10).

[30] Matti Yli-Ojanperä, Seppo Sierla, Nikolaos Papakonstantinou, and Valeriy Vyatkin. «Adapting an Agile Manufacturing Concept to the Reference Architecture Model Industry 4.0: A Survey and Case Study». In: *Journal of Industrial Information Integration* (2019). DOI: 10.1016/j.jii.2018.12.002 (cit. on pp. 12, 15, 17).

[31] Yuting Li, Junhui Jiang, Changdae Lee, and Seung Ho Hong. «Practical Implementation of an OPC UA TSN Communication Architecture for a Manufacturing System». In: *IEEE Access* 8 (2020). MAG ID: 3094721173 S2ID: c00b3be675a4af2c621c461d5db3262ca8c00a3f, pp. 200100–200111. DOI: 10.1109/access.2020.3035548 (cit. on p. 12).

[32] Yuqian Lu, Chao Liu, Kevin I-Kai Wang, Huiyue Huang, and Xun Xu. «Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues». en. In: *Robotics and Computer-Integrated Manufacturing* 61 (Feb. 2020), p. 101837. ISSN: 07365845. DOI: 10.1016/j.rcim.2019.101837. URL: https://linkinghub.elsevier.com/retrieve/pii/S0736584519302480 (visited on 07/11/2023) (cit. on p. 12).

[33] Alexandros Bousdekis and Gregoris Mentzas. «Enterprise Integration and Interoperability for Big Data-Driven Processes in the Frame of Industry 4.0». In: *Frontiers in Big Data* 4 (June 2021), p. 644651. ISSN: 2624-909X. DOI: 10.3389/fdata.2021.644651. URL: https://www.frontiersin.org/

articles/10.3389/fdata.2021.644651/full (visited on 07/11/2023) (cit. on p. 12).

[34] Pedro Pinheiro and Goran Putnik. «Organizational efficiency prospects for management in Industry 4.0». en. In: *FME Transactions* 49.4 (2021), pp. 773–783. ISSN: 1451-2092, 2406-128X. DOI: 10.5937/fme2104773P. URL: https://scindeks.ceon.rs/Article.aspx?artid=1451-20922104773P (visited on 07/11/2023) (cit. on p. 12).

[35] Lihui Wang and Azadeh Haghighi. «Combined strength of holons, agents and function blocks in cyber-physical systems». en. In: *Journal of Manufacturing Systems* 40 (July 2016), pp. 25–34. ISSN: 02786125. DOI: 10.1016/j.jmsy.2016.05.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S0278612516300152 (visited on 09/13/2023) (cit. on p. 12).

[36] Jehn-Ruey Jiang. «An improved cyber-physical systems architecture for Industry 4.0 smart factories». en. In: *Advances in Mechanical Engineering* 10.6 (June 2018). ISSN: 1687-8140, 1687-8140. DOI: 10.1177/1687814018784192. URL: http://journals.sagepub.com/doi/10.1177/1687814018784192 (visited on 09/13/2023) (cit. on pp. 12, 13, 16).

[37] Rabiya Abbasi, Pablo Martinez, and Rafiq Ahmad. «The digitization of agricultural industry – a systematic literature review on agriculture 4.0». en. In: *Smart Agricultural Technology* 2 (Dec. 2022), p. 100042. ISSN: 27723755. DOI: 10.1016/j.atech.2022.100042. URL: https://linkinghub.elsevier.com/retrieve/pii/S2772375522000090 (visited on 09/13/2023) (cit. on p. 13).

[38] Shengjing Sun, Xiaochen Zheng, Bing Gong, Jorge García Paredes, and Joaquín Ordieres-Meré. «Healthy Operator 4.0: A Human Cyber–Physical System Architecture for Smart Workplaces». en. In: *Sensors* 20.7 (Apr. 2020), p. 2011. ISSN: 1424-8220. DOI: 10.3390/s20072011. URL: https://www.mdpi.com/1424-8220/20/7/2011 (visited on 09/13/2023) (cit. on p. 13).

[39] Mengru Tu, Ming K. Lim, and Ming-Fang Yang. «IoT-based production logistics and supply chain system – Part 2: IoT-based cyber-physical system: a framework and evaluation». en. In: *Industrial Management & Data Systems* 118.1 (Feb. 2018), pp. 96–125. ISSN: 0263-5577. DOI: 10.1108/IMDS-11-2016-0504. URL: https://www.emerald.com/insight/content/doi/10.1108/IMDS-11-2016-0504/full/html (visited on 09/13/2023) (cit. on p. 13).

[40] Luis V. Calderita, Araceli Vega, Sergio Barroso-Ramírez, Pablo Bustos, and Pedro Núñez. «Designing a Cyber-Physical System for Ambient Assisted Living: A Use-Case Analysis for Social Robot Navigation in Caregiving Centers». en. In: *Sensors* 20.14 (July 2020), p. 4005. ISSN: 1424-8220. DOI: 10.

3390/s20144005. URL: https://www.mdpi.com/1424-8220/20/14/4005 (visited on 09/13/2023) (cit. on p. 13).

[41] Ahmadzai Ahmadi, Ali Hassan Sodhro, Chantal Cherifi, Vincent Cheutet, and Yacine Ouzrout. «Evolution of 3C Cyber-Physical Systems Architecture for Industry 4.0». In: *Service Orientation in Holonic and Multi-Agent Manufacturing*. Ed. by Theodor Borangiu, Damien Trentesaux, André Thomas, and Sergio Cavalieri. Vol. 803. Series Title: Studies in Computational Intelligence. Cham: Springer International Publishing, 2019, pp. 448–459. ISBN: 978-3-030-03002-5 978-3-030-03003-2. DOI: 10.1007/978-3-030-03003-2_35. URL: http://link.springer.com/10.1007/978-3-030-03003-2_35 (visited on 09/13/2023) (cit. on p. 13).

