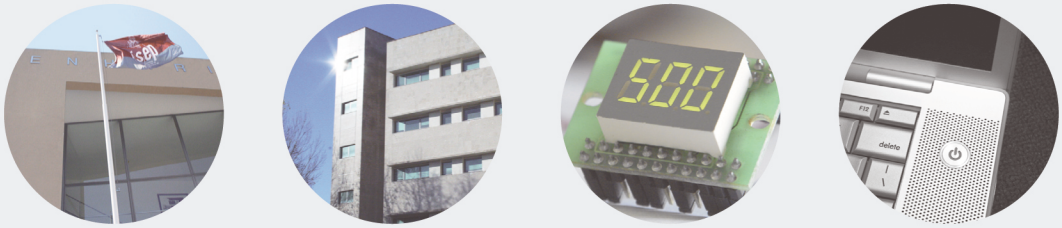
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES



Proof of Concept for a Visualization Interface into the Intralogistics Process

STÉPHANE CASTANHEIRA OLIVEIRA
Setembro de 2023

POLITÉCNICO
DO PORTO

Politécnico do Porto
Instituto Superior de Engenharia do Porto

Proof of Concept for a Visualization Interface into the Intralogistics Process

Stéphane Castanheira Oliveira

Master in Electrical and Computer Engineering
Specialization Area of Automation and Systems

Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto

September, 2023

This dissertation partially satisfies the requirements of the Thesis/Dissertation course of the program Master in Electrical and Computer Engineering, Specialization Area of Automation and Systems.

Candidate: Stéphane Castanheira Oliveira, No. 1181181, 1181181@isep.ipp.pt

Scientific Guidance: Manuel Fernando dos Santos Silva, mss@isep.ipp.pt

Company: Siemens S.A.

Advisor: Tiago Teixeira Almeida Aguiar, tiago.aguiar@siemens.com

Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto
Rua Dr. António Bernardino de Almeida, 431, 4200–072 Porto

September, 2023

Acknowledgements

I would like to express my sincere gratitude to the Siemens intralogistics team, particularly Eng. Tiago Aguiar, Eng. Nelson Cadilhe, and Eng. Fabian Zohm. Their generous opportunity for me to partake in the internship laid the very foundation upon which this thesis stands. Their unwavering support and invaluable guidance were indispensable factors in bringing this master's thesis to realisation.

A special acknowledgement goes to Dr. Manuel Silva for his consistent support and wisdom, which were instrumental in steering me towards the successful culmination of this milestone in my academic journey.

I'm deeply thankful for the support of my family and friends. Their encouragement and belief in me have been crucial in my academic pursuits and in successfully finishing both my internship and thesis.

Abstract

For the past few decades, the intralogistics sector has experienced constant growth and development. As digitalization has taken hold, the need for modern and efficient solutions has become more pressing. Industrial automation manufacturers have risen to the challenge and are dedicated to satisfying this demand.

To ensure effective control and monitoring of industrial applications, it is essential to consider the interaction between humans and machines. In the past, visualization systems were limited, regarding the functionality they offered, and visually unappealing, leading to failures and high associated costs. However, with the rapid development of web technologies, it is now possible to create clean and modern interfaces that are easy to read and interact with. To meet market demands, Siemens has recently developed SIMATIC WinCC Unified, a new system for the development of visualizations for control and supervision levels that is characterised by hardware abstraction and flexibility.

The primary objective of this dissertation is to utilise the new WinCC Unified system to redesign and improve the visualizations of the Siemens standard for intralogistics implementations, with a particular focus on the conveyor system. Additionally, the project involves hardware implementation using a structure that includes a server, to store and manage the visualizations, and multiple distributed clients that can access them. Furthermore, a study is conducted on the technologies that could be used to develop a completely wireless concept for this system.

Given that the implementation of a completely wireless system did not prove to be the most suitable course of action for the INTRALOG CS standard in the near future, due to the complexity and costs it would entail, a wired concept was employed. This choice allowed for the successful execution and validation of the transition from the WinCC Advanced visualization system to Unified, accomplishing all the proposed objectives at both the hardware and software levels.

Keywords: industrial automation, intralogistics, conveyor systems, visualization systems, Siemens SIMATIC WinCC Unified.

Resumo

Ao longo das últimas décadas, o setor da intralogística apresentou um rápido crescimento e constante desenvolvimento. À medida que a digitalização se consolida, a necessidade de implementar soluções modernas e eficientes torna-se cada vez mais evidente. Para suprir estas necessidades, os fabricantes de sistemas de automação industrial dedicaram-se a solucionar este desafio.

Para garantir um controlo e monitorização efetiva das aplicações industriais, é essencial abordar a interação entre o Homem e a máquina. No passado, os sistemas de visualização eram limitados, relativamente às funcionalidades que proporcionavam, e visualmente pouco apelativos, levando a falhas e altos custos associados às mesmas. Contudo, com o rápido desenvolvimento das tecnologias *web*, atualmente é possível desenvolver interfaces mais limpas e modernas, sendo mais acessível a sua leitura e interação. Para acompanhar os requisitos impostos pelo mercado, recentemente a Siemens desenvolveu o SIMATIC WinCC Unified, um novo sistema para o desenvolvimento de visualizações para os níveis de controlo e supervisão, caraterizado pela abstração ao *hardware* e flexibilidade.

O principal objetivo desta dissertação passa pela utilização desta solução, WinCC Unified, para redesenhar e melhorar as visualizações do *standard* desenvolvido pela Siemens para implementações de intralogística, particularmente para o sistema de transportadores. Adicionalmente, o projeto também envolve a implementação em *hardware*, usando uma estrutura constituída por um servidor para armazenar e gerir as visualizações e múltiplos clientes distribuídos que lhes podem aceder. De forma complementar, é efetuado um estudo sobre tecnologias que poderiam ser utilizadas para implementar um conceito completamente sem fios deste sistema.

Dado que a implementação de um sistema completamente sem fios não se revelou a opção mais adequada para o INTRALOG CS *standard* num futuro próximo, devido à complexidade e aos custos associados, foi adotado um conceito com fios. Esta escolha permitiu a execução e validação bem-sucedida da transição do sistema de visualizações de WinCC Advanced para Unified, alcançando todos os objetivos propostos tanto ao nível do *hardware* como do *software*.

Palavras-Chave: automação industrial, intralogística, sistema de transportadores, sistema de visualizações, Siemens SIMATIC WinCC Unified.

Contents

List of Figures

List of Tables

Listings

List of Acronyms

AGV	Automatic Guided Vehicle
AP	Access Point
AS/RS	Automated Storage & Retrieval System
CNC	Computerized Numerical Control
CP	Control Point
CPU	Central Processing Unit
CSS	Chirp Spread Spectrum
DB	Data Block
DCF	Distributed Coordination Function
ERP	Enterprise Resource Planning
FB	Function Block
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HMI	Human-Machine Interface
HTML5	HyperText Markup Language 5
I/O	Input/Output
IDB	Instance Data Block
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
IP	Internet Protocol
IPC	Industrial Personal Computer

iPCF	industrial Point Coordination Function
iPCF-MC	industrial Point Coordination Function-Management Channel
ISA	International Society of Automation
ISO	International Organization for Standardization
IT	Information Technology
IWLAN	Industrial Wireless Local Area Network
IWN	Industrial Wireless Network
KPI	Key Performance Indicators
LAN	Local Area Network
LED	Light Emitting Diode
MES	Manufacturing Execution System
OPC	Open Platform Communications
PC	Personal Computer
PCF	Point Coordination Function
PDF	Portable Document Format
PLC	Programmable Logic Controller
PROFIBUS	Process Field Bus
PROFINET	Process Field Net
PRP	Parallel Redundancy Protocol
PSK	Phase Shift Keying
RAM	Random Access Memory
RFID	Radio Frequency Identification
ROM	Read-Only Memory
RS	Recommended Standard
RTLS	Real-Time Locating System
RTU	Remote Terminal Units

SCADA Supervisory Control And Data Acquisition

SQL Structured Query Language

SVG Scalable Vector Graphics

TCP Transmission Control Protocol

TCP/IP Transmission Control Protocol/Internet Protocol

TDOA Time Difference of Arrival

TIA Portal Totally Integrated Automation Portal

TWR Two Way Ranging

UCP Unified Comfort Panel

UDT User Defined Data Type

USB Universal Serial Bus

UWB Ultra Wide-Band

VB Visual Basic

VoT View of Things

WAN Wide Area Network

WDS Wireless Distribution System

Windows CE Windows Consumer Electronics

WLAN Wireless Local Area Network

WMS Warehouse Management System

Chapter 1

Introduction

Intralogistics, also known as in-house logistics, was recognised as a component of logistics by the Mechanical Engineering Industry Association in 2005. Their definition specifies that intralogistics entails organising, controlling, implementing, and optimising the flow of materials, information, and goods within industry, commerce, and public institutions. This involves managing various aspects of internal logistics, such as warehouse operations, material handling, and the use of conveying technology and software. Coordinating sensors, actuators, and identification systems, as well as planning and executing logistics infrastructure and processes, are all part of the intralogistics domain. The overarching objective is to ensure the efficient and effective operation of these internal systems [1, 2].

The history of intralogistics dates back to the postwar period. Manufacturing was the main driver of economic and industrial development at that time, and in-plant transportation relied on basic equipment such as bag carts, trolleys, and overhead cranes. However, the introduction of transport techniques in the United States of America during the early 1950s led to the development of the universal pallet, which is still used today. The use of standardised loading devices like containers and pallets led to the development of forklifts and stacker cranes, making vertical storage a space-saving and efficient option. In the 1960s, economic growth resulted in the rapid expansion of manufacturing, distribution, and international trade. Company managers recognised the need for efficiency in all operations, not just manufacturing. Utilising available technology, storage configurations were reformulated, resulting in greater processing capacity and cost optimisation. The 1970s

saw the introduction of new technologies like electric monorail systems, Automatic Guided Vehicle (AGV), and Automated Storage & Retrieval System (AS/RS). The 1980s marked the deployment of new Information Technology (IT) and communication technologies, as well as the use of barcode labels and scanners to link flows of goods and information. Computer systems and data networks based on Ethernet were developed, optimising warehouse management. New strategies motivated the construction of larger industries with their customers. E-commerce and Wireless Local Area Network (WLAN) technologies, among other significant advances, were important driving forces for this expansion. More recently, Radio Frequency Identification (RFID) technology was introduced, which allows transport devices to carry more information than the bar code system through a microchip. Overall, intralogistics has a rich history of innovation and technological advancements, driven by the need for efficient and cost-effective storage and transportation of goods [2].

After decades of growth, the market has reached a position where the continued pursuit of digitalization is also applied to intralogistics. Being described as the initial conversion of products and services into a digital format, when applied to the intralogistics scenario it involves primarily four aspects: people, objects, processes and facilities [1]. Even though all of them are intrinsically related, two of them can be highlighted for the purpose of this project, being people and processes. With the ongoing development of automation, processes have a tendency to require less human intervention, however control and supervision capabilities are a must in this kind of systems. So, it is crucial to have a modern and easy to read/operate Human-Machine Interface (HMI). This will allow fewer errors, quicker reaction times to faults, and make it easier to employ personnel with little training [2].

1.1 Background

Siemens decided to create a single standard for industrial intralogistics installation projects after the release of the Totally Integrated Automation Portal (TIA Portal). The goal was to develop a consistent foundation for conveyor technology installations and AS/RS. The TIA Portal INTRALOG standard was created in 2015, by the Siemens Stuttgart branch, to meet the stated goal. The standard is separated into sub-standards for various types of intralogistics systems. For example, the standard for AS/RS is INTRALOG SC (SC stands for stacker crane), while the standard for conveyor systems is INTRALOG CS. In the meantime, it has been used and tested on a multitude of projects. Based on the experience gained during these implementations, many adjustments and modifications to the standard have been employed and new versions released.

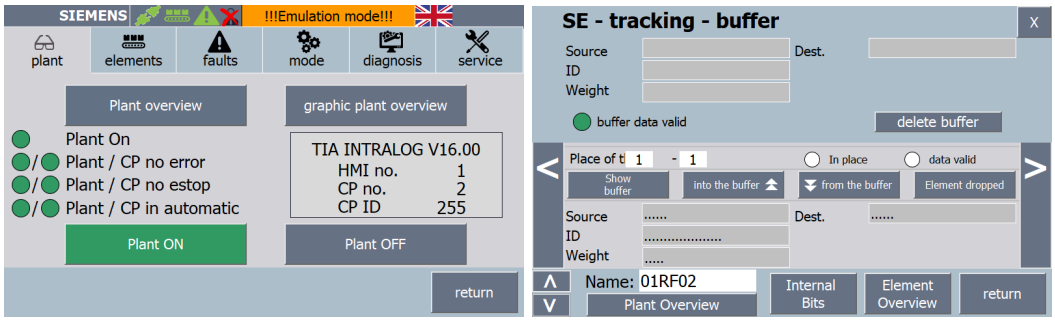
The TIA Portal INTRALOG standard not only encompasses the control technology program but also incorporates the project planning for system visualization.

These visualizations are utilised to exhibit system data and information, while also serving as the user interface for operating the system. Process visualization systems are employed to create these visualizations, which facilitate the development and provide an environment for configuring and designing the operator panels. In the standard, these visualizations are categorised into separate systems for the control level and process/supervision level. Specifically, TIA Portal INTRALOG utilises two distinct process visualization systems: SIMATIC WinCC Advanced for the control level and SIMATIC WinCC Professional system for the supervision level.

The work that follows is intended to contribute to the development of the TIA Portal INTRALOG CS, particularly the development of a concept for a visualization interface for conveyor technology on the control level, using the most recent visualization solution provided by Siemens - SIMATIC WinCC Unified (recurrently referred to in short form as WinCC Unified or simply Unified). This new solution seeks to integrate and reinvent the two previously existing ones, enabling for the development of visualizations for the control and supervision levels.

1.2 Problem Definition

The current system's outdated appearance and cumbersome operating patterns stem from the limited options available for configuring visualizations for the HMI through WinCC Advanced (Figure 1.1). Additionally, the lack of flexibility in the visualization negatively impacts the system's user-friendliness by today's standards, resulting in poor visual appeal and ultimately leading to reduced customer satisfaction.



(a) (b)

Figure 1.1: Examples of the visualizations on the current solution with WinCC Advanced: (a) Plant and (b) Data Tracking.

Siemens released SIMATIC WinCC Unified, a process visualization system, in late 2019. This platform utilises a new engineering approach and system concept, incorporating web technologies such as HyperText Markup Language 5 (HTML5) and

JavaScript. The updated system boasts significant advancements over its predecessor, including platform abstraction, increased flexibility, more efficient engineering, and the ability to create dynamic and modern user interfaces.

The transition to WinCC Unified is essential for the TIA Portal INTRALOG standard to meet market demands and provide contemporary solutions that keep up with competing manufacturers. Together with more up-to-date visualizations and flexible implementations, WinCC Unified makes it simple to reuse the visualization, which requires less engineering effort both at the control and process levels. So, the switch to the new process visualization system presents an opportunity to reevaluate the operating concept and implement a fresh idea for an HMI.

The migration of the visualization system to Unified is not straightforward because a completely new visualization concept is used. Porting, from WinCC Advanced to Unified, of the visualizations and operating programs, such as screens and scripts, is not possible natively. This means that, even if the same look and functionality are desired, the entire system's architecture will have to be recreated from scratch.

WinCC Unified's greater flexibility can help minimise the standard's development and maintenance effort in the medium to long term. The creation of operational programs can be reduced because some can run in parallel at the control and process levels. Furthermore, higher flexibility occurs as a result of platform independence, which on one hand broadens the spectrum of possible hardware for implementation and on the other simplifies the fulfilment of particular client requirements. Even if the generated work of this project is not directly implemented in the future, the expertise given through this document and the essential foundation of the operational programs developed can assist make future advances more expeditious.

Regarding the TIA Portal INTRALOG CS system architecture (Figure 1.2), in the current WinCC Advanced solution, at the control level there are essentially two ways of accessing the visualization system: fixed or mobile panels. In reference to the mobile panels, access is established via cable by connecting them to a connection box. During the installation phase of the TIA Portal INTRALOG CS standard, all panels will receive a copy of the visualization system via download. During operation, using the connection box to connect a mobile panel to the system, it is possible to provide the operator with control over certain groups of elements. As the connection boxes are located at fixed points, the position of an operator interacting with them is granted information. Access to the elements is based on the ability of the operators to physically see them for safety purposes.

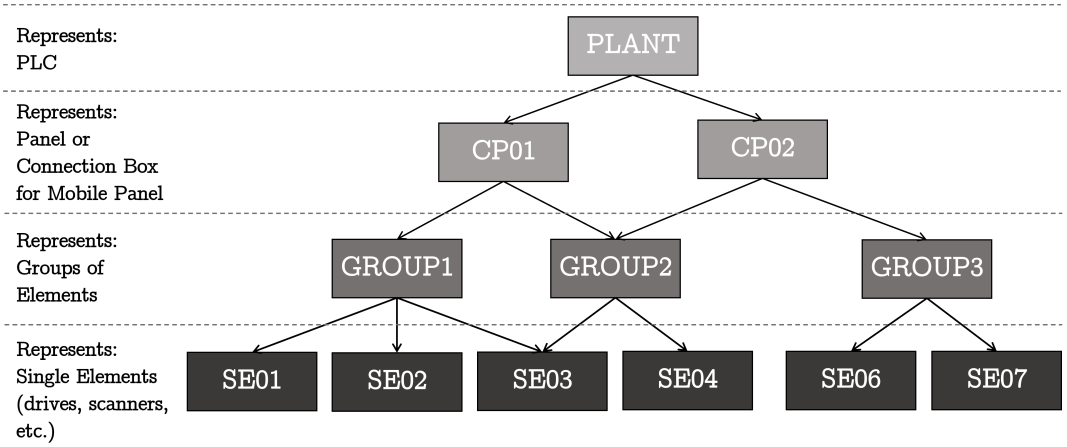


Figure 1.2: Overview of the INTRALOG CS installation structure.

The implementation in Unified can follow the same structure or adopt a completely wireless architecture. In both cases, the adoption of a system in which there are several clients distributed throughout the plant also requires adaptation in terms of the data access method. In Unified, a client-server architecture is adopted. In this instance, the clients do not have a copy of the visualization system on their hardware. The access to the visualizations by the Unified clients is made through a Unified server address. This leads to a problem: to meet the same safety requirements as the current version, it will be necessary to develop a way to determine where a certain web client is located in order to guarantee access only to the elements of the group in which it is located. Even if a mobile web client physically connected by cable to a connection box approach is used, since multiple web clients from various connection boxes will be accessing the same server at the same time, a method to identify which client is connected to a particular group of elements will still need to be developed, in order to manage the control based on location.

1.2.1 Objectives

The primary goal of this project is to create and integrate a visualization concept, built using WinCC Unified, into the INTRALOG standard for controlling conveyors. This implies an assessment of both hardware and software requirements. To accomplish this, the following objectives must be fulfilled:

- Analyse and comprehend the current state of the conveyor system control technology embedded in the INTRALOG CS standard.
- Study the functionality of WinCC Unified as the primary technology for implementation.
- Evaluate the hardware and associated costs needed to transition the visualization system to WinCC Unified on a completely wireless format.

- Development of cross-platform WinCC Unified visualizations that can be used on mobile panels or tablets available on the industrial plant.
- Deployment and validation of the visualization system on the chosen hardware.

1.3 Work Plan

In order to organise the work phases related to this project, it became necessary to estimate their duration. With the objective to clarify them, in Figure 1.3 it is possible to see a Gantt chart that represents the work plan, with the tasks, subtasks and respective duration.

Firstly, there are two tasks associated with the initial study of the technologies inherent to the project in development. Then a division of this work into three groups of tasks is established. The first is related to the research component, not only to apply the knowledge acquired in the implementation phase but also to carry out the writing of this document. The second group of tasks focuses on the development and implementation of the practical part of this project, as well as its validation. Lastly, the tasks related to writing the Thesis Dissertation are described, this being the longer-lasting ones.

1.4 Thesis Structure

This document is divided into seven chapters and two appendices that present the work accomplished over time; its content is briefly presented below.

Chapter 1 provides an introduction to the development of this work, beginning with a brief presentation of the concept of intralogistics and progressing to a description of the framework of this project as well as the objectives that are intended to be fulfilled.

Then, in Chapter 2, the relevant theoretical foundations for carrying out this project are approached.

In the Chapter 3, a study of the visualization systems currently available on the market is conducted, with a greater emphasis given on the technologies developed by Siemens.

Chapter 4 introduces the technologies that could be used if a completely wireless implementation of the visualization system is intended. To this end, in addition to an approach to relevant theoretical concepts, components and systems that could be used to achieve the desired solution are presented.

In Chapter 5, the conceptualisation of the new visualization interface is performed. Encompassing the chosen system architecture after the feasibility of the development of a complete wireless deployment is performed. Moreover, the design guidelines for the interface are established and explored.

Then, Chapter 6 is characterised by the description of the implementation process based on the conceptualisation previously performed.

Chapter 7 draws several conclusions about the work presented in this thesis as well as some ideas for future work.

In Appendix A, two sections can be found regarding the JavaScript code developed to support the functionalities of the WinCC Unified visualisation system. The first refers to the global module functions, and the second to local scripts.

Finally, Appendix B contains an explanation about the adjustments and new developments added to the PLC code of the INTRALOG CS standard.

ID	Task Name	Duration	Start	Finish
1	WinCC Unified study and preliminary tests	44 days	Tue 01/11/22	Fri 30/12/22
2	TIA INTRALOG CS standard analysis	66 days	Tue 01/11/22	Tue 31/01/23
3	**Research**	**85 days**	**Tue 01/11/22**	**Mon 27/02/23**
4	Theoretical foundations	85 days	Tue 01/11/22	Mon 27/02/23
5	Visualization systems available on the market	48 days	Tue 01/11/22	Thu 05/01/23
6	Technologies for implementing a wireless concept	42 days	Thu 22/12/22	Fri 17/02/23
7	**Development and implementation**	**132 days**	**Wed 01/02/23**	**Thu 03/08/23**
8	Visualization interface design and integration of new functionalities	128 days	Wed 01/02/23	Fri 28/07/23
9	Implementation and tests on the server panel	108 days	Wed 01/03/23	Fri 28/07/23
10	Integration of the mobile WebClients	16 days	Fri 07/07/23	Fri 28/07/23
11	Final testing and validation	5 days	Fri 28/07/23	Thu 03/08/23
12	**Thesis**	**205 days**	**Fri 18/11/22**	**Thu 31/08/23**
13	Writing	181 days	Fri 18/11/22	Fri 28/07/23
14	Revision	24 days	Mon 31/07/23	Thu 31/08/23

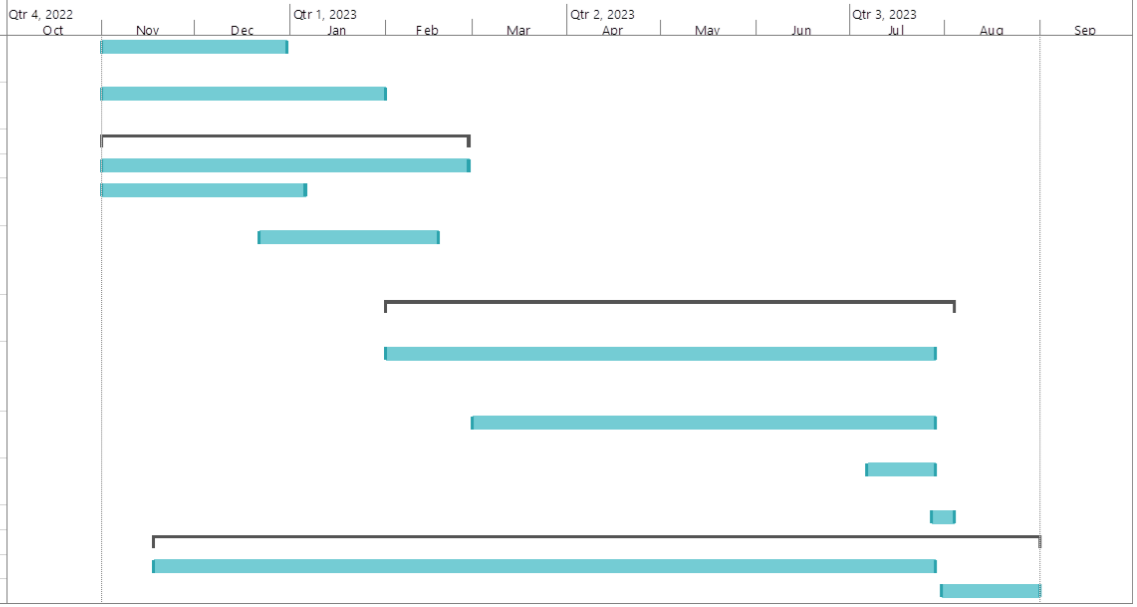


Figure 1.3: Gantt chart.

Chapter 2

Theoretical Foundations

The primary goal of this chapter is to explore the underlying theoretical principles inherent to the development of this project.

In the first phase, an approach will be taken into existing visualization systems in the context of industrial automation. Initially, a contextualisation of how they are incorporated in this sector is performed. Following that, each of them is studied in greater depth, explaining their evolution and the main characteristics of current systems.

Then, a brief overview of conveyor systems is given, with a focus on their application in the intralogistics sector. Several examples of the components that integrate these solutions are provided to help acknowledge their capabilities.

2.1 Visualization Systems

According to a presentation made in 2020 by Ramey Miller, Siemens' HMI Product Manager, the volume of manufacturing data generated each year will increase by five times until 2025, or an estimated increase to 175 zettabytes [3]. Numerous production facilities are becoming more complex as a result of the industry's and technology's ongoing development. This might result in faults not being detected in time, which would cause a machine or possibly an entire line to fail. Therefore, it is crucial to adopt a robust visualization solution in order to help reduce these issues and guarantee that plant operators, maintenance staff, and production managers have the proper overview of their production facilities [4].

In order to contextualise the insertion of visualization systems on the industry, Figure 2.1 exhibits the automation pyramid defined accordingly to International Electrotechnical Commission (IEC) 62264. The pyramid's higher levels are aimed at the corporate and plant levels. Enterprise Resource Planning (ERP), the fourth level, enables for the planning and control of resources like materials and staff. The third level, Manufacturing Execution System (MES), relates to a multi-layered manufacturing management system's process-related level. The most relevant levels to analyse in the context of this work, which encompass visualization systems, are levels two and one. Supervisory Control And Data Acquisition (SCADA) systems can be found on level two; they are used to remotely monitor and control technical processes, typically using visualizations that operate on computer systems. PLC and Industrial Personal Computer (IPC) are commonly located at the control level, which is the first in this architecture, to manage the shop floor. On the control level, HMI panels are present to enable direct interaction between the operators and the machines/processes [5].

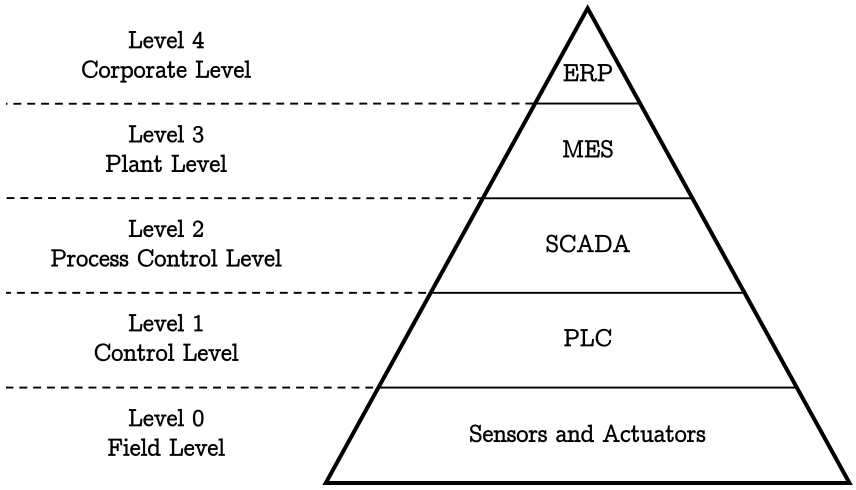


Figure 2.1: IEC 62264 industrial automation pyramid (based on [5]).

2.1.1 Human-Machine Interface

The term Human-Machine Interface, or HMI, is quite broad, encompassing any device that allows communication between a human and a machine; nonetheless, it is more commonly used in industry. In this instance, HMI are purpose-built hardware/software components designed to communicate via a predetermined protocol for interfacing with a PLC or an IPC. This way, an HMI is usually employed to oversee and manage a variety of industrial machinery, production lines, and continuous processes [6].

Historical Aspects of Human-Machine Interface

Nowadays, it is common to associate the term HMI with a tactile device with which an operator interacts; however, even a basic button or light indicator can be classified as an HMI because they provide control and/or monitoring over the system. Given that this concept encompasses several technologies, ranging from the simplest like a button, to a more complex system like a touch panel, it has been in use for many years, undergoing remarkable evolution. As such, it is possible to divide this development process into four essential generations, in chronological order, which describe revolutions in this technological field [7, 8]:

- **1st generation** - The initial generation of HMI were basic push-button interfaces used for machine control. They were frequently confined to a few buttons and indicators and did not offer much feedback or flexibility. Control rooms were utilised for remote visualization and control, while shop floor panels provided the same capability on a smaller scale. The third industrial revolution, with the introduction of PLC, motivated this generation.

- **2nd generation** - Graphical displays, such as monochrome text displays, were added in the second generation of HMI to provide more detailed feedback and control options. These systems were frequently built on proprietary hardware and software and were relatively uncustomizable. After some time, manufacturers eventually produced HMI that featured colour visuals and touchscreens, providing more intuitive and flexible control capabilities. Remote visualizations running on desktop apps were prevalent, as were local visualizations running on touch panels. Initially, networks were built on RS-232 and RS-485, but they later progressed to Ethernet. This revolution is distinguished by the emergence of concepts such as networking, computers, and computer graphics.

- **3rd generation** - Mobile and web-based interfaces were introduced in the third generation of HMI, allowing for remote monitoring and control of industrial systems. These interfaces are frequently built with HTML5, or other web technologies, and can be accessed from a variety of devices, including smartphones and tablets. The benefit of running visualizations on servers is that both local and remote visualization can be performed on the same platform. The display arrangement can be customised based on authentication, for local operators or remote supervisors, and a responsive design can be achieved based on the device accessing the system. This generation was propelled by the rapid development of the internet and web technologies.

- **4th generation** - The fourth generation of HMI is actively being researched and evaluated. It is focused on merging sophisticated technologies, such as

augmented and virtual reality, to deliver more immersive and intuitive interfaces for industrial control. These systems may also use machine learning and artificial intelligence to automate operations and provide predictive maintenance.

Given the background of the generations and the current state of the industry, the second and third generations are the most common nowadays [7].

Current State And Functionalities

Companies dedicated to the manufacture of products for industrial automation typically make available to the market a variety of tactile solutions that differ in their connectivity, size of display, and capabilities provided by the software for the development of visualizations, which is usually proprietary. Some of the most relevant solutions for this project, manufactured by Siemens, will be explored in further detail in the Chapter 3. However, for reference, and considering the large variety of products available on the market, some solutions from different manufacturers will be analysed as well.

When it comes to selecting an HMI panel for a specific implementation project, a number of factors must be evaluated and fulfilled. In terms of hardware, HMI can offer a variety of ports for connecting to controllers and peripherals, allowing the usage of a specific communication protocol or a Universal Serial Bus (USB) port. Another factor to consider is the screen type and size, as well as the panel's ability to function in harsh conditions, such as freezing or moist environments. The software functions that a visualization interface should provide are also generally defined based on the intended implementation project; however, it is common for a modern interface to be capable of providing the following functions [8]:

- Keep track of the machine/process inputs and outputs in real-time.
- Visually display and track data relevant to the machine or process being monitored/controlled, like production time, trends, and tags.
- Supervision of Key Performance Indicators (KPI), to maintain certain variables within acceptable ranges, by adjusting the working conditions of the process directly from the HMI.
- Management of alarms in order to detect problems or misbehaviour throughout the manufacturing process.

The visualization with which the operator interacts is an important component of an HMI solution; it should be designed in a way that it allows for easy reading of the contents displayed as well as quick detection of faults and alarms, allowing for faster

intervention to minimise costs associated with them. In the past, visualizations displayed complex, colour intensive, and overly informative designs. This made it difficult to determine whether the systems were working properly, leading to a very inefficient decision process [9].

In order to solve this problem, the International Society of Automation (ISA) published a standard in 2015, called ISA-101, with the aim of standardising the development of visualizations. The standard is targeted to end users, designers and developers of HMI. According to a publication made by the ISA in 2019, the "ISA-101 standard provides information, guidelines and a methodology to enable users to be more effective in yielding improved safety, quality, production, and reliability" [10].

2.1.2 Supervisory Control And Data Acquisition

Supervisory Control And Data Acquisition are not full-fledged dedicated control systems, but rather focus on the supervisory level. A SCADA system is an amalgamation of hardware and software that is often used to monitor and control industrial equipment and processes [11]. Unlike HMI panels on the shop floor, the goal of these systems is to provide higher-level supervision over a sector or an entire industrial plant. SCADA systems are expected to handle tasks such as alarm management, data historian (functionality to keep track and store all the data collected by the system using databases), report generation, and real-time tag and trend analysis.

Historical Evolution of SCADA Systems

Given that SCADA systems have been in use for several decades, their architecture has evolved in tandem with technological advancement. As a result, in chronological order, four major generations of these systems may be defined [11, 12]:

- **Monolithic** - This architecture was based on standalone mainframe systems that were not interconnected because networks were essentially nonexistent in those systems. The term Wide Area Network (WAN) was used to describe the connection to Remote Terminal Units (RTU) and the data exchange with the master computer. At that time, the SCADA installation was pretty much entirely dependent on a single vendor, and the communication protocols were proprietary.

- **Distributed** - This generation was propelled forward by the technological advancement of the Local Area Network (LAN). At the time, it was possible to establish real-time communication and data exchange between several stations. These began to resemble the industry's current structure, presenting

HMI, smaller controllers, and database servers. Even though the communication protocols were still proprietary, their capabilities demonstrated improvements in communication reliability and speed. However, due to the inability to communicate with external devices, this remained a limitation.

- **Networked** - The transition from previous generations to networked SCADA systems is primarily due to industry growth and the development of solutions from various vendors. This generation is distinguished mostly by the implementation of open system architectures and the integration of systems from different manufacturers. This enabled the removal of the previously mentioned limitations and the possibility of distributing SCADA functionalities across WAN using protocols such as Internet Protocol (IP).
- **Internet of Things (IoT) Cloud Based** - The current solutions use IoT technologies and cloud computing services, allowing data to be accessed from anywhere in the world. These systems are simple to interconnect, flexible, optimised, and quickly scalable.

As previously stated for the HMI on control level, the graphical visualizations that began to be deployed in control rooms in the 1980s were extremely difficult to read (Figure 2.2), not only as a result of technological limitations but also due to a lack of a well-formed basic guideline to build the visualizations.

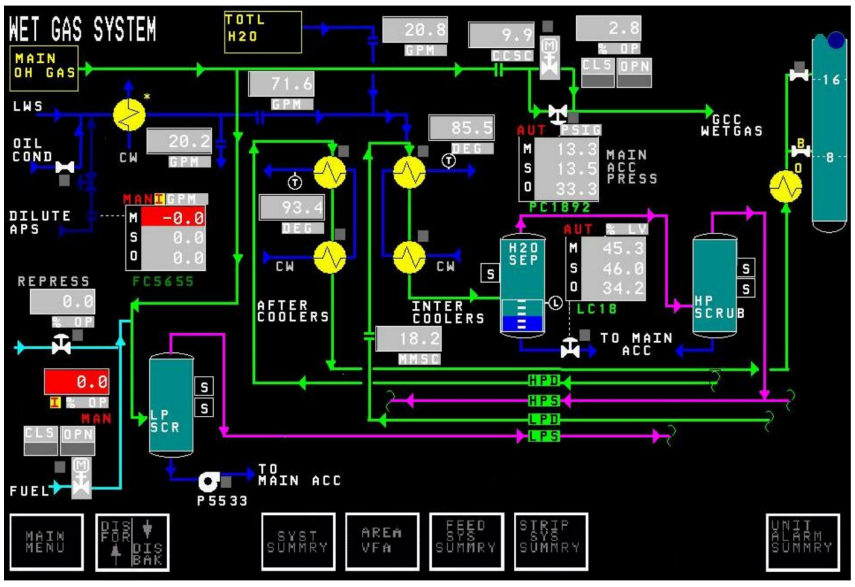


Figure 2.2: Difficult to read HMI example [9].

With the advancement of software focused on visualization design, new 3D type components and animation arose, resulting in even more complex and distracting panels, making it more difficult to identify alarms and alerts regarding faults and

problems on the shop floor. This condition became extremely problematic when accidents began to occur, resulting in the death and injury of people. The explosion at the Texaco Refinery in the United Kingdom in 1994 is one example of this type of situation. This disaster occurred as a result of flammable hydrocarbon liquid being continually injected into a process vessel that had its outlet closed due to a valve failure. Following an extensive examination, it was determined that part of the reasons that led to this catastrophe were connected to incorrect alarm management and lack of a properly designed supervisory system to assist operators in controlling the process. The issue could have been identified and prevented if the system had been set up to display information on volumetric or mass balance [9].

Current State And Functionalities

The High Performance HMI Handbook which was co-authored by two companies, PAS Global LLC and User Centered Design Services, was released in 2008 with this issue in mind. The book exposes numerous poor yet widespread HMI practises, argues for reform, and demonstrates in great detail how to create and deploy an HMI that performs at the highest level. The principles studied and specified in this book have become quite popular, and the term high performance HMI is now widely used and applied. According to the book's contents, concepts such as the proper use of colour, screen hierarchy, and the use of trends are covered with detail, when it comes to the implementation of visualizations. With the goal of offering practical tools for the development of visualizations, the principles outlined in this book can be summed to basically three, which operate as the base guidelines for the development and deployment of interfaces between the operator and the machines or processes [9]:

- **Clarity** - Modern visualization interfaces are mostly built of graphics, which should be easy to read and intuitively intelligible in order to clearly represent the process state and conditions. As anticipated, the graphics should be free of excessive detail and clutter, and should concentrate on providing useful information rather than presenting a large amount of data. Graphic elements used to manipulate the process, as well as prominent information, such as alarms and indications of abnormal conditions, should be quickly differentiated.

- **Consistency** - The visualization should be built with a consistent hierarchy and with performance and logical navigation in mind. To accomplish the desired functionality, the graphics should be uniform across the complete visualization solution and require minimal interaction.

- **Feedback** - In all circumstances, graphic elements and controls must act in a consistent manner. When dealing with significant actions, confirmation measures should be used to prevent accidental activation. To lessen operator fatigue brought on by continuous interaction with the interface, design methodologies should be put into practise.

Figure 2.3 illustrates a visualization developed using the high performance HMI approach. In contrast to the antiquated and potentially error-prone visualization shown in Figure 2.2, this modern interface employs grayscale colours and simple visuals. With a more structured layout and a cleaner appearance, operators will be able to interpret the information offered by the visualization and respond quickly to any failures that the system may present. Despite the fact that the book focuses on principles for decentralised control systems, many of the concepts provided might equally be used on ground-level HMI.

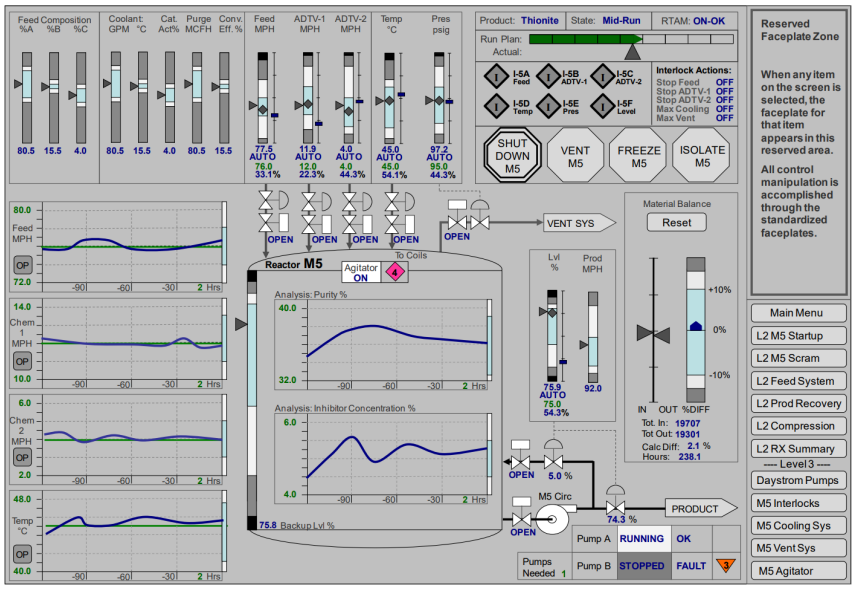


Figure 2.3: High performance HMI example [9].