[42] Vincent Havard, M'hammed Sahnoun, Belgacem Bettayeb, Fabrice Duval, and David Baudry. «Data architecture and model design for Industry 4.0 components integration in cyber-physical production systems». en. In: *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 235.14 (Dec. 2021), pp. 2338–2349. ISSN: 0954-4054, 2041-2975. DOI: 10.1177/0954405420979463. URL: http://journals.sagepub.com/doi/10.1177/0954405420979463 (visited on 09/13/2023) (cit. on p. 13).

[43] Juan Nie, Rui Zhi Sun, and Xiao Hua Li. «A Precision Agriculture Architecture with Cyber-Physical Systems Design Technology». In: *Applied Mechanics and Materials* 543-547 (Mar. 2014), pp. 1567–1570. ISSN: 1662-7482. DOI: 10.4028/www.scientific.net/AMM.543-547.1567. URL: https://www.scientific.net/AMM.543-547.1567 (visited on 09/13/2023) (cit. on p. 13).

[44] Dabeeruddin Syed, Ameema Zainab, Ali Ghrayeb, Shady S. Refaat, Haitham Abu-Rub, and Othmane Bouhali. «Smart Grid Big Data Analytics: Survey of Technologies, Techniques, and Applications». In: *IEEE Access* 9 (2021), pp. 59564–59585. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3041178. URL: https://ieeexplore.ieee.org/document/9272794/ (visited on 09/13/2023) (cit. on p. 13).

[45] B. Syed, A. Pal, K. Srinivasarengan, and P. Balamuralidhar. «A smart transport application of cyber-physical systems: Road surface monitoring with mobile devices». In: *2012 Sixth International Conference on Sensing Technology (ICST)*. Kolkata: IEEE, Dec. 2012, pp. 8–12. ISBN: 978-1-4673-2248-5 978-1-4673-2246-1 978-1-4673-2245-4. DOI: 10.1109/ICSensT.2012.6461796. URL: http://ieeexplore.ieee.org/document/6461796/ (visited on 09/13/2023) (cit. on p. 14).

[46] Gernot Steindl, Martin Stagl, Lukas Kasper, Wolfgang Kastner, and Rene Hofmann. «Generic Digital Twin Architecture for Industrial Energy Systems». en. In: *Applied Sciences* 10.24 (Dec. 2020), p. 8903. ISSN: 2076-3417. DOI: 10.

3390/app10248903. URL: https://www.mdpi.com/2076-3417/10/24/8903 (visited on 09/13/2023) (cit. on p. 14).

[47] Pablo F. S. Melo, Eduardo P. Godoy, Paolo Ferrari, and Emiliano Sisinni. «Open Source Control Device for Industry 4.0 Based on RAMI 4.0». en. In: *Electronics* 10.7 (Apr. 2021), p. 869. ISSN: 2079-9292. DOI: 10.3390/electronics10070869. URL: https://www.mdpi.com/2079-9292/10/7/869 (visited on 09/21/2023) (cit. on p. 14).

[48] Salvatore Cavalieri and Marco Giuseppe Salafia. «A Model for Predictive Maintenance Based on Asset Administration Shell». en. In: *Sensors* 20.21 (Oct. 2020), p. 6028. ISSN: 1424-8220. DOI: 10.3390/s20216028. URL: https://www.mdpi.com/1424-8220/20/21/6028 (visited on 07/11/2023) (cit. on p. 15).

[49] Chao Liu, Hrishikesh Vengayil, Yuqian Lu, and Xun Xu. «A Cyber-Physical Machine Tools Platform using OPC UA and MTConnect». en. In: *Journal of Manufacturing Systems* 51 (Apr. 2019), pp. 61–74. ISSN: 02786125. DOI: 10.1016/j.jmsy.2019.04.006. URL: https://linkinghub.elsevier.com/retrieve/pii/S0278612518301110 (visited on 07/11/2023) (cit. on p. 15).

[50] Sebastian Bader, Maria Maleshkova, and Steffen Lohmann. «Structuring Reference Architectures for the Industrial Internet of Things». In: *Future Internet* (2019). DOI: 10.3390/fi11070151 (cit. on p. 15).

[51] Helbert da Rocha, Reza Abrishambaf, João Pereira, and António Espírito Santo. «Integrating the IEEE 1451 and IEC 61499 Standards With the Industrial Internet Reference Architecture». In: *Sensors* (2022). DOI: 10.3390/s22041495 (cit. on pp. 15, 17).

[52] Diego G.S. Pivoto, Luiz F.F. De Almeida, Rodrigo Da Rosa Righi, Joel J.P.C. Rodrigues, Alexandre Baratella Lugli, and Antonio M. Alberti. «Cyber-physical systems architectures for industrial internet of things applications in Industry 4.0: A literature review». en. In: *Journal of Manufacturing Systems* 58 (Jan. 2021), pp. 176–192. ISSN: 02786125. DOI: 10.1016/j.jmsy.2020.11.017. URL: https://linkinghub.elsevier.com/retrieve/pii/S0278612520302119 (visited on 06/16/2023) (cit. on pp. 16, 17).