A research on a petrochemical facility measured and contrasted workers' situational awareness between a traditional distributed control system environment and a high-performance HMI environment, using grayscale graphics and integrated alarm management. The improvements listed in Table 2.1 resulted in an estimated USD 800,000 in cost savings annually, due to the quick detection and action on faults [13].

Table 2.1: Impact of high performance HMI on operator performance [13].

Task	Traditional HMI	High performance HMI	Results
Detecting abnormal situations before alarms occur	10% of the time	48% of the time	5 x increase
Success rate handling abnormal situations	70%	96%	26% over base case
Time to complete abnormal situation tasks	18.1 minutes	10.6 minutes	41% reduction

2.2 Conveyor Systems

Conveyor systems are a vital aspect of intralogistics in modern industrial facilities, playing a critical role in the handling and transportation of several materials. These automated systems offer a reliable solution for transporting goods between different stages of manufacturing or storage within a plant. Industrial conveyors are capable of handling raw products, finished goods, and packaging materials. This technology is available in various designs and configurations, from basic gravity-driven roller conveyors to advanced automated systems moved by motors. They typically consist of a frame that holds a conveyor belt, chain, or set of rollers. Conveyors can be customised to meet specific production requirements, such as handling heavy loads or operating at high speeds. Moreover, conveyor systems can be integrated with other types of industrial equipment, such as picking stations, palletizers, and AS/RS. Conveyor systems are extensively used in intralogistics for tasks like material transportation, sorting, merging, and buffering. In an industrial configuration, conveying technologies have the potential to enhance production efficiency by expediting procedures and improving material transportation, leading to overall productivity improvement.

To meet the varying needs of several sectors, a variety of conveyor systems are offered. Among the many conveyor designs (Figures 2.4 to 2.6) are the following [14]:

- **Belt conveyor** - They utilise continuous belts looped endlessly between two or more end-pulleys, with one or both ends supported by a roller. Power for the conveyor system is provided by motors, which can use either constant or variable speed reduction gears. Belt conveyors are typically driven and can operate at various speeds to meet the desired throughput. Furthermore, they can be inclined or used horizontally. Belt conveyors fall into one of two categories: general material handling, which includes moving boxes around a plant, and large or bulk material handling.

Figure 2.4: Belt conveyor [15].

- **Chain conveyor** - Are a type of material handling system that employs mechanical devices, such as chains or cables, to drag or tow products. These devices are attached to moving members and are driven by a continuous chain. Chain conveyors are primarily used for carrying heavy loads and are relatively easy to install, requiring only minor maintenance like chain lubrication.

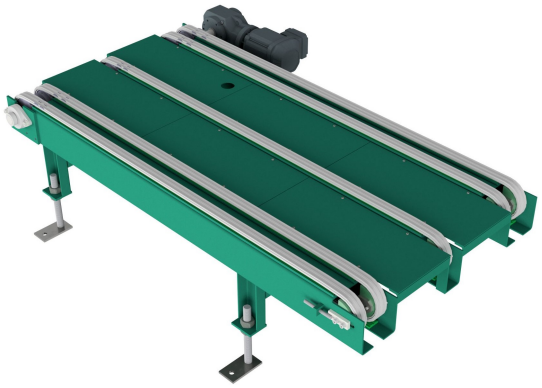


Figure 2.5: Chain conveyor [15].

- **Roller conveyor** - Frames house rollers that are employed in these systems for product transportation, either by gravity or manual means. Despite their straightforward design, roller conveyors have seen numerous advancements to align with evolving industry trends. A significant breakthrough in this technology involves linking the conveyor to a motor via a chain, belt, or shaft to enhance controllability.

Figure 2.6: Roller conveyor [15].

In order to enhance the flow and flexibility of the material transportation, the following elements (Figures 2.7 to 2.10) can be found integrating these types of systems:

- **Vertical conveyor** - Are used to move objects from one level to another by lifting or lowering a load. These conveyors are frequently placed between two horizontal conveyors to maintain continuous movement despite vertical deviations. They have a variety of applications, including saving floor space, connecting many levels, building vertical accumulation systems, establishing vehicle pick-up and drop-off sites, and a variety of other specific functions [16].

Figure 2.7: Vertical conveyor [17].

- **Turntable** - Provide a mechanism to rotate loads such that the product orientation is maintained through junctions where material lines intersect or change direction. Depending on the requirements of the particular application, they can also invert a product's orientation or let it pass through [18].

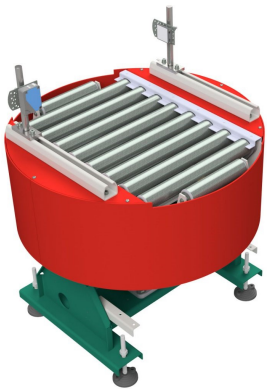


Figure 2.8: Turntable [15].

- **Corner converter** - Is made especially to turn the direction of product/pallet movement by 90 degrees. The fact that a corner converter achieves this shift in direction without changing the orientation of the objects being delivered is one of its special characteristics [15].

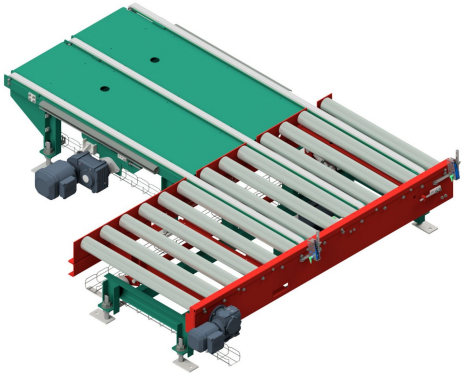


Figure 2.9: Corner converter [15].

- **Transfer car** - Allows loads to be redirected or combined between different lanes [19].

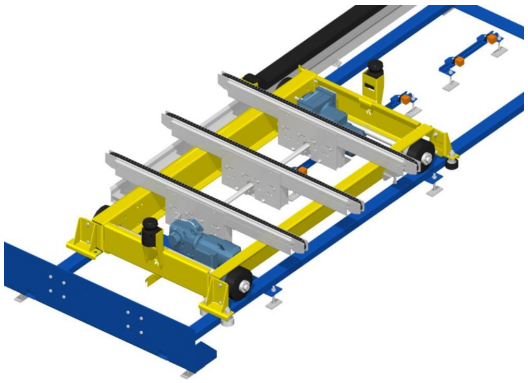


Figure 2.10: Transfer car with chain conveyor [19].

2.3 Conclusion

Manufacturing plants are becoming increasingly complex and automated as technology advances, with a strong focus placed on digitalization in recent years.

Over the past few decades, manufacturers of automation products have been developing more sophisticated visualization systems capable of meeting the needs of industrial projects. After a period in which interaction between humans and machines proved excessively complex, leading to economic and human losses, there is currently a push for simple interaction through modern and clean interfaces. To achieve this, standards and guidelines are emerging to guide developers in creating visualization systems and maximizing their capabilities. To apply these concepts, it is necessary to use hardware and systems developed by manufacturers in this sector. Therefore, Chapter 3 will present some alternatives that can be found currently on the market.

Finally, it is worth mentioning that conveyor systems play a vital role in intralogistics. They are highly versatile and can easily adapt to the unique requirements of each automation project.

Chapter 3

Visualization Systems Available on the Market

With the first sections, this chapter will provide an overview of Siemens's various visualization system solutions. In the first phase, the classic WinCC product line, which has been on the market for several years, will be discussed. Following that, WinCC Unified will be presented as the most recent version of this type of technology. The existing range of hardware and software for shop floor control and supervisory systems will be exposed in both instances.

On the final stage of this chapter, a brief overview of some visualization solutions from various manufacturers will be provided in order to give a more complete perspective of what the market has to offer nowadays. Consumers can now find a wide range of products that can fulfil the requirements of their projects, including fixed and mobile panels and software for SCADA development. Since the primary focus of this project is the implementation on the control level, the solutions that meet these requirements will be reviewed in greater detail than those that fit the supervision level.

3.1 Siemens SIMATIC WinCC

SIMATIC WinCC's visualization software comes in four different variations (Figure 3.1), each with its own set of capabilities and configurable operating systems. With the exception of WinCC Unified, there are four engineering variations of

WinCC available on the TIA Portal: Basic, Comfort, Advanced, and Professional [20]. Despite this segmentation into four divisions, due to their popularity, greater focus is normally given to the two largest: WinCC Advanced, which is often used for panel-based and Personal Computer (PC) single-user systems, and WinCC Professional, which is used for SCADA applications.

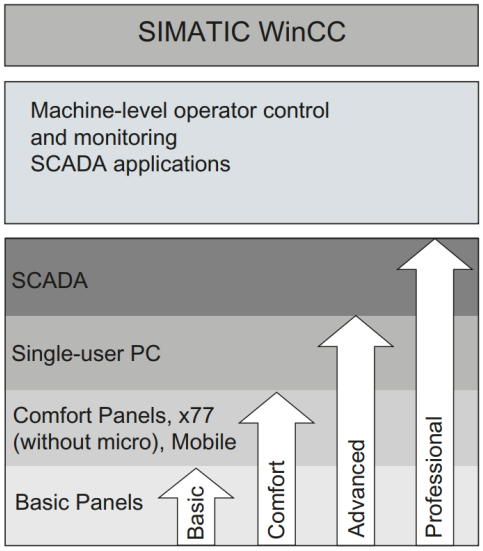


Figure 3.1: SIMATIC WinCC versions [20].

3.1.1 SIMATIC WinCC Advanced

As previously stated, WinCC Advanced can run on a panel or a PC. Three primary types of SIMATIC HMI panels can be found on the market: Basic, Comfort, and Mobile. There are numerous panel variations within a family, which are mostly defined by the screen diagonal size and resolution. Basic and Comfort Panels are intended for permanent installation, such as on a control cabinet door. Mobile panels, on the other hand, allow for in-hand operation and can be connected to connection boxes through a cable. Lastly, WinCC Runtime Advanced is intended to be executed on a PC. These alternatives will be discussed in greater depth below.

SIMATIC HMI Basic Panels

SIMATIC HMI Basic Panels (Figure 3.2a) are a starting point for simple HMI applications. The device series includes panels ranging from 4 to 12 inches in size, as well as key and touch operation. The 2nd generation features high resolution displays and 64000 colours support. In terms of communication, these panels include a USB interface that allows for the attachment of a keyboard, mouse, and barcode scanner, as well as data storage on a USB stick. Furthermore, depending on the version, an inbuilt Ethernet or RS 485/422 interface allows the connection to the controller [21].

SIMATIC HMI Comfort Panels

SIMATIC HMI Comfort Panels (Figure 3.2b) are intended for the machine-level deployment of high-performance visualization applications. These panels come in a variety of shapes and sizes, from 7 to 21.5 inches, with some featuring keys in addition to the display that enables multitouch. The panels integrate features such as a Portable Document Format (PDF) viewer, internet browser and media player. Concerning the interfaces available in the panels, they provide Process Field Bus (PROFIBUS) and Process Field Net (PROFINET) support as standard. Screens larger than 7 inches feature two PROFINET connections with an integrated network switch, while panels larger than 15 inches include a third PROFINET interface. There is also an audio Input/Output (I/O) interface and two USB ports [21, 22].

(a) (b)

Figure 3.2: SIMATIC HMI Panels: (a) Basic [21] and (b) Comfort [22].

SIMATIC HMI Mobile Panels

The SIMATIC HMI Mobile Panels 2nd generation (Figure 3.3a) are a wired mobile solution that offers the same performance and functionality as the HMI Comfort Panels. With a Mobile Panel, it is possible to control and monitor multiple plant areas. At the corresponding place, the panel can be plugged to a connection box (Figure 3.3b), allowing access to the desired plant section. These panels feature display sizes of 4, 7, 9 or 10 inches and are available in four variations [21]. The standard Mobile Panels do not include an emergency button on top or an acknowledgement button on the back and they may be useful in circumstances where failsafe functionality is not required but there is a need to employ devices that are compact and easy to handle [23]. There is a failsafe Mobile Panel variant available if the project calls for both an emergency stop button and an enabling switch. The SIMATIC HMI KTP700F Mobile Arctic is the last variant; it has a glass touch screen and is optimised for usage in places with low ambient temperatures, with a working temperature range of -28 °C to +45 °C [21].

(a) (b)

Figure 3.3: SIMATIC HMI Mobile Panels [23]: (a) Failsafe variant (b) Connection box.

SIMATIC WinCC Runtime Advanced

In contrast to the panel-based solutions mentioned above, SIMATIC WinCC Runtime Advanced is an open PC-based visualization solution focused on operator control and monitoring for single-user systems (Figure 3.4), allowing the parallel coupling of up to 8 controllers. The core package for visualization, reporting and logging, and user administration may be expanded using option packages and Visual Basic (VB) scripts development. Services such as remote operation, diagnostics, and administration across the intranet and internet in conjunction with email communication are possible due to its integration with automation systems based on Transmission Control Protocol/Internet Protocol (TCP/IP) networks. Although Siemens advises using a SIMATIC IPC for its implementation, any computer that satisfies the minimal operating requirements can be utilised to run WinCC Advanced Runtime [21].

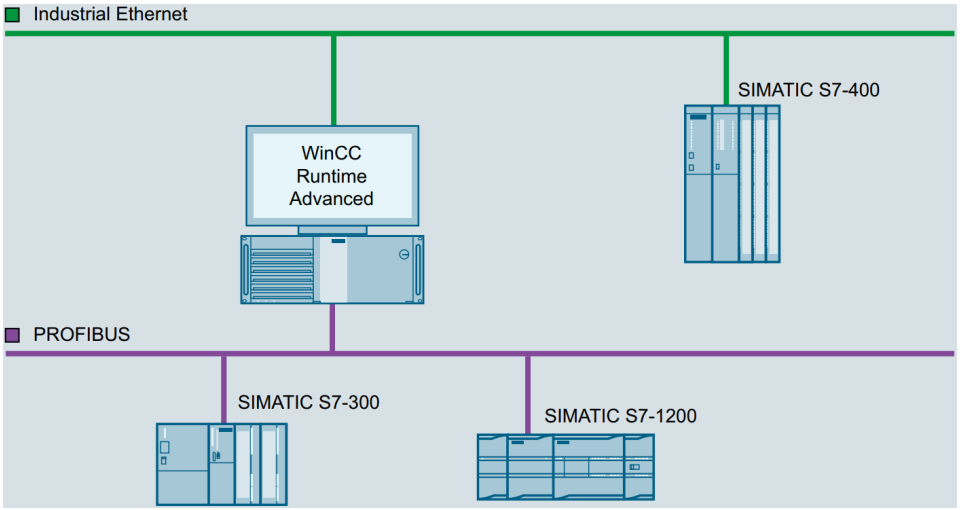


Figure 3.4: WinCC Runtime Advanced single-user system architecture [21].

3.1.2 SIMATIC WinCC Runtime Professional

WinCC Runtime Professional is a PC-based visualization solution for controlling and keeping track of processes, production flows, machinery, and plants across all industries. Instead of merely supporting single-user stations, as WinCC Runtime Advanced does, it may also be expanded to enable distributed multi-user systems (Figure 3.5) and cross-location solutions with web clients. WinCC Runtime Professional can be categorised as a SCADA system in summary.

If a more customised solution is required, the functionality may be extended using VB and C scripts. Additionally, WinCC Runtime Professional supports open interfaces like Open Platform Communications (OPC), enabling cross-vendor solutions and the capability to combine automation and IT applications. Regarding the hardware, depending on the communication processor installed, the number of controllers connected to the system via Industrial Ethernet or PROFIBUS can range from 8 to 64 [21, 24].

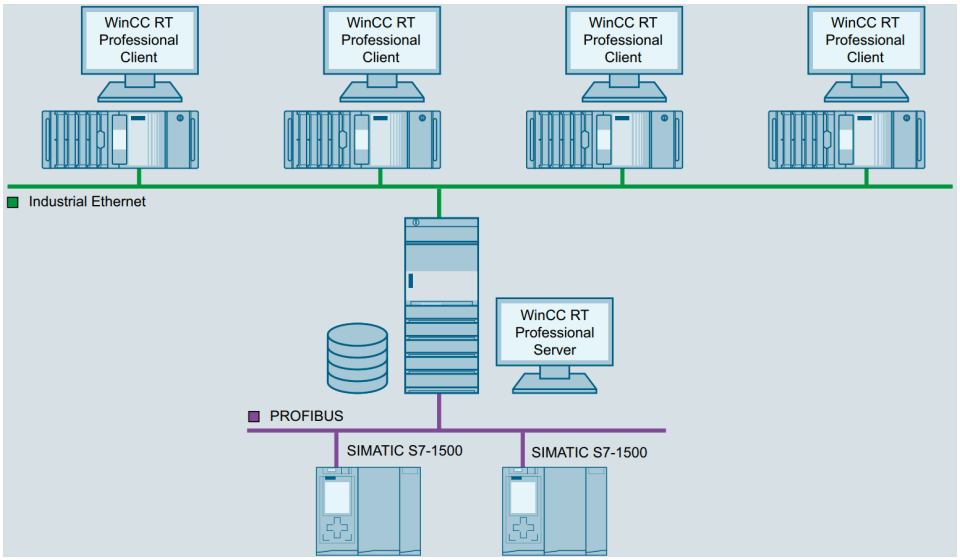


Figure 3.5: WinCC Runtime Professional multi-user system with operable server [21].

3.2 Siemens SIMATIC WinCC Unified

To meet market expectations, Siemens launched a new process visualization system (Figure 3.6), SIMATIC WinCC Unified, towards the end of 2019, for usage with the TIA Portal. With the help of this visualization system, the machine and plant engineering sector is able to overcome the challenges of digitalization. Modern web and edge technologies, paired with open interfaces, allow for a flexible execution of projects and application-specific requirements. To execute WinCC Unified, SIMATIC HMI Unified Comfort Panel (UCP) and PC-based systems can be used.

Since HTML5, JavaScript, and Scalable Vector Graphics (SVG) are supported, the visualization opens up a wide range of possibilities, allowing the project to be independent of devices and environment [25].

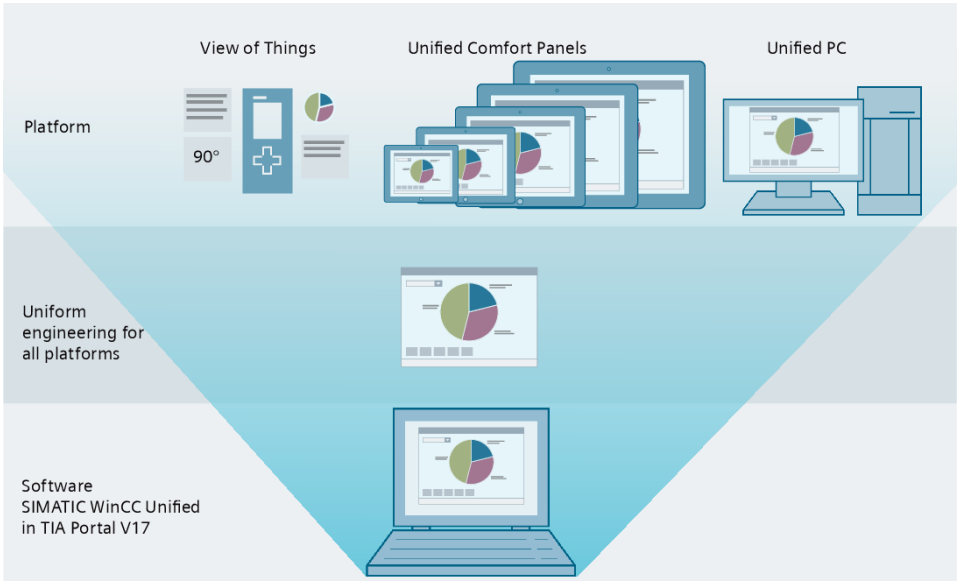


Figure 3.6: SIMATIC WinCC Unified architecture and solutions [25].

When compared to prior technologies, such as WinCC Advanced and Professional, Unified has a high level of hardware abstraction. The visualizations are no longer limited to platform-specific runtime systems, like the ones stated above. As a result, the hardware chosen for project implementation is significantly more adaptable. In addition to the more common operating systems, such as Windows PC and UCP, mobile devices, such as smartphones and tablets, can be used to access the visualizations. With this technology, SCADA systems and HMI panels no longer need to be designed independently, allowing for the avoidance of redundant development of operational programs for different platforms or operating systems. Taking into consideration the facts provided, WinCC Unified is distinguished by ease of scalability [26].

There are basically three different types of platforms that make up the WinCC Unified System. The first one, called View of Things (VoT), is the simplest and consists of a basic system of web browser visualizations that allows users to access data from the S7-1500 family of controllers through their built-in server. WinCC UCP and Unified PC are the other two solutions, and they are the more complete tools for developing HMI visualizations within the Unified family of products. The main features of each of the referred platforms are shown in Table 3.1. Although not exclusively dedicated to the Unified System, Siemens offers a Mobile WebClient referenced as IWP10F that can be used to access the visualization system. Each of these will be covered in further detail in the subsequent sections.

Table 3.1: Differences between simple web visualization and a complete HMI solution in the WinCC Unified System [27].

	View of Things	Comfort Panel	Unified PC
Basic objects and elements	✓	✓	✓
Graphics	✓	✓	✓
Screens	✓	✓	✓
Screen windows	✓	✓	✓
Faceplates		✓	✓
Alarming		✓	✓
Archive		✓	✓
Audit and reporting		✓	✓
Parameter control		✓	✓
Unified collaboration		✓	✓
Plant intelligence options			✓

3.2.1 WinCC Unified View of Things

The View of Things (VoT) capability allows the user to control and monitor essential control parameters directly from a S7-1500 controller's web server. This makes VoT a valid approach to construct basic tools for monitoring and controlling data in a PLC for commissioning or maintenance. Furthermore, VoT solutions might be particularly appealing when a machine is placed in a complex infrastructure, making it difficult to install an HMI panel on-site [27]. When conventional tablets or smartphones are adequate for operational control or monitoring and the ruggedness of industrial equipment is not required, VoT can appear as a less expensive choice for a simple visualization system. If the required functionality grows, the VoT visualization may be applied on a UCP or PC. Due to the fact that this solution is meant to develop small web applications that operate on the web server of a S7-1500 controller, there is a restriction of 10 screens and 100 tags per controller, due to performance concerns [28].

View of Things enables users to construct user-defined web pages in TIA Portal using the Unified editor and pre-defined graphical components. To create the visualizations, a drag-and-drop editor may be used, thus no programming skills and HTML5 code are required [27].

3.2.2 SIMATIC HMI Unified Comfort Panels

Based on the WinCC Unified visualization software, the UCP (Figure 3.7) provide particular operation concepts when compared to older solutions. All panels have the same amount of hardware connectors and functionality; the only difference between them is the screen size, which ranges from 7 to 21.5 inches. Furthermore, each UCP

comes in two design variations. The standard design features Siemens and SIMATIC HMI logos as well as a silver-coloured aluminium frame. There is a variation with no branding and a black aluminium frame if a more neutral look is required [29].

Figure 3.7: SIMATIC HMI UCP [29].

UCP come preloaded with applications that let users work in documents, watch instructional videos via a media player, and have access to web-based services. If more functions are required, the capabilities of the panels can be expanded by the installation of edge apps. Up to three simultaneous web clients enable remote access to the visualization, independent of the device's on-site operation, and can be accessed through a web browser without the need for any additional software. Aside from that, the panels support multitouch and are able to function with conventional working gloves. The panels' hardware has also been revised in comparison to previous versions in order to extract better performance, enabling their integration in projects requiring complex visualizations [29].

3.2.3 SIMATIC WinCC Unified PC

In addition to the possibility of sharing the screens with the UCP, Unified PC allows extensive scaling and includes capabilities such as logging and exchange of data with external databases or IT systems. Furthermore, the PC version of Unified offers “plant intelligence” capabilities such as [30]:

- **Calendar** - This option makes it possible to visualise a planning of the production operations. It is used to specify the operating hours of machinery, industrial facilities and to develop standardised templates for typical production shifts. Additionally, calendar-based report creation and process execution planning can also be done using this function.
- **Performance Insight** - It is a tool that gives the user access to a wide range of controls for KPI display and analysis, as well as the ability to edit and recalculate each KPI individually in accordance with International Organization for Standardization (ISO) standard 22400. The outcome of calculated KPI can then be reported.

- **Sequence** - This feature enables control and monitoring of recipe-controlled processes and the possibility to change sequences and parameters without modifying the PLC program.
- **Line Coordination** - Allows the automation of recipe and batch-controlled production processes. Based on S7-1500 controllers and WinCC Unified, production sequences involving connected machines in a manufacturing line can be coordinated, synchronised, and monitored.

WinCC Unified PC Runtime allows up to 128 S7 controller connections and 150 concurrent clients, meaning that it can be a SCADA solution for distributed systems (Figure 3.8). Due to its support of Microsoft Structured Query Language (SQL) Server, the PC system is able to manage the logging of up to 30000 tags, providing a foundation for in-depth historical data analysis and the tracing of processes over long time periods [31].

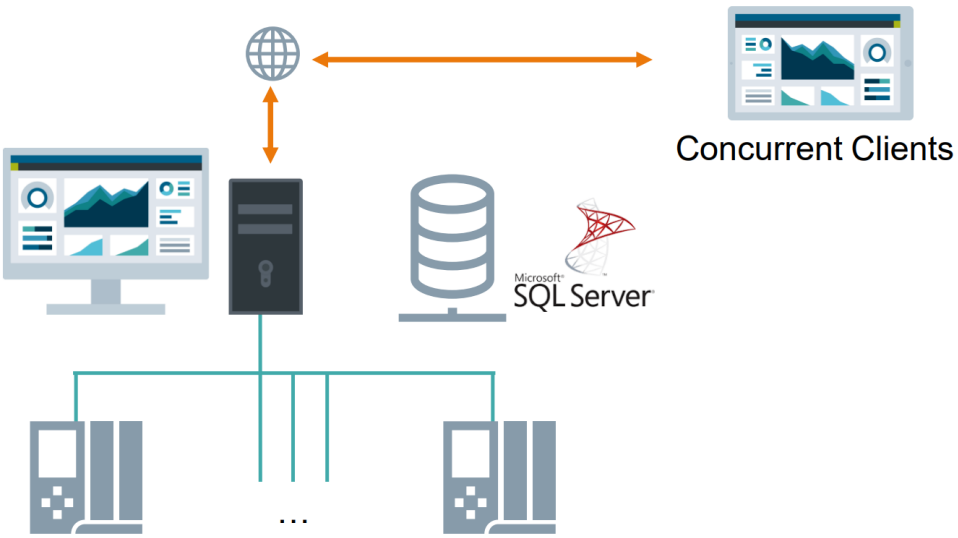


Figure 3.8: WinCC Unified PC architecture example [31].

3.2.4 SIMATIC HMI IWP10F Mobile WebClient

The IWP10F WebClient (Figure 3.9) presents a 10.1-inch screen and can function as a wired Unified Client, utilising the installed browser in kiosk mode to access the runtime of the previously mentioned Unified systems. To gain system access, the panel should be connected to a connection box.

In terms of hardware capabilities, the IWP10F is equipped with an emergency stop and an enabling button. For user interaction, additionally to the touchscreen, the panel comes with a handwheel and seven programmable function buttons [32].

Figure 3.9: SIMATIC HMI IWP10F Mobile WebClient [32].

3.3 Schneider Electric

Schneider Electric, as one of the most important players in the present industrial automation scene, provides a diverse range of HMI. Regarding the hardware, their solutions are divided into two major categories, Basic and Advanced HMI panels, with each group presenting smaller fractions with distinct properties.

The most basic versions, are designed to handle the most simple needs of a machine or production line, such as diagnosis, control, and modifying system settings on simple applications [33]. Inside the scope of the basic panels, the Harmony STO & STU (Figure 3.10a) are small and low-cost solutions with panel sizes ranging from 3.5 to 5.7 inches. The Harmony ST6 series varies in display size from 4 to 15 inches and comes from standard ready for web implementations, due to its integrated HTML5 browser. In addition to these specifications, depending on the model, they provide one or two Ethernet and communication ports [33].

If the project proves to be more demanding, the Advanced HMI panels can be implemented [34]. The Harmony GTU panels are IoT-ready modular HMI panels ranging in size from 7 to 19 inches and have WLAN features that enable wireless connectivity between the GTU and any capable device. The Harmony GTUX, which come in three screen sizes (7, 12, and 15 inches), are an option if the panel is going to operate in harsh conditions. They are certified for hazardous locations and have a wide operating temperature range of -30 °C to +70 °C, rated water and dust protection (IP66/67), resistance to chemical gas and high brightness display to be sunlight readable. To round out the advanced solutions, Magelis XBT GH (Figure 3.10b) is a hand-held panel with an emergency stop button, providing operator mobility and convenience of usage, as well as safety features including a dead-man switch [34].

(a) (b)

Figure 3.10: Schneider Electric HMI panels: (a) Harmony STO & STU [33] and (b) Magelis XBT GH [34].

For the engineering and development of the visualizations targeted to the panels, Schneider offers two distinct software. One of them is the EcoStruxure Operator Terminal Expert and it can be used for managing user interfaces supporting touchscreen configurations. The other is Vijeo Designer, which is intended to handle classic systems across the entire HMI range. Regarding SCADA systems, EcoStruxure Machine SCADA Expert is the software to use, making it possible to configure visualization projects using Harmony IPC [35].

3.4 Rockwell Automation

Similar to the previously stated manufacturer, Rockwell Automation offers a variety of HMI panels on the market, though some of them appear to be quite dated. The Series B PanelView Plus 7 Performance (Figure 3.11a) terminals are one of the most recent products on the market from this vendor. These have a display size range of 7 to 19 inches, with button variations being an option. The panel structure is similar to the previous generation, Series A. The major difference lays on the fact that Series B has updated hardware and runs Windows 10 IoT, allowing for modern implementations, in contrast to Windows Consumer Electronics (Windows CE). The software offers functions like a PDF viewer, a web browser and the possibility to perform remote desktop connection. Regarding the software for visualizations, FactoryTalk View Machine Edition is the one to use [36].

In reference to mobile solutions, the manufacturer provides the MobileView Tethered (Figure 3.11b) panel. These have a 10-inch display and software-assignable function keys. In addition to the IP65 junction box, the panel includes an emergency stop button that can be activated when both are connected via a cable [37].

(a) (b)

Figure 3.11: Rockwell Automation panels: (a) PanelView Plus 7 Performance [38] and (b) MobileView Tethered [37].

3.5 B&R

B&R is an ABB Group member and manufacturer of industrial automation components. In terms of HMI panels, they provide a number of alternatives for a variety of applications. If the project necessitates an HMI panel with embedded control capabilities, the Power Panel C-Series will meet these requirements, being intended for use in Computerized Numerical Control (CNC) applications, robotics, or conveyor systems [39]. There are two major groupings of solutions that are more relevant to this project. The Power Panel T-Series terminals, for example, can be used as a web browser or an interface to communicate with the system. There are numerous varieties within this category; for example, the Power Panel T80 (Figure 3.12a) is the most powerful for web-based visualizations, varying in size from 7.0 to 15.6 inches and supporting multiple Ethernet and USB connections [40]. The Mobile Panel 7100 series (Figure 3.12b) is an alternative for mobile solutions. The handheld device is cabled to a connection box, much like the previously mentioned for other manufacturers, and enable access to visualizations. They have integrated safety features, including an emergency button and a 3-step enable switch. The panel is available in 3.4, 7 or 10 inches, in portrait or landscape mode, and with a variety of button configurations for user interaction [41].

(a) (b)

Figure 3.12: B&R panels: (a) T80 [40] and (b) 7100 [41].

3.6 Conclusion

After analysing the available visualization systems in the market, from the most renowned manufacturers, it can be concluded that there are numerous options to consider when implementing a control or supervision solution. The choice between these products will depend heavily on the specific requirements of each project. It is worth noting that there is a noticeable trend towards web technologies for these systems, allowing fixed or mobile panels to have capabilities comparable to those of a computer.

Following a thorough analysis of each solution's characteristics, with a primary focus on Siemens technologies, due to the implementation requirements, it is concluded that the WinCC Unified system is the best choice for a project that aims to develop a modern interface with simple interaction, hardware abstraction, and content reusability.

Chapter 4

Technologies for Implementing a Wireless Concept

With the goal of transitioning from WinCC Advanced to Unified in mind, completely wireless access to the visualizations comes as a milestone, considering Unified only requires a browser and network connectivity to allow data interchange between the server and the client. However, it is crucial to note that this type of complete transition will require a thorough assessment of its viability in terms of cost and benefit, due to the fact that significant changes to the current solution, both in terms of hardware and software, will be required.

4.1 Contextualisation

Since the TIA Portal INTRALOG CS has been implemented numerous times and its operation has been refined over the years, major changes should be kept to a minimum in order to reduce system deployment time and the appearance of unanticipated problems during installation and operation. However, in order to offer complete wireless access to the conveyor system visualizations, some modifications to the base project would be required. The base project is currently using PROFINET as the protocol for Ethernet implementation on the shop floor. As a result, even mobile panels require physical connections to connection boxes through PROFINET in order to interface with the controllers. Since operators physically connect their HMI to a control point in a specific zone, it is possible to determine their location

without resorting to an external system. This way, it is feasible to allow access to visualizations that contain elements that they can physically see, improving safety.

The adjustments to the underlying project for a wireless concept can be divided into four major categories:

- **Industrial Wireless PROFINET** - Adapt the current Ethernet based implementation of PROFINET, replacing the existing connection boxes used by the mobile panels for Access Points (AP).
- **Real-Time Locating System (RTLS)** - Implementing a locating mechanism on the tablet to detect its precise position based on geofenced plant areas. This system can be disregarded in projects where this high precision location does not appear to be necessary, and the mere indication if a client is in a given zone is sufficient. This can be retrieved by the connection to the corresponding AP.
- **Unified Client** - In theory, any tablet that supports a standard browser and functions properly as a WinCC Unified client.
- **Unified Server** - To store and manage access to the visualizations by Unified Clients.

Following a brief introduction to Industrial Wireless Network (IWN) fundamental concepts, this chapter will cover the theoretical aspects related to each referred topic listed above, along with an assessment of the required components for implementation.

4.2 Industrial Wireless Network

Wireless communication improved in reliability and robustness with the ongoing development of technology, making it possible to be utilised in the field of industrial automation. In light of this, the concept of IWN developed to meet the demands of applications that would be rendered impossible or very challenging if a cable installation were to be employed. An IWN is utilised, for instance, to enable the use of portable control panels and the application of autonomous mobile robots and guided vehicles [42, 43].

Opposed to conventional wireless networks, IWN are designed to withstand extreme temperatures, dust, and vibration because industrial environments are typically known for their harsh conditions. To do this, they frequently employ specialised hardware and software, relying on protocols designed to be used in industrial automation projects to ensure reliable, secure and real-time communication [44]. Due to Ethernet's ability to connect all levels of a plant, which supports Industry 4.0's

system interconnectivity, it is now widely employed in industry. Since real-time communication is essential for industrial applications but is not enabled by Institute of Electrical and Electronics Engineers (IEEE) Ethernet standard, some manufacturers have begun to create their own proprietary Ethernet modifications for industrial systems, including PROFINET, EtherCAT, Modbus Transmission Control Protocol (TCP), and Powerlink [45].

Even though wireless communications in industrial facilities are often based on WLAN (IEEE 802.11), alternative technologies that interact via radio networks exist. Bluetooth (IEEE 802.15.1), Wireless HART (IEEE 802.15.4), Zigbee (IEEE 802.15.4), and WiMAX (IEEE 802.16) are a few examples [46].

Due to the fact that PROFINET is one of the most widely used Industrial Ethernet solutions worldwide [47], and because, as previously stated, the current form of the TIA Portal INTRALOG CS makes use of this protocol for the network backbone, its wireless implementation will be discussed in the following sections. Furthermore, since Ethernet-based protocols are similar, several concepts, such as network topology, are shared among them.

Finally, in order to deploy an Industrial Wireless Local Area Network (IWLAN), components such as AP and antennas are required, which can be produced by a variety of vendors, like Siemens or Phoenix Contact [48]. Given the insertion context of this project, some solutions from the first will be analysed.

4.3 Industrial Wireless PROFINET

The use of Wireless PROFINET in large facilities can reduce the cost of network implementation and maintenance by suppressing the need to run cables over long distances. Another application is the installation of sensors and actuators laid out in a mesh topology, which reduces the network's complexity and eases the challenges brought on by the need to route the cables. In both situations, Wireless PROFINET typically functions as an extension of the Ethernet network's backbone, allowing remote terminals to exchange I/O data with the primary control system (Figure 4.1). Although real-time data exchange and safety-critical messages are supported by Wireless PROFINET, its use does not render wired installations obsolete. The wireless channel is frequently a shared medium, so the management of the I/O exchange of data between the controller and the peripherals may require adaptation and the frequency at which data is updated may need to be adjusted. This installation typically occurs in specific situations and requires a careful evaluation in order to guarantee that interference or delays will be avoided [42].

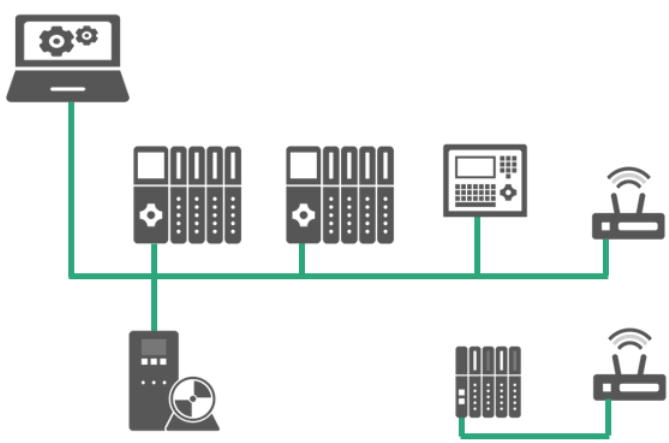


Figure 4.1: Basic architecture of a Wireless PROFINET network [42].

4.3.1 Overview of the Supported Wireless Standards

Any wireless standard from the IEEE 802.15.1 or 802.11 set can be used with PROFINET, because it is based on industry-standard Ethernet technology. These cover Bluetooth applications, as well as 2.4 and 5 GHz WLAN (Table 4.1). It's also crucial to note that common security mechanisms can be used to protect communications and provide safety messages like PROFIsafe [42].

Table 4.1: Wireless technologies supported by PROFINET [42].

Wireless technology	WLAN	Bluetooth
Protocol	IEEE 802.11 (a, b, g, h, n, ac)	IEEE 802.15.1
Bandwidth (Mbit/s)	11-1300	1-3
Average range (m)	Up to 92	Up to 100
Power consumption	High	Low
Working frequencies (GHz)	2.4 and 5	2.4
Nodes per network	32	8

Unlike the IEEE 802.11, concurrent standards like the 802.15.4 are oriented to low-rate wireless devices that often transmit small payload packets over a static network [47]. As presented in Table 4.1, Bluetooth, when compared to WLAN, has a much lower bandwidth and is only capable to work on one frequency band

that is usually shared with several wireless networks, being prone to interference due to congestion. Additionally, when deploying on large installations with multiple devices, the eight node restriction per network may be a constraining factor. Therefore, WLAN comes as the appropriate technology for implementation when the main goal is to have several clients that can flow throughout the industrial plant and exchange a substantial amount of data periodically with a main server.

In order to coordinate the transmission of data of multiple clients across a shared medium, the IEEE 802.11 standard defines two approaches [49, 46]:

- **Distributed Coordination Function (DCF)** - In a WLAN, all nodes use uncoordinated access to the wireless channel in accordance with the IEEE 802.11 standard, making it difficult to determine if a node is trying to access the medium while carrying critical data. Since DCF is a contention-based protocol, each station in a cell must compete with every other station in that cell for access to a given channel. When a station is prepared to send data, it waits for the communication channel to become available before sending data by listening to the medium for a certain period of time. The station waits until the end of the data transmission if the medium is already in use. This mechanism does not ensure that a certain amount of data will be transferred within a predetermined window of time, as a result, it is best suited for asynchronous data transmission.

- **Point Coordination Function (PCF)** - It is the second mechanism defined on the 802.11 standard for media control access, recurrently implemented in applications that require continuous flow of data. Although this method is not mandatory, it can be used to avoid some of the drawbacks of the DCF mechanism. This technique is based on one or several AP acting as central administrators of the network, so not all nodes have equal rights. The AP assigns time slots to the clients, making it possible to reserve a certain frequency for the transmission of data without disruptions. The implementation of this mechanism is preferable for synchronous data transmission.

4.3.2 Characteristics for Industrial Deployment

PROFINET doesn't need specific networking hardware because, as stated previously, it is based on standard Ethernet. A PROFINET network can integrate AP that are commercially available for purchase. These components allow wireless transmission of data between capable devices or networks, by means of a communication protocol. Despite not being required, dedicated industrial AP are widespread in industrial environments due to the demand for reliable, robust equipment and the need of minimal cycle-times, allowing real-time communications [44].

In order to allow the transmission and reception of data, via radio frequency energy, each AP has at least one antenna, these exist primarily in two forms [42, 46]:

1. **Omnidirectional** - Allow the signal to be widespread across 360 degrees and usually are implemented in situations where one AP must cover the maximum possible area.

2. **Line-of-sight** - Can be used to radiate the signal to a particular target, leading to less signal degradation amongst connected nodes of a certain direction.

Industrial hardware variations may contain special characteristics developed for industrial automation projects that demand strict specifications, in contrast to commercially available AP [42].

Listed below are some key concepts that emerge as a result of an IWLAN implementation [42, 46]:

- **Bandwidth reservation** - Makes it possible to reserve a frequency band for critical data transmission while lower priority data uses another frequency band.

- **Redundancy** - It is possible to improve network reliability by adding multiple paths that allow data transmission between the same users. The Mesh network topology is one of the most powerful solutions for achieving high communication reliability due to the high level of redundancy of communication paths. Redundancy can also be achieved by using the Parallel Redundancy Protocol (PRP), which is specifically designed for cable-based networks and is defined in IEC 62439 Part 3. PRP creates two completely separate networks. In the event of a disruption in one network, data transmission continues seamlessly through the redundant network. PRP-equipped devices have at least two Ethernet interfaces connected to different networks. To achieve the same concept and provide the same redundancy benefits but in wireless networks with paths of different speeds, Siemens developed iPCF. It enables redundant communication over two WLAN routes, even for mobile applications. If roaming is delayed or there is interference, communication continues uninterrupted through the secondary path [50].

- **Radiating or RCoax cables** - Are meant to behave as antennas to broadcast radio waves. They are commonly employed in wireless communication systems, where they can replace standard antenna configurations and provide a flexible and adjustable solution. Radiating cables are normally coaxial cables with slits in the shielding material to allow radio waves to radiate out of the cable, as illustrated in Figure 4.2. For instance, it might proceed exactly along the

path of an overhead monorail conveyor, allowing for reliable illumination of the parts of the wireless cell that are hard to reach in complex structures.

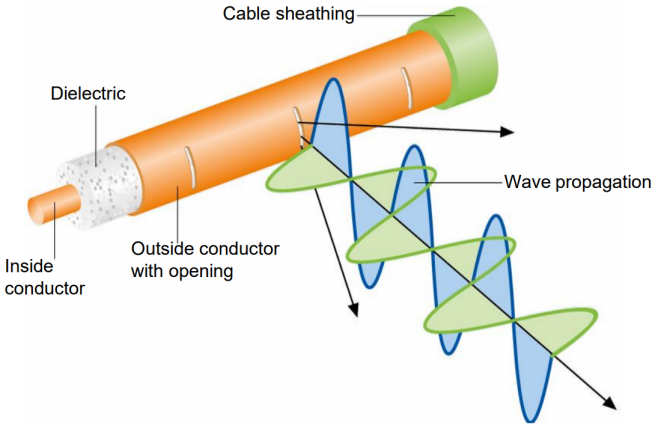


Figure 4.2: Radiating cable structure [46].

- **Multiple antennas** - The addition of multiple antennas to an AP increases the communication's robustness. Multiple reflections can occur before a signal reaches its destination, degrading, attenuating, and distorting it. The AP can retrieve various forms of the transmitted signal by using multiple antennas. Furthermore, the implementation of some wireless topologies may require the usage of several channels, leading to the necessity of implementation of multiple antennas on one AP.

- **industrial Point Coordination Function (iPCF)** - This iFeature is a substitute to the PCF defined by IEEE 802.11, with the goal of satisfying industrial applications requirements. It allows clients to swiftly switch between wireless cells with log off and log in processes that meet the real-time communication prerequisites. Access points in iPCF regularly poll clients in their cell at short intervals, enabling them to log on and transmit data as needed, but only after receiving permission from the AP (Figure 4.3). The AP polling frequency can be adjusted to reduce response times to about 2 ms per node and guarantee a response time of under 10 ms with four clients. Non-time-critical messages are transmitted when available cycle time is free. The scanning of a node is observable by all other nodes within the cell, enabling a client to assess the quality of their wireless connection to the AP even when not actively communicating with it. The short polling cycle ensures clients quickly detect a lost connection and switch to an alternative AP if necessary.

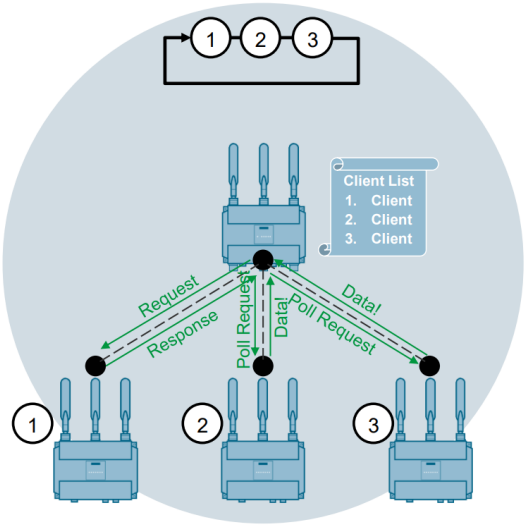


Figure 4.3: iPCF architecture [46].

- **industrial Point Coordination Function-Management Channel (iPCF-MC)** - The development of the iFeature named iPCF-MC aimed to bring it's advantages to nodes that move freely and communicate regardless of their use of an RCoax cable or directional antenna. With iPCF-MC, clients can still search for available AP if they receive iPCF requests from an AP and if their connection is free from interference, allowing for quick AP changes if needed. iPCF-MC does not have handover times dependent on the number of radio channels used, unlike iPCF. To utilise iPCF-MC, a dual AP with two radio interfaces is required - one serving as a management channel transmitting administrative information, and the other solely transmitting user data, as shown in Figure 4.4.

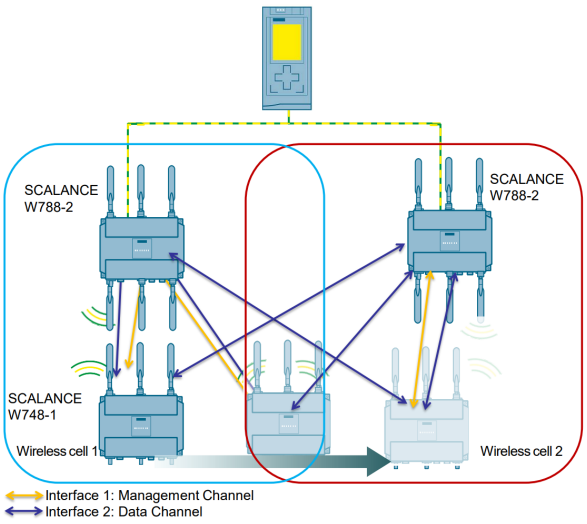


Figure 4.4: iPCF-MC architecture [46].

- **Roaming** - Roaming refers to the movement of nodes from one AP to another. In IWLAN covering larger areas, one AP radio range is often insufficient, requiring multiple AP for complete coverage. Clients should be able to move freely within their cell and into neighboring cells. When AP have overlapping radio coverage, clients should be able to switch without disrupting the network connection. For smooth roaming, overlapping wireless cells must communicate on different channels. A client located in the overlapped region would have a permanent reception impairment if all wireless cells were to use the same channel. Roaming per IEEE 802.11 standards can lead to a delay of hundreds of milliseconds as the process of detecting a client's exit from the previous cell and connecting to the new one takes place. To achieve faster roaming and deterministic data traffic, AP and clients that support iPCF should be implemented.

4.3.3 Wireless Network Topologies

To deploy wireless data exchange between multiple nodes, a specific topology that specifies the network structure must be established. After this introduction, some of the most important industrial wireless topologies will be discussed. As previously stated, PROFINET supports WLAN and Bluetooth; however, while both use wireless data transmission, their operating principles are not the same. One significant distinction is that Bluetooth does not employ AP, hence the Infrastructure Network Topology and Wireless Distribution System (WDS) are only applicable to WLAN.

Point to Point

Point to Point implies a dedicated connection between the two nodes in order to establish wireless communication between them (Figure 4.5). The wireless link's maximum bandwidth is available because the channel is not a shared medium. For instance, a wired LAN connection could be replaced using this method [42].

Figure 4.5: Point to Point topology [42].

Point to Multipoint

A network architecture style known as Point to Multipoint involves connecting a central node to multiple other nodes (Figure 4.6). Data can be sent and received by the central node to and from the multipoints. Each multipoint in this topology can

communicate with the point but not with other multipoints directly. Several computers or HMI can be connected to a PROFINET network using this topology [42].

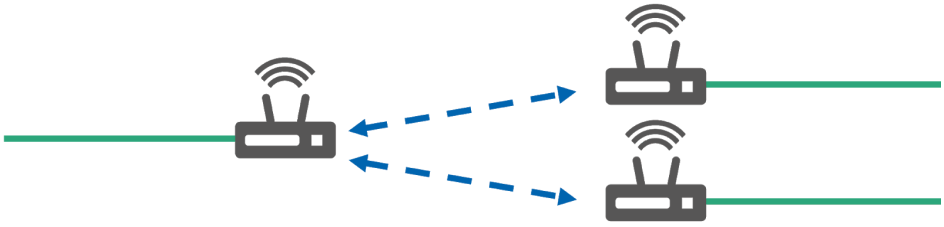


Figure 4.6: Point to Multipoint topology [42].

Mesh

This topology is a type of architecture in which numerous nodes are wirelessly connected to form a mesh network (Figure 4.7). In this topology, each node can function as both a client and a router, forwarding data packets to other nodes in the network. This provides multiple paths for data to move from one node to another, increasing network reliability through redundancy, allowing data to be transmitted from one node to another, even if one or more nodes fail. This topology is frequently used to cover a broad area when cable connections would be difficult or extremely expensive to establish. It is crucial to note that mesh networking topologies, which are ubiquitous in IEEE 802.15.4, are not well supported by IEEE 802.11 standards [42, 51].

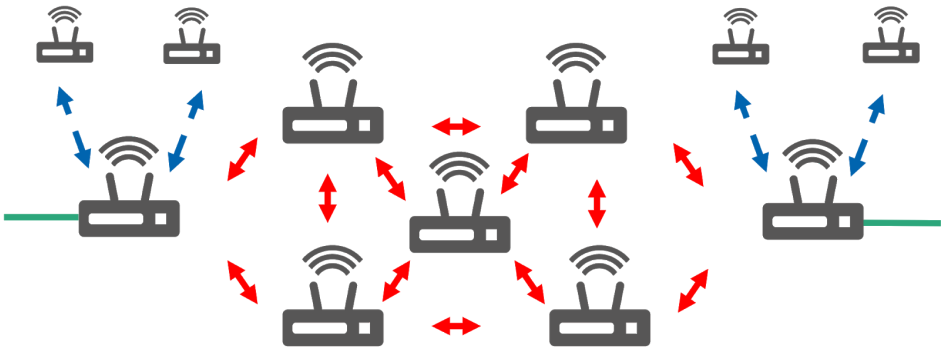


Figure 4.7: Mesh topology [42].

Infrastructure

An infrastructure topology is used to extend a wired LAN to incorporate wireless devices. This architecture allows devices to communicate with the wired LAN via an AP, which functions as a bridge between the wired and wireless network.

A standalone network is formed when one AP is only used to allow and coordinate the communication of several clients, so all of them must be within the AP range, as represented in Figure 4.8 [46].

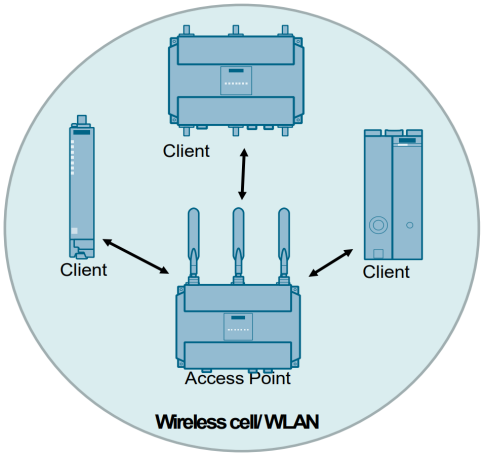


Figure 4.8: Standalone network topology [46].

When one or more AP are used to connect to a cable-based network, the concept of mixed networks emerges. Since multiple AP can be interconnected, typically via an Industrial Ethernet backbone, several wireless cells will be formed. If these cells are slightly overlapped, the client will be able to roam (represented as a dotted arrow in Figure 4.9) between them without losing connection. If the AP uses the same frequency, interference may occur within the overlapping range of the radio cells. A multi-channel configuration should be employed to eliminate this issue and expedite the roaming process, therefore enhancing system performance. The network architecture remains the same, but the second AP will connect to its clients over a different channel, for instance Channel B.

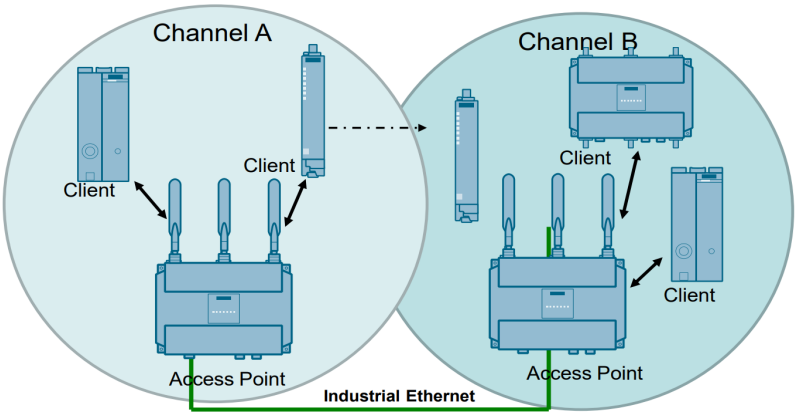


Figure 4.9: Mixed network topology with multi-channel configuration [46].

To create redundancy and improve system reliability, AP with two wireless interfaces and the flexibility to transmit on several frequencies could be employed (Figure 4.10). In this circumstance, one wireless interface is predominantly used; but, if a fault is detected, the communication to clients can be shifted to the other interface, keeping the connection alive [46].

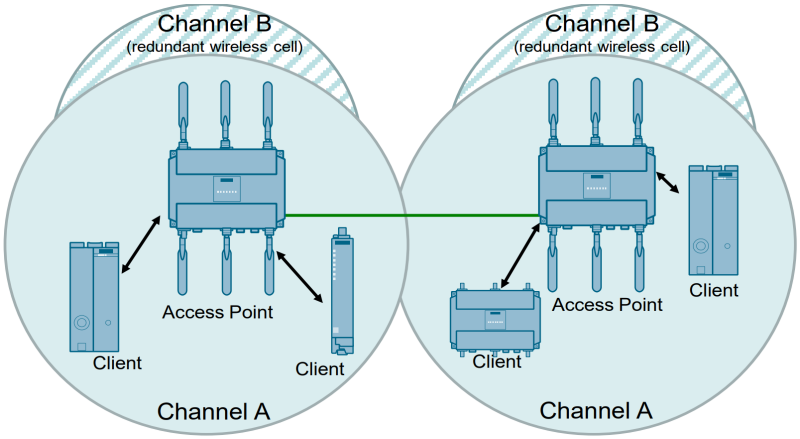


Figure 4.10: Mixed network topology with multi-channel configuration and redundancy [46].

Wireless Distribution System

A WDS can be used in circumstances when multiple AP need to communicate with one another, such as when extending wireless coverage or establishing a wireless backbone (Figure 4.11). The architecture is comparable to the previous multi-channel configuration. The key distinction is that the connection between AP is made over a wireless network rather than Industrial Ethernet. They must communicate on the same channel in order to establish connection. When multiple connections are formed on the same frequency, the data rate of each AP is reduced since the bandwidth is shared.

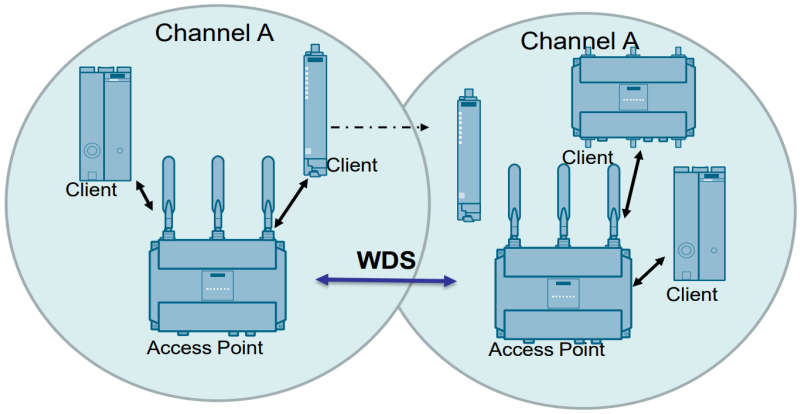


Figure 4.11: Wireless Distribution System topology basic configuration [46].

To establish redundancy on a WDS, the AP should communicate with each other using a second set of antennas rather than the primary frequency, as shown in Figure 4.12 [46].

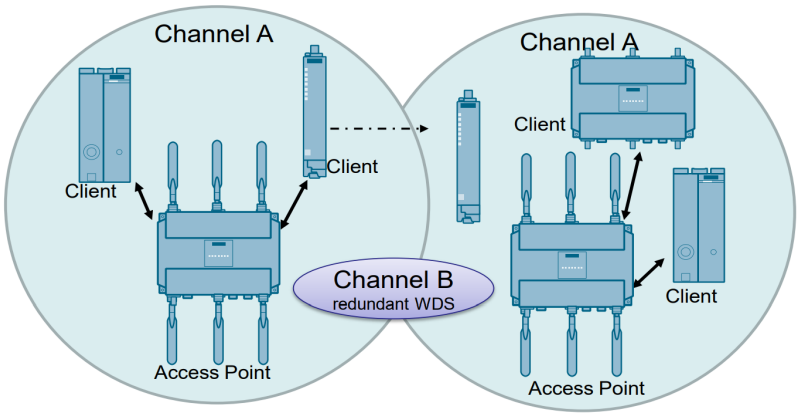


Figure 4.12: Wireless Distribution System topology with redundancy [46].

4.3.4 SCALANCE W

The Siemens SCALANCE W series of IWLAN devices are designed for use in industrial environments. These products are part of Siemens' networking solution portfolio, which is designed to provide reliable, real-time and secure communication between machines, equipment, and other devices in an industrial context. The SCALANCE W series includes AP, client modules, and antennas. These products are constructed to withstand harsh environmental conditions such as, high temperatures, moisture, and vibration. The components support several industrial protocols, such as PROFINET and Modbus TCP, and can be integrated into existing industrial networks.

There are mainly three product classes for different solutions regarding AP and client modules: basic, demanding, and high performance. The decision between them will be determined by the degree of complexity, the features provided, and the performance and reliability required for the industrial application. The high performance variants (SCALANCE WAM766-1) have one significant distinguishing feature: support for IEEE 802.11ax (Wi-Fi 6) for applications that require higher bandwidths like live video transmission [52, 53].

One key factor to think about when deploying the IWLAN is whether the AP should be installed within or outside the control cabinet. Some products (Figures 4.13a and 4.13b) only support the first instance, whereas others (Figure 4.13c) can be deployed outside the control cabinet.

(a) (b) (c)

Figure 4.13: SCALANCE W AP: (a) W761-1 RJ45 [54], (b) W774-1 RJ45 [55] and (c) W778-1 M12 [56].

Table 4.2 provides a comparison of three AP with different features:

Table 4.2: SCALANCE W Access Point comparison (prices based on Siemens Industry Mall).

Access Point	W761-1 RJ45 [54]	W774-1 RJ45 [55]	W778-1 M12 [56]
Article Number	6GK5761-1FC00-0AA0	6GK5774-1FX00-0AA0	6GK5778-1GY00-0AA0
Number of electrical connections for external antenna(s)	1	2	2
Max. transfer rate with WLAN (Mbps)	150	300	300
Support of iFeatures	No	Yes	Yes
Installation	Inside cabinet	Inside cabinet	Inside/Outside cabinet
Price (€)	918	1200	1428

Antennas (Figure 4.14), in addition to AP, are crucial elements for enabling wireless communication. Various types are available, each serving specific purposes. Among these are omnidirectional antennas, which offer extensive radio coverage in all directions, directional antennas that provide precise and focused radio coverage, and sector or wide-angle antennas that efficiently cover specific zones. For an application like the one proposed with this project, where separate zones would be defined in order to allow control over what elements an operator should have access through the HMI, omnidirectional or sector antennas are the more appropriate components to fulfil these requirements. The decision between one or another would lay primarily

on the architecture of the industrial plant. With this in mind, some antennas that fit these specifications and are compatible with the AP from Table 4.2, are presented in Table 4.3.

(a) (b) (c)

Figure 4.14: SCALANCE W IWLAN antennas: (a) ANT795-6DC [57], (b) ANT795-6MP [58] and (c) ANT795-6MN [59].

Table 4.3: SCALANCE W antennas comparison (prices based on Siemens Industry Mall).

Antenna	ANT795-6DC [57]	ANT795-6MP [58]	ANT795-6MN [59]
Article Number	6GK5795-6DC00-0AA0	6GK5795-6MP00-0AA0	6GK5795-6MN10-0AA6
Type of antenna	Sector	Omnidirectional	Omnidirectional
Frequency range (GHz)	2.4/5	2.4/5	2.4/5
Mounting	Mast and wall	Mast and wall	Roof and wall
Price (€)	252	252	240

4.4 Real-Time Locating System

As per ISO/IEC 24730, RTLS are wireless systems that are capable of determining the location of an item within a defined area, in real-time or close to it. The position is calculated using measurements of the radio link's physical properties. According to the standard, RTLS can be conceptually separated into four groups in terms of asset location [60]:

1. **Terrestrial** - This method uses terrestrial mounted receivers across a large area, such as cell phone towers, to track assets. The accuracy is approximately 200 meters.

2. **Satellite-based** - Satellite technology is employed to locate assets and requires a clear line-of-sight to achieve an accuracy of approximately 10 meters.

3. **Controlled area** - This approach is used for tracking assets within controlled areas, such as warehouses or airports. The area of interest is equipped with instruments, resulting in accuracy within 3 meters.

4. **Confined area** - This method is used for tracking assets within a more confined area. The area of interest is instrumented, and the accuracy is within tens of centimetres.

The implementation of the concept of a RTLS can be achieved by utilising multiple technologies, such as WLAN, Global Navigation Satellite System (GNSS), Zigbee, and Bluetooth. Every technology has benefits and drawbacks; for example, GNSS works well outdoors but is typically inaccurate indoors. In outdoor environments where there is clear line-of-sight between objects and the sky, RTLS can utilise global navigation satellite systems, such as Global Positioning System (GPS), Galileo or GLONASS to provide real-time positioning data. Regarding Wi-Fi-based locating systems, they present the advantage of utilising some existing AP of the network but fall short of providing the location accuracy needed for many industrial applications.

The attainable accuracy of a RTLS is heavily influenced by the radio technology employed. With the introduction of Ultra Wide-Band (UWB) chips, industrial indoor applications may now attain location precision of less than one metre. As a result, several manufacturers of RTLS, relevant to the industrial sector, rely on UWB rather than less accurate systems based on infrared, Wi-Fi, or Bluetooth [61, 62]. Currently on the market, systems from SEWIO [63], KINEXON [64], Inpixon [65] and Siemens [66] can be found employing UWB technologies for real time location on industrial environments. These systems rely on established hardware, often known as nodes, to supply positional information to smaller portable nodes linked to items of interest. These mobile nodes are generally referred to as tags or transponders, while fixed nodes are referred to as anchor points, wall units, or gateways. The first are typically small, battery-operated devices that can be attached to, or integrated into, any object and send out some kind of signal on a regular basis, which is picked up by one or more anchors. After receiving the data, the anchor can send it to another element of the network, usually a central controller or server running a specific software, in order to compute the data retrieved from the position of the tag [61, 62].

After evaluating the working principle and components available on the market, provided by the manufacturers stated previously, it is possible to conclude that they are all very similar, when it comes to the deployment of UWB. Essentially

the solutions lay on the combination of transponders, anchors and software to retrieve positions. As a result, on the following subsections the Siemens RTLS will be presented with the objective of providing a representation of these types of solutions.

4.4.1 Siemens RTLS

The Siemens RTLS employs, separately or in combination, 2.4 GHz and UWB technology. According to Siemens, 2.4 GHz can detect a tagged object within 3 metres, so this solution can be used in circumstances when maximum accuracy is not required, such as tracking a container. If the project demands a solution that maximises accuracy, UWB can be used, resulting in a location accuracy of tens of centimetres. This solution might be used in a circumstance when the validation of installation of a certain component on a specific place is required [67].

Applications for Siemens RTLS include asset or personnel tracking and inventory management. To increase efficiency, safety, and productivity, they are frequently used in logistics, manufacturing, and healthcare.

Siemens RTLS offers several important characteristics, such as [67]:

- **Real-time tracking** - RTLS allows users to observe an object or person's current location and movement on a map or dashboard.
- **Efficiency gain** - By providing real-time information on the movement and placement of assets like trucks, equipment, or inventories, organisations may benefit from the implementation of RTLS by minimising downtime, increasing asset usage, and streamlining procedures.
- **Enhanced safety** - Contributes to safety improvements by sending out real-time notifications when a person or object leaves a specified region or when there is a chance of a collision.
- **Scalability** - RTLS are expandable to meet the requirements of various businesses and plant configurations.
- **Customisation** - RTLS can be altered to suit the particular requirements of a business, for example, by linking them with other systems or data sources, or by including more sensors or transponders to cover a broader area.

The locating system is composed of three major components: transponders, gateways and software. Figure 4.15 presents the architecture of this solution based on the enunciated components. Represented by the orange squares, the transponders can be attached to any relevant asset or person in order to exchange localisation signals with the gateways (blue squares). After receiving the data, the gateways route the information via an Ethernet connection to the Locating Manager software.

Following that, after completing the necessary calculations, the software performs information mapping for higher-level systems like an ERP or MES.

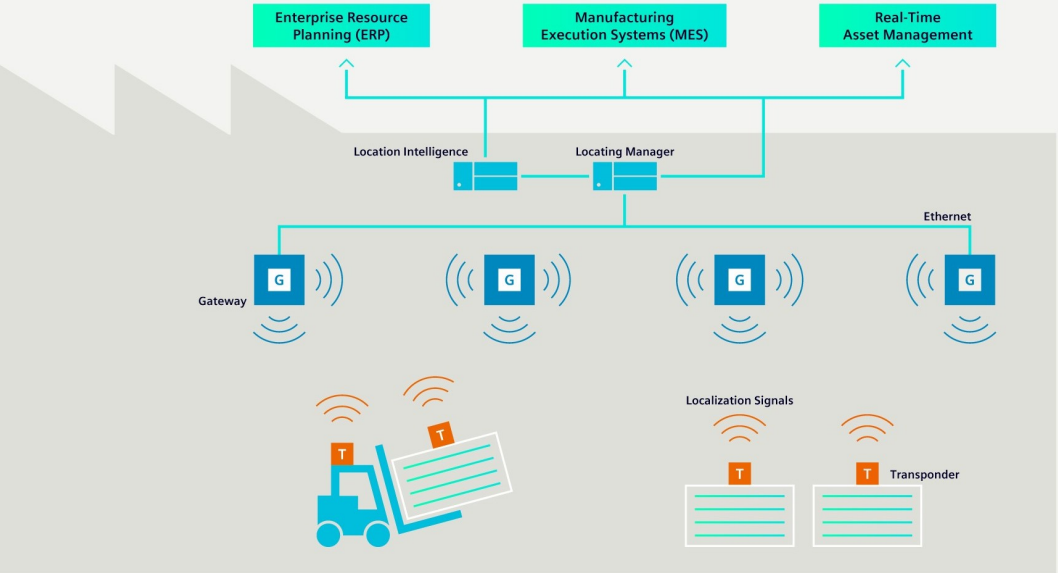


Figure 4.15: Real-Time Locating System architecture [67].

Transponders

Transponders can be mounted to several elements like workpieces, robots and vehicles. SIMATIC RTLS provides a wide range of transponders for a variety of purposes, that can send a signal at predetermined intervals. All transponders are active and feature a battery or rechargeable battery that may be changed or recharged. As previously indicated, these devices support, depending on the version, technologies such as UWB, Phase Shift Keying (PSK), and Chirp Spread Spectrum (CSS) in order to communicate with gateways. In terms of physical characteristics and functionalities, the transponders have a robust construction and are resistant to environmental factors. Furthermore, some of them have e-link displays, which allow for the simple visualization of data such as optical codes, text, and images. Some variations also have interfaces and can send location data directly to the control system [68].

Gateways

The architecture of the SIMATIC RTLS locating platform is made up of gateways, also known as infrastructure devices, which are located in the facilities or on the premises where transponder locating is to take place. The gateways act as fixed reference points for real-time location, depending on the surroundings and site-specific characteristics. The gateways may be deployed via an Ethernet connection, or if a more flexible solution is needed, they can also be implemented entirely wirelessly. They are able to communicate in two different ways. The first possibility is direct

connection with the transponder to determine distances or exchange data, while the second method uses a mesh network for wireless communication between all gateways. After the gateways gather data, it is sent to the Locating Manager, where it is turned into real-time location data [67, 69].

Communication can take place through one or more technologies or signal modulation techniques, these being:

- **UWB** - It is a technology that leverages a broad radio frequency spectrum to transmit data at high speed. This allows for greater data transmission compared to conventional technologies. UWB operates in the frequency range of 3.1 to 10.6 GHz [70].
- **PSK** - The carrier's phase varies in discrete levels in response to the input digital signal, while its amplitude remains constant [71].
- **CSS** - Is a spread spectrum communication technology used in wireless communications systems to send data across a wide frequency band. It employs a linear frequency modulation approach, in which the frequency of the transmitted signal is continually changed over time.

The RTLS solution is ready to deploy both indoors and outdoors by combining two of the aforementioned technologies, enhancing the system's flexibility. Depending on the features sought for the implementation project, three SIMATIC RTLS families arise to meet the needs: RTLS4000, RTLS4300, and RTLS4400. The technologies and systems that utilise them, as well as the main characteristics that they inherit, are provided in Table 4.4.

Two locating principles can be used to determine the location of the transponder. One of these is Two Way Ranging (TWR), in which the transponder queries which gateways are in range and then looks for the closest one based on the answers. This bidirectional communication enables for the exchange of signals that are passed to the Locating Manager in order for the transponder position to be calculated. The other way is Time Difference of Arrival (TDOA), in which the transponder sends a blink signal that the infrastructure devices detect. The time of detection is recorded and transmitted to the Locating Manager via the gateways. The position of the transponder can be calculated using the propagation time data [67, 69].

Table 4.4: SIMATIC RTLS families comparison [69].

Technology	UWB	PSK	CSS
SIMATIC RTLS	RTLS4000 RTLS4300	RTLS4000	RTLS4300 RTLS4400
Frequency range (MHz)	3993.6-6489.6	2410-2480	2400-2480
Protocol	IEEE 802.15.4-2015 HRP UWB	IEEE 802.15.4	IEEE 802.15.4a CSS
Max. range indoors (m)	30	50	100
Max. range outdoors (m)	30	500	1000
Max. locating accuracy (m)	0.2	1	1.5
Max. transfer rate (Kbps)	6800	1000	1000

Software

The SIMATIC RTLS Locating Manager is the software that calculates and sends the real-time positions of individual transponders to higher-level systems [72].

Geofences, which are virtual perimeters set up around a certain location or area [73], can be created using the Locating Manager. These can be used in conjunction with RTLS systems to generate automated alerts or trigger actions when an object or person enters or exits a specific area. A geofence around a building site, for example, may be set up to inform workers when they reach a hazardous area or to trigger alarms if unauthorised persons enter the site.

Another noteworthy feature of this software is the ability to integrate it with SIMATIC S7 controllers via the LRTLS block library (Figure 4.16). This library enables users to change the predefined themes of the ePaper transponders, perform simple geofencing tasks using the received transponder coordinates, and handle transponder capabilities such as turning a Light Emitting Diode (LED) ON or OFF [74].

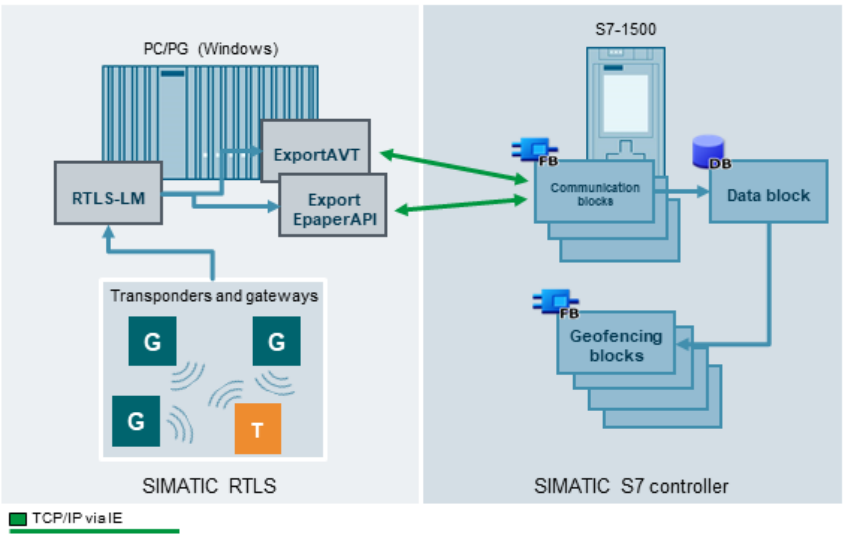


Figure 4.16: Data exchange between S7 controller and RTLS Locating Manager via dedicated FB [74].

4.4.2 Evaluation of the Integration of a RTLS into this Project

A transponder could be attached to a tablet in order to track the location of the operator using it as a Unified Client, and grant control to the elements that he can see. In addition, a few gateways should be placed throughout the zones that contain the elements that need to be controlled in order to detect the transponder's location. Additionally, in order to run the Locating Manager and Unified, an IPC would have to be set up as a server.

According to Siemens [69], the recommended wireless technology combination for industrial applications is UWB and PSK, so based on the information provided in Table 4.4, the RTLS4030G Gateway (Figure 4.17a) and the RTLS4030T Transponder (Figure 4.17b) come as the appropriate hardware for the implementation of a real-time locating system in this particular solution.

(a) (b)

Figure 4.17: RTLS hardware: (a) RTLS4030G and (b) RTLS4030T.

Table 4.5 shows the indicative prices of components proposed for such projects, according to data obtained from Siemens' Industrial Mall.

Table 4.5: SIMATIC RTLS components price.

Product	Article Number	Price (€)
Locating Manager Basic for 10 transponders	6GT2780-0DA00	1 200
Gateway RTLS4030G	6GT2701-5DB03	1 644
Transponder RTLS4030T	6GT2700-3DA03	150

Despite the possibility that this type of technology will be necessary in projects requiring a high degree of certainty in the location of the operator controlling the visualizations, it is crucial to weigh the benefits and drawbacks of its implementation. Listed below are some of the advantages that the RTLS would bring to the project in work:

- RTLS has a block library for integration with S7 controllers, facilitating the integration of this technology.
- Introducing geofences would eliminate the necessity of connecting a cable to a connection box for panel-based control of elements. This would promote smoother operator mobility across the shop floor. Upon the operator's entry into the geofenced area, automatic control over the physically visible elements within it would be automatically granted.

Regarding disadvantages, some of them are:

- Added costs and complexity to the installation.
- The batteries of the transponders need to be replaced (rated for 6 months using UWB).
- In case of an RTLS failure the access to the visualization system would be limited.

4.5 WinCC Unified Clients

The WinCC Unified Client is in charge of providing a graphical interface for controlling and monitoring the machine or plant. Users can access the WinCC Unified Runtime through the client via web mechanisms on the station itself or via a network. There are two kinds of Unified Clients [21]:

1. **WinCC Unified Client** - provides operators with the ability to control and monitor operations.

2. **WinCC Unified Client Monitor** - enables monitoring only (view-only) capabilities.

Two key aspects that must be considered while implementing the Unified Clients are licensing and hardware.

Regarding licensing, depending on whether the Unified Server is running on a panel or a computer, the Unified Client requires a panel-based or server-based licence of the Unified Runtime. Each Panel comes with a Unified Client (Operate) licence. With each Unified PC Runtime licence, two Unified Clients and one Client Monitor licence are supplied, allowing for local operation on the runtime station, as well as operation utilising an additional independent client. The monitor licence also allows for local or remote monitoring using another client. The client licenses for Unified are stored on either the Unified Runtime Station or the UCP and are classified based on the number of concurrent web clients accessing the visualizations. To increase the number of clients, additional license packages can be obtained, available with 1, 3, 10, 30 or 100 Unified Operate Clients or 1, 3 or 10 Unified Monitor Clients [21].

As a point of reference, and given that typically there is no need for more than three concurrent clients, the pricing details for different licence types are provided in Table 4.6.

Table 4.6: SIMATIC WinCC Unified Client licenses price (based on Siemens Industry Mall).

Product	Article Number	Price (€)
3 Operate Clients	6AV2157-3JW00-0LB0	6 968
3 Monitor Clients	6AV2157-3JM00-0LB0	2 280

In terms of hardware, the most appropriate solution for mobile deployment would be specially developed by Siemens to be used as a Unified Client but, until the moment of writing this document, there is no information available to the public about the launching of this component. However, according to the documentation and given the flexibility offered by the system, any type of tablet with a modern browser and a display size of roughly 10 inches (to ease interaction between the operator and the panel) can be a candidate for implementation. It is crucial to highlight that, while standard tablets could be used to access the visualizations, the majority of projects will require industrial variations capable of coping with harsh environments and low/high temperatures. Concerning the operating system, either Android or Windows can be utilised because both can run Google Chrome, the Unified documentation's suggested web browser. Due to the operating system ease of manipulation, long term security updates and existing Siemens software, Windows variants could be considered more adequate for this project.

There are several manufacturers offering products that fulfil the basic characteristics stated. Some of them are Advantech, Honeywell (Figure 4.18a), Getac and Siemens (Figure 4.18b).

(a) (b)

Figure 4.18: Industrial tablets: (a) Honeywell RT10W [75] and (b) Siemens ITP1000 [76].

Table 4.7 presents some models of the aforementioned brands, as well as some of the main characteristics and relevant comparative topics. It is crucial to note that several configurations of hardware and operating systems exist within each model, resulting in the existence of multiple variants with different prices.

Table 4.7: Industrial tablets comparison [69].

Manufacturer	Advantech	Honeywell	Getac	Siemens
Model	AIM-68 [77]	RT10W [75]	UX10 [78]	ITP1000 [76]
Operating System	Windows IoT	Windows IoT	Windows 10	Windows 10
Display Size (in)	10.1	10.1	10.1	10.1
Processor	Intel Atom	Intel Pentium	Intel Core i5	Intel Core i5
Memory RAM/ROM (Gb)	4/64	8/128	8/256	8/256
Operational temperature (°C)	-10 to 50	-10 to 50	-29 to 63	5 to 45
Price (€)	1 765 [79]	2 716 [80]	2 600 [81]	5 031 [82]

4.6 WinCC Unified Server

To store and manage access to the visualizations by Unified Clients, a Unified Server must be implemented, either as an UCP or IPC. The former is a simpler and cost-effective option, while the latter is a more expensive but high-performing solution, especially necessary if the project demands more than three clients accessing the server simultaneously and if an RTLS needs to be implemented. The IPC can be used to run both the Unified Runtime and the RTLS Locating Manager.

Choosing an UCP will only depend on the desired display size because all share the same hardware and interfaces. The MTP700 (Figure 4.19), 7-inch variant, could be utilised to cut expenses if using the panel as a Unified Server is the primary goal of implementation. The MTP1000 (10-inch version), although more expensive, might be more appropriate for a friendlier experience in terms of display size. Table 4.8 provides the price associated with each panel.

Figure 4.19: SIMATIC UCP MTP700 [83].

Table 4.8: SIMATIC UCP prices (based on Siemens Industry Mall).

Product	Article Number	Price (€)
MTP700	6AV2128-3GB06-0AX1	1 092
MTP1000	6AV2128-3KB06-0AX1	1 804

If the solution demands more powerful/complete control capabilities, an IPC is the best option. An entry-level SIMATIC IPC127E (Figure 4.20a) should be sufficient for running the Unified system and, eventually, the Locating Manager for the RTLS. Whenever a more powerful computer is required, the SIMATIC IPC227G (Figure 4.20b) could be suggested [84]. The indicative prices of the IPC are presented in Table 4.9.

(a) (b)

Figure 4.20: IPC: (a) SIMATIC IPC127E [85] and (b) SIMATIC IPC227G [86].

Table 4.9: SIMATIC IPC prices (based on Siemens Industry Mall).

Product	Article Number	Price (€)
IPC127E	6AG4021-0AB11-1CA0	1 636
IPC227G	6ES7647-8CF33-2AB3	2 725

4.7 Conclusion

With the study of technologies that allow for a completely wireless implementation of the visualization system, it was possible to determine that, when using PROFINET, the implementation of WLAN will be the most suitable for providing access of the visualizations to Unified Clients who may be distributed throughout the plant. However, it is important to note that the simple integration of WLAN into this system does not accurately solve one of the main challenges of this project, which is to determine the position of operators in order to restrict access to the system, based on their location. To solve this issue, it would be necessary to integrate an RTLS.

Although everything indicates that the implementation of a wireless network and a locating system appear to be a viable option, it should be emphasised that both will considerably increase the price of the solution to be presented to potential customers. This, in addition to the inherent challenges of restructuring the INTRALOG CS standard to accommodate these new technologies, and the fact that they do not have as proven reliability and robustness as wired installations, leads to the possibility that the path towards a completely wireless solution may not be followed in the near future, considering that the transition to WinCC Unified alone is already a complex process.

Chapter 5

Conceptualisation

The primary objective of this chapter is to delve into the conceptualisation of an innovative visualization interface for the TIA INTRALOG CS standard, utilising the process visualization system WinCC Unified. The chapter commences with an evaluation of the proposed hardware system architecture, where the rationale behind the chosen components is justified, drawing upon the information presented in earlier chapters. Subsequently, the chapter proceeds to outline the detailed conceptualisation of the interface design.

5.1 System Architecture

As stated in earlier sections of this thesis, the industrial sector is experiencing an increasing need for effective and smart automation solutions, resulting in a notable expansion of industrial wireless deployments.

Despite the appeal of deploying the visualization interface through a wireless network, locating system and a tablet, there are significant concerns that must be addressed. This approach offers enhanced flexibility and mobility in industrial environments but comes with notable drawbacks, such as increased installation costs and the complexity of integrating these new technologies into the existing INTRALOG standard. Moreover, considering the viewpoints and preferences of both past and prospective customers, it becomes evident that wireless networks within an industrial context are occasionally met with resistance, particularly when control is not in their hands. Furthermore, at present, Siemens does not provide a tablet with a

built-in emergency stop button, which is a common requirement for many customers and projects.

To tackle these issues, this thesis takes a pragmatic approach, drawing inspiration from the well-established INTRALOG standard. Instead of pursuing a fully wireless solution, the focus remains on maintaining the reliability of wired installations while leveraging the benefits of web and mobile technologies. This hybrid approach aims to strike a balance between innovation and practicality.

Recognising the growing importance of web-based technologies and mobile accessibility, this thesis suggests adopting a recently launched Mobile WebClient that harnesses the capabilities of WinCC Unified. The introduction of this panel addresses the previous lack of mobile technologies within the scope of this new visualization system. In contrast to stationary panels like the UCP, this approach provides enhanced flexibility in accessing the visualization interface while maintaining the advantages of wired installations.

The conceptual architecture for the visualization system can be succinctly described in four key components:

- **Control Technology** - This segment is primarily defined by the PLC, responsible for managing the conveyor system. It enables the visualization interface to provide valuable data to the user as well as executing their control instructions. As the INTRALOG CS standard already incorporates a designated PLC, the current project will leverage the established SIMATIC S7-1500, 1517-3 Central Processing Unit (CPU), visually represented in blue within the system architecture (Figure 5.1).

- **Unified Server** - The MTP700 UCP, represented in red, serves as the visualization system backbone, storing the interface and managing clients access to the control technology. The choice to opt for the 7-inch variant of the Comfort Panel was primarily driven by the fact that all variants possess identical hardware and software capabilities, varying solely in screen size. Opting for the smaller variant allowed for a reduction in the overall solution cost. Additionally, since Mobile WebClients are more commonly used, having a smaller display on the server would not pose any constraints.

- **Unified Clients** - The system has the capability to accommodate multiple IWP10F Mobile WebClients (illustrated in purple), and operators can physically connect them to Connection Boxes distributed throughout the plant (represented in grey). This arrangement allows for establishing connections to the Unified Server, granting access to the visualization system. The actual number of Mobile WebClients utilised may vary depending on the project, but when utilising an UCP as the server, the system can support up to three clients connected simultaneously.

- **PROFINET network** - To establish interconnections between all system components, a PROFINET network is utilised, adhering to the established principles of the INTRALOG CS standard.

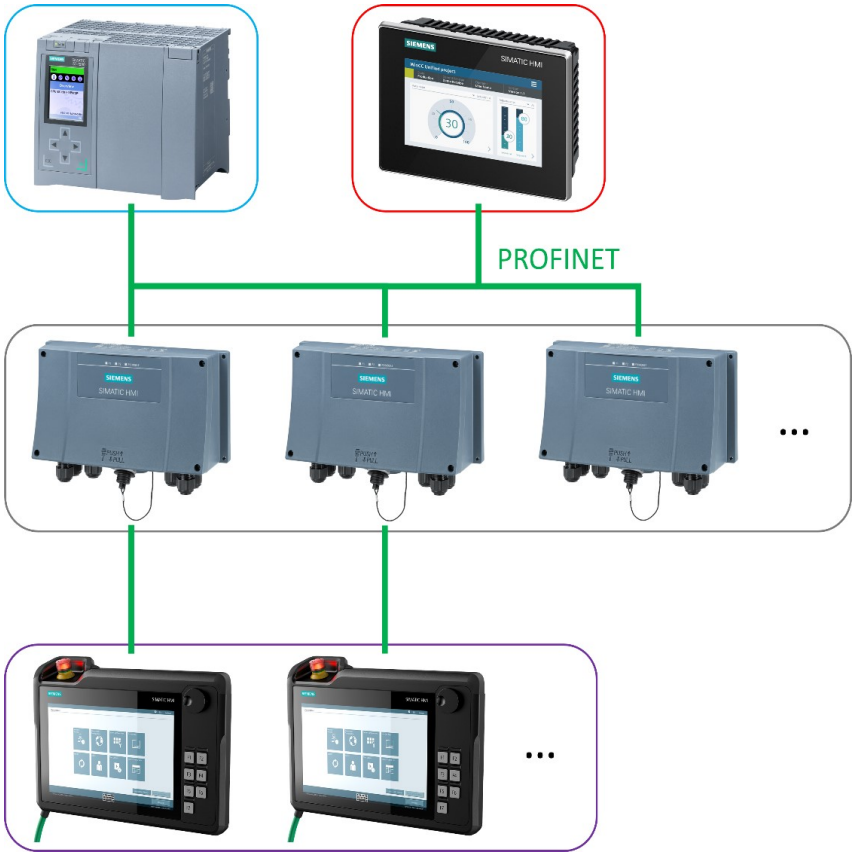


Figure 5.1: System architecture.

5.2 Visualization Interface

This section aims to explore the conceptualisation of the new visualization interface for the INTRALOG CS standard. Regarding the design, the concept is composed primarily by the idealisation of the interface layout, navigation architecture, and colour palette definition. Furthermore, the evaluation of screen components to be implemented, as well as the planning to transition all interface graphics to an SVG format, will be conducted. Lastly, the conceptualisation of the multiple clients approach is pursued, being given an explanation of why and how the usage of faceplates appear to be the most appropriate way to solve this topic.

5.2.1 Screen Layout

When a new project is initiated in WinCC Unified, after selecting the target device that will run the visualization system, a start screen must be configured. This screen

will be the first one loaded when the runtime is initiated, and its layers (support for 32) can be used to achieve several screen structures by means of screen windows. These windows can be loaded with a specific screen and stay unchanged throughout the entire runtime execution, or they can be modified via scripting based on the content that the interface must present. Screen windows can also be dynamically shown or hidden and repositioned or resized in order to allow the generation of multiple layouts during runtime. The arrangement of the screen windows layout on the start screen (Figure 5.2) will essentially generate the screen with which the operators will first interact.

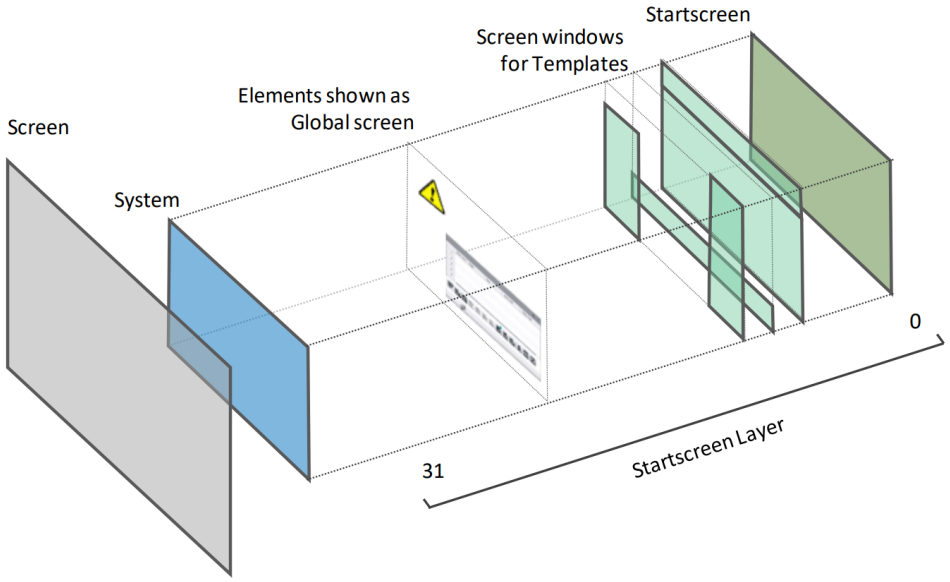


Figure 5.2: Unified screen structure [87].

The device that will run the visualization system for this project is made up of a 800x480 pixel panel. Because of its small size, it is critical to achieve a layout that only displays the most relevant information in order to avoid negatively affecting the visualization system's clarity and usability. With this in mind, Figure 5.3 illustrates the base layout for the start screen, which is composed of elements for operation, navigation, and status. There are four main areas in this layout:

- **Information bar** - Referring to the elements commonly found on mobile devices, the user interface should incorporate an upper bar that presents essential status and system information. This area will provide details about the connection status to the PLC, indicate the current state of the plant, display the currently active screen, and show the logged-in user. Additionally, this bar will enable the user to access the settings screen for further customisation and configuration.

- **Menu bar** - Since it is always visible, it should enable access to the key categories of user-system interaction while also showing an indication of the currently selected category.
- **Side navigation** - It comprises the necessary navigation elements for the sub-structures of the menu items, as seen on modern web and mobile systems. If all of the options are not viewable at the same time, by usage of the touch screen, the operator should be able to vertically scroll over the list. To increase clarity, the presently selected option will be highlighted. Although it is common for a side navigation to take up the entire height of the screen, in this application it should not overlap the information and menu bars in order to allow the operator to always be able to see their content, especially the first one, which provides vital system and plant status. Finally, since the side navigation overlaps the main content, it should not be visible at all times. To toggle it's visibility, a button, commonly referred as navigation handle, is used.
- **Main content** - The main area of interaction between the system and the operator will be loaded with the screens (usually referred as process screens) mostly requested by pushes on the menu bar or side navigation buttons.

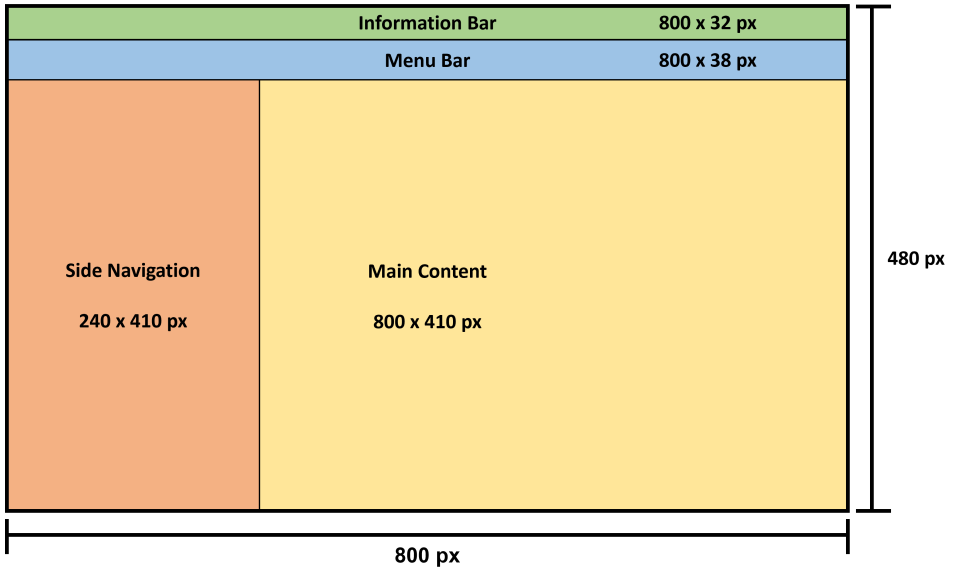


Figure 5.3: Start screen layout.

To enhance customer appeal and offer a versatile layout, the final solution aims to provide customisation options that can be utilised by both customers and operators. The settings screen allows users to modify the layout according to their preferences. Leveraging the built-in capabilities of WinCC Unified, the positioning and size of layout components can be easily adjusted. However, due to the textual content it

presents, the information bar is restricted to a horizontal form and can be positioned either at the top or bottom of the display. The menu bar can be oriented vertically or horizontally, offering the flexibility to position it on the left or right side, or alternatively, at the top or bottom of the interface. Lastly, the side navigation can be displayed on either the right or left side of the screen.

5.2.2 Navigation Architecture

The current visualization interface of the INTRALOG CS standard utilises a hierarchical navigation structure accompanied by a "return" button that loads the previously visited screen. Unfortunately, this leads to a cumbersome and unsatisfactory navigation experience, as multiple presses are often required to reach the desired content. In order to clarify the problem described, in Figure 5.4, it is possible to see the current navigation structure regarding the elements category.

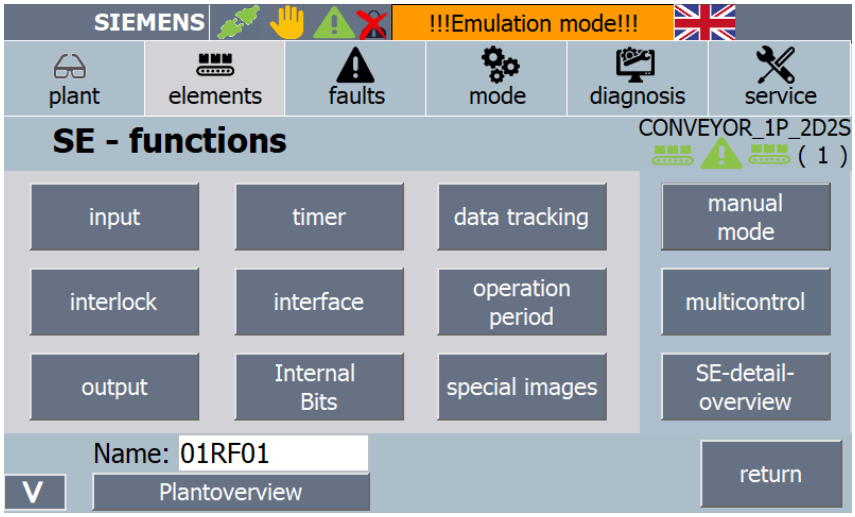


Figure 5.4: Elements overview screen on the old visualization system.

To enhance operational efficiency, a combined navigation structure should be implemented in the new visualization solution. In the proposed approach, a flat structure will be used for overview screens, while hierarchical structures will be employed for more complex relationships. One notable advantage of this navigation structure is the facilitation of faster and more efficient screen changes. Figure 5.5 illustrates a two-level combined navigation structure to be incorporated into the operating program.

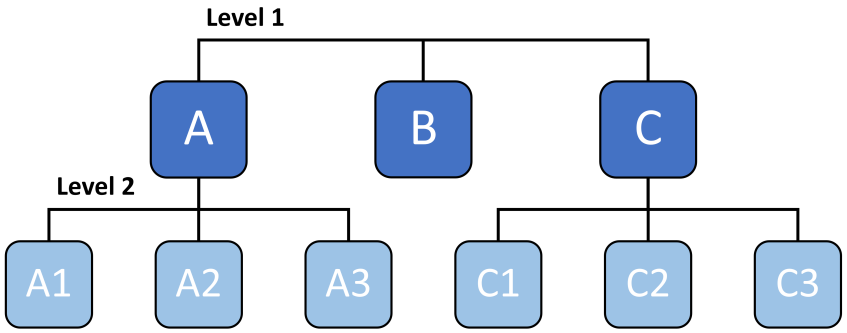


Figure 5.5: Proposed navigation architecture for the Unified interface.

The navigation elements in level 1 should always be readily available and usable, regardless of the user's current navigation level. As a fundamental principle, when the user is on level 2, only the corresponding navigation elements within the same substructure should be accessible. For instance, if the user is in A2, they can access other elements within the A substructure but not those in the C substructure (C1 - C3). However, in some cases, this rule may be disregarded if transitioning between different substructures is essential or if it enhances the user experience by enabling faster navigation.

The flat navigation structure for the first level will be implemented using a menu bar. Unlike the previous design, the menu bar will be constantly visible and not hidden when specific process images are displayed. This change ensures that direct navigation is always possible and prevents unexpected operating events not getting quickly evaluated, caused by the menu bar being concealed. The continuity of navigation and information elements is reminiscent of current user interfaces found in apps or websites, where the general navigation structure remains consistent regardless of the user's location within the application.

As represented in the "Elements Overview" screen (Figure 5.4) of the visualization in WinCC Advanced, the new menu bar will include the same categories: Plant, Elements, Faults, Mode, Diagnosis and Service.

Similar to before, sub-menus will be available for the Diagnosis, Elements, and Service categories. The display of sub-menus will be implemented using a side navigation, allowing for quick changes between process screens. The side navigation should have the option to be hidden, ensuring that the visualization of corresponding process screens is not obstructed or restricted.

The Service menu category will undergo changes. It will now solely encompass system functionalities, as several functionalities are no longer relevant in the new system (e.g., calibration screen). The panel functionalities and settings will be available through a separate screen, accessible via a scroll gesture on the information bar.

5.2.3 Colours

To achieve a unified and aesthetically pleasing design, it is advisable to employ a consistent colour scheme throughout the entire design to display all screen components. The colour palette should be derived from the established HMI Template Suite, which offers a variety of templates for creating modern user interfaces across Siemens process visualization systems [88]. By adopting a consistent colour palette, it becomes easier to maintain a cohesive look and feel, further enhanced by the repeated use of screen components. The existing projects within the HMI Template Suite serve as a foundation and provide guidance for creating contemporary visualizations in industrial environments. Overall, adhering to a unified colour scheme based on the HMI Template Suite not only ensures visual consistency but also strengthens the alignment of the INTRALOG CS standard with Siemens' established design language.

The Template Suite employs a flat design approach, characterised by a minimalist style that avoids three-dimensional effects. This approach creates a clear and focused visual representation, directing the user's attention to the pertinent information. The colour concept comprises different colours, serving specific purposes [88]:

- **Accent Colour** - Utilised to highlight important components or elements.
- **Navigation Colours** - Used to distinguish and design the different levels of navigation, setting them apart from the content area of the visualization.
- **Background Colour** - Employed for the content or operation-relevant section of the user interface, typically the main screen or window.

While the colours from the Template Suite will largely be utilised in this work, it is necessary to introduce additional colour tones to improve contrast. The colours provided below (Table 5.1) are those shared between the new design for the INTRALOG standard and the HMI Template Suite. By incorporating different shades of the base colours, the visual connection to existing Siemens user interfaces can be maintained, preserving a sense of familiarity.

To enhance the existing range of colours mentioned earlier, additional ones are needed to provide a more intuitive representation, particularly concerning the processes and system state, as well as instructions imposed by the user. These colours are further explored in Table 5.2.

Table 5.1: Colour design concept shared with the HMI Template Suite [88].

Colour	RGB colour code	Function
	0,161,209	Accent colour System or user controlled active state
	38,39,41	General navigation background
	72,73,78	Information bar background Main navigation bar background
	181,190,197	System or user controlled inactive state
	255,255,255	Content area background
	205, 211, 215	Main content background

Table 5.2: Colour design concept for process or system related states and instructions [88].

Colour	RGB colour code	Function
	170, 0, 0	Active fault background
	255,0,0	Active fault accent colour
	94,209,173	No active fault Automatic mode
	255, 153, 0	Manual mode
	50, 168, 125	Process related active state
	234, 206, 33	Warning colour 1
	222,56,88	Warning colour 2

5.2.4 Screen Components

It is recommended to utilise the existing screen components available in the engineering toolbox of the TIA Portal for designing the interface screens in order to achieve the best performance of the system. Custom web controls should be avoided because they are not easily accessible to the designer, which hampers the maintainability of the operating programs. Furthermore, adjusting custom web controls would require external code editing software and programming knowledge. Since conventional screen components and scripts can achieve the necessary functionalities, like a scrollable side screen navigation, the use of custom web controls is not obligatory and would only increase the software complexity.

To establish a structural basis for the screens, rectangular blocks are employed to arrange screen components, representing contextual boundaries or functional relationships of the content. These rectangles have rounded corners and a white background colour, adhering to the flat design approach. The colour scheme described earlier should also be implemented in the screen components, aiming for consistency and a minimalist style with fewer colours.

Boolean states, such as sensors or interface signals, are depicted using circles. The background colour of the circles should indicate the corresponding state, with green representing an active signal and grey denoting an inactive signal. User-controllable elements can be represented using an ON/OFF switch if appropriate for reflecting the current parameter state.

Output values, which cannot be altered by the user, should be displayed as either text boxes or output fields with a transparent background. They may have a light grey border or no border at all. Input fields should be marked with a dark grey border.

Control elements or buttons used in the process screen serve either as system operation or data adjustment controls, or navigation elements. To visually distinguish the functionalities of these control components, colour differentiation is recommended. Dark grey colour should be used for general control-related functions, while anthracite should be employed for navigation-related tasks, like the transition to a different screen.

Finally, pop-up windows can be employed to offer extended functionalities or display warnings.

5.2.5 Usage of SVG

The inclusion of SVG file support in the Unified visualization interface presents an excellent opportunity to revitalise the outdated graphics and integrate new ones. By leveraging SVG, the graphics can be appropriately scaled and displayed on screens of any size, ensuring no compromise in quality or distortion. Furthermore, transitioning

to SVG graphics offers the advantage of dynamic elements, enabling the provision of additional information to the user through colour changes.

For example, in a SVG graphic representation of a conveyor system, the conveyor housing can be dynamically changed to red to signify a failure or issue with that specific element. This interactive element improves user experience by providing more intuitive visual indications.

By adopting SVG files and replacing current visuals with this format, the visualization interface is given not only aesthetic but also functional benefits. SVG graphics' adaptability and flexibility open up possibilities for effectively and engagingly presenting information to users.

The Siemens HMI Template Suite makes use of several SVG that will be reutilised on the INTRALOG CS visualization project.

5.2.6 Multiple Instances Approach

As mentioned earlier, WinCC Unified utilises a client-server architecture. The clients act as mere connections without possessing a copy of the visualization system. Consequently, the server assumes a central role in managing all Unified instances. In order to have multiple clients connected to a server and make each one of them able to simultaneously access the control system, more specifically the PLC data, it is necessary to create a mechanism that supports the loading of a certain set of variables based on a specific client connection.

It is crucial to grasp the concept that each HMI connected to the system will operate with a corresponding FB instance running on the PLC. This FB instance acts as an intermediary between the visualization interface and the control system. Its role is to manage access to the control functionalities for each server/client, depending on the control point to which the panel is connected. This approach enables the distinction of multiple control points within the plant and grants control access solely to the elements that the user can physically see, determined by the location of the operator panel. Additionally, in the Unified Server project, a tag table will be created for each HMI FB instance. When an HMI panel is connected, the predefined tag table's tags will be loaded into the visualization, enabling the user to control the system.

However, a certain level of complexity arises because only the server is subject to engineering in the TIA Portal, as it is the only HMI that has a downloaded copy of the project. Consequently, a mechanism is required to allow the server to identify and differentiate both its own instance and all connected clients. Furthermore, when a screen is loaded, this information becomes crucial in order to retrieve the appropriate tag table. This ensures that all screen parameters are correctly associated with the corresponding tag.

Due to the large number of parameters associated with a single screen and the necessity of refreshing its content when data changes occur, relying on scripts to manage this would be highly complex and resource-intensive, resulting in poor performance. To address these challenges, in the current state of WinCC Unified, the most viable solution appears to be the utilisation of faceplates. According to Siemens documentation: "faceplates are user-defined groups of display and operating components" [89]. Faceplates are commonly implemented as reusable pieces of screens that can be instantiated multiple times with different sets of tags. This approach reduces engineering time and promotes visual coherence. In this project, while adhering to this ideology, the usage of faceplates will be more prominently emphasised, enabling the loading of specific tags for each screen based on the current server/client instance.

Essentially, almost every screen will consist of one or more faceplates. Upon loading the screen, a script mechanism will evaluate the appropriate tag table to use and subsequently load the corresponding tags onto the faceplates. This allows the server to accommodate its own instance as well as multiple clients. By developing all the screens content as faceplates, the definition of a specific tag table, that would be used for all Unified instances, can be disregarded. With this approach, faceplate parameters are filled with the UDT straight from the PLC rather than being FB instance-specific.

When utilising a faceplate, its interfaces can be either hard-coded or manipulated through scripting. In this case, compared to the scenario where all screen parameters would need to be manipulated without using the faceplate approach, only a few interface parameters need to be written. These parameters essentially determine the tag table to use for a particular PLC UDT, defined on the creation of the "Tag Interface", during the faceplate development process (Figure 5.6).

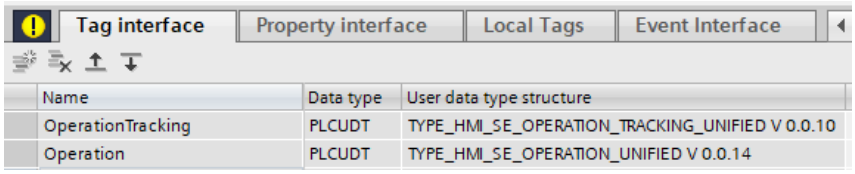


Figure 5.6: Definition of a faceplate interface tag using PLC UDT.

Chapter 6

Implementation

This chapter explores the implementation process of the previously conceptualised visualization solution. Firstly, it provides an explanation of how the interface between the control technology and the visualization system is executed. Next, it discusses the development of the interface itself using Unified. Lastly, the testing process is explained, along with the consequent validations conducted.

As a final note about the implementation of this project, it is important to mention that the version 18 of the TIA Portal and WinCC Unified are used, with the latest updates installed.

6.1 Interfacing Control and Visualization

This section aims to provide an overview of the necessary steps involved in interfacing the primary components of this project: the control technology, represented by the PLC, and the visualization interface, which will be deployed on the HMI panels. The following discussion will shed light on the general structure of the PLC FB and Data Block (DB) essential for implementing the new visualization solution.

The subsequent part will focus on the procedure specific to the Unified HMI project. It will examine the tags required for data exchange with the PLC and address the scripting necessary to simulate the functionality of WinCC Advanced area pointers, which are no longer available in Unified.

6.1.1 Control Data Provision to the Visualization Interface

The primary FB for interfacing the control and visualization components of the project is referred to as "FB_HMI_PANEL". It handles the preparation of data for exchange (reading/writing) with the visualization interface. The majority of external data for visualization programs and parameters originate from the "FB_HMI_PANEL" Instance Data Block (IDB). This principle applies to both panels and SCADA systems, although there are slight differences in the information exchanged between the "FB_HMI_PANEL" and the operator panel.

To facilitate the transition from Advanced to Unified, while preserving the functionalities of the original visualization system, a naming convention has been established. Whenever a block requires adjustments or the implementation of new functionalities, the suffix "_UNIFIED" is added. For instance, FB like "FB_HMI_PANEL_UNIFIED" and "FB_HMI_PANEL_TRACKING_UNIFIED" have been created. Initially, these FB were identical copies of the original ones. However, as the new interface was developed, they started to diverge from their original counterparts. Implementing new features not only involves modifying FB, but also PLC UDT and constants. The naming logic applied to these elements follows the same convention. This approach enables simultaneous usage of both runtimes (Advanced and Unified) for comparisons and validations. Once Unified becomes the predominant visualization system for the standard, the original naming will be restored, as WinCC Advanced will no longer be required.

The "FB_HMI_PANEL_UNIFIED" is invoked once in the "FC_HMI" (Figure 6.1) function for each visualization instance that the system is programmed to contemplate. This means that in addition to the Unified Server instance, for each Unified Client, a function block must be instantiated. This FB holds several key functionalities for implementing the visualization interface, such as:

- Managing the panel's connection to the system.
- Preparing data from various memory areas and making them accessible to the visualization based on the active screen.
- Writing back data from the visualization, such as user inputs or control commands, to the corresponding memory areas.

In addition to the "FB_HMI_PANEL_UNIFIED", there is a dedicated FB called "FB_HMI_PANEL_TRACKING_UNIFIED" that serves as an extension of the first. This FB is designed to interact with the HMI and specifically handles the processing and interfacing of data tracking associated with each single element that supports it. It is crucial to note that data tracking is not supported by all elements. For instance, elements such as barcode scanners do not offer data tracking capabilities.

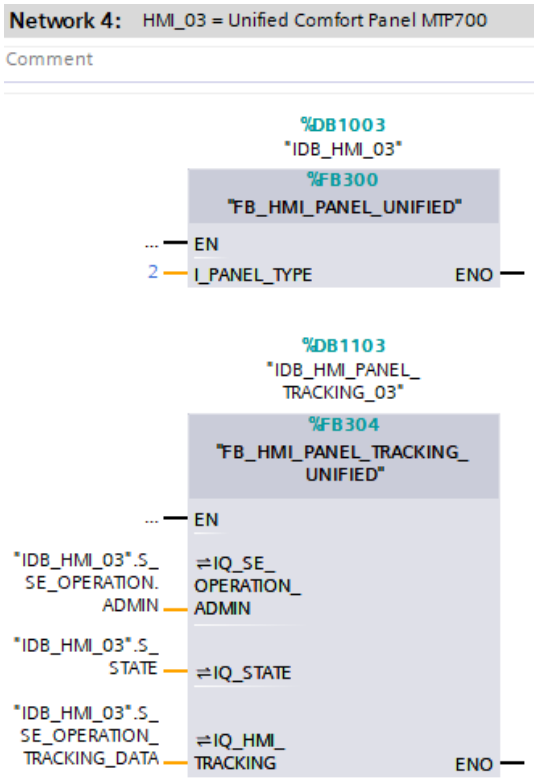


Figure 6.1: Network in the "FC_HMI" for interface with MTP700 UCP.

Although the "FB_HMI_PANEL_UNIFIED" accesses multiple DB, only the most relevant ones will be briefly introduced.

The "DB_PARAMETER" is essential for all visualization functions regarding single elements, as this DB lists all of them (maximum value defined by the PLC constant "C_SE_NO") and stores relevant data for system operation. The DB is formed by an array of structured data types, with each single element having its own entry (Figure 6.2).

The current online data from the PLC is highlighted in orange, and in this specific project, the first single element is the conveyor "01RF01". Within the data structure, under the "Operation" section, there is a list of structs that will be accessed from the "FB_HMI_PANEL_UNIFIED". Based on the predefined screens, it is possible to display specific data or control the single elements.

DB_PARAMETER

	Name	Data type	Monitor value
1	Static		
2	NEW	Bool	FALSE
3	DEL	"TYPE_PARAMETER"	
4	SE	Array[1.."C_SE_NO"] of "TYPE_PARAMETER"	
5	SE[1]	"TYPE_PARAMETER"	
6	INFO	Struct	
7	NAME	String["C_NAME"]	'01RF01'
8	FUNCTION	String["C_FUNC"]	'CONVEYOR_1P_2D2S'
9	GROUP_CL...	Array[1.."C_GROUP_NO"] of Bool	
10	ADMIN	Struct	
11	OPERATION	Struct	
12	STAT	"TYPE_STAT"	
13	STAT_CONV	"TYPE_STAT_CONV"	
14	MANUAL	Struct	

Figure 6.2: "DB_PARAMETER" data structure.

Regarding the "DB_OPERATION" (Figure 6.3), it focuses on storing information related to overall plant and CP status, as well as operating modes of element groups. This DB provides insights into various aspects, such as whether the plant is in an ON/OFF state, whether a CP is experiencing a fault, or if a group is operating in automatic or manual mode. In order to make the DB adjustable to any project, the size of the arrays, regarding CP and groups, are parameterised by changing the value of the PLC constants "C_CP_NO" and "C_GROUP_NO".

DB_OPERATION

	Name	Data type
1	Static	
2	PLANT	"TYPE_OPERATION_PLANT"
3	STAT	"TYPE_STAT_PLANT"
4	STAT_COLLECT	"TYPE_STAT"
5	FBACK	"TYPE_STAT"
6	CONTROL	Struct
7	FBACK_USER_MESSAGE	Struct
8	CP	Array[1.."C_CP_NO"] of "TYPE_OPERATION_CP"
9	GROUP	Array[1.."C_GROUP_NO"] of "TYPE_OPERATION_GROUP"

Figure 6.3: "DB_OPERATION" data structure.

On the other hand, the "DB_PARAMETER_GLOBAL" (Figure 6.4) contains general data about the system. It allows for the extraction of information such as determining if a specific visualization option is active for the entire project or for a CP. Furthermore, for the multiple CP distributed throughout the plant, is is possible to evaluate what groups they can individually control, based on their position on the plant. Additionally, the DB allows retrieving labels for element groups.

DB_PARAMETER_GLOBAL

	Name	Data type
1	Static	
2	PARAMETER	"TYPE_PARAMETER_GLOBAL"
3	MESSAGES	Struct
4	FUNCTIONS	Struct
5	LIGHTS_PARA	Struct
6	VISU_FUNCTIONS	"TYPE_VISU_FUNCTIONS"
7	VISU_FUNCTIONS_GLOBAL	Bool
8	AUTO_START_ACTIV	Bool
9	LABEL	Struct
10	TIMER	Struct
11	CP_CLASS	Array[1.."C_CP_NO"] of Struct
12	GROUP	Array[1.."C_GROUP_NO"] of Struct
13	GROUP_COUNT	Int
14	ONLY_ONE_GROUP	Bool

Figure 6.4: "DB_PARAMETER_GLOBAL'" data structure.

The "DB_TRACKING" (Figure 6.5) provides valuable insights into the data tracking of components being conveyed by the system. This DB consists of an array that encompasses all the available places supported by each conveyor. For each place, it contains information about the corresponding element and detailed tracking data, including dimensions and the origin and destination of a pallet or container.

DB_TRACKING

	Name	Data type	Monitor value
1	Static		
2	ADMIN	Struct	
3	DEL	"TYPE_TRACKING"	
4	PLACE	Array[1.."C_TR_PLACES"] of "TYPE_TRACKING"	
5	PLACE[1]	"TYPE_TRACKING"	
6	NAME	String["C_NAME"]	'01RF01'
7	SE_NO	UInt	1
8	PLACE	USInt	1
9	IN	Bool	TRUE
10	SHARE_PLACE	Bool	FALSE
11	WMS_INDEX	UInt	0
12	STAT	"TYPE_TRACKING_STAT"	
13	COORD	"TYPE_TRACKING_COORD"	
14	DATA	"TYPE_TRACKING_DATA"	
15	MF	"TYPE_TRACKING_MF"	

Figure 6.5: "DB_TRACKING" data structure.

Figure 6.6 presents a block diagram with the purpose of visually illustrating and condensing the information described earlier. This diagram showcases the interactions between the PLC blocks, exhibiting how they enable the control of the system through the visualization interface.

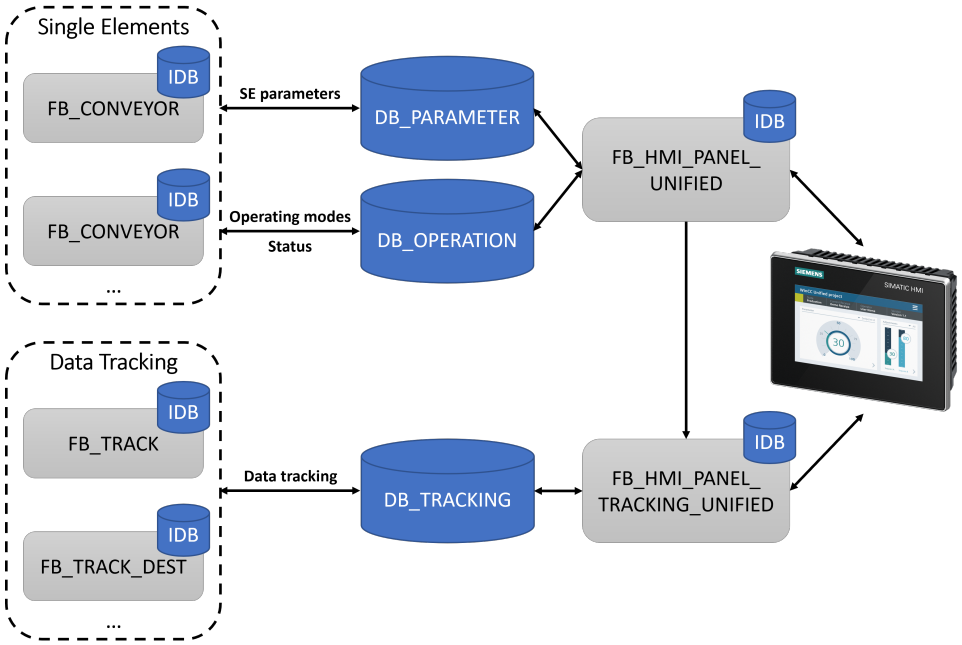


Figure 6.6: PLC blocks structure for interfacing control and visualization.

As an extension package available for the customers, the INTRALOG CS standard offers the possibility of providing screens with the overall graphic representation of the plant. In order to providing the required data to construct these screens, the "FC_HMI_GRAFIC_OVERVIEW_MULITPLEX" function (Figure 6.7) is used on a separate network in the "FC_HMI" function. Following the same working principle of the "FB_HMI_PANEL_UNIFIED", this function is called for each visualization instance and, depending on the currently active screen on the panel, tags multiplexing is performed in order to construct several plant layouts while maintaining the number of necessary tags to a minimum.

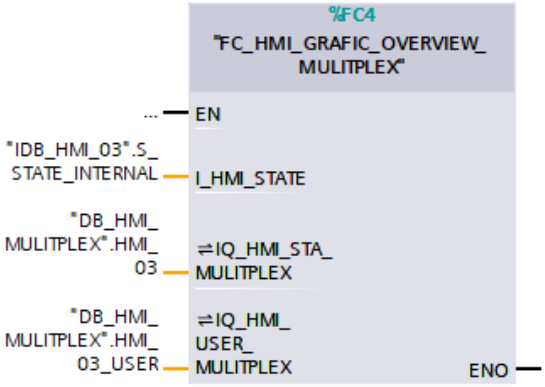


Figure 6.7: Network in the "FC_HMI" for providing the data to construct the graphic plant overview.

6.1.2 Integration of the IWP10F Mobile WebClients

To facilitate the management of connections for the IWP10F Mobile WebClients within the system, a global DB named “DB_IWP10F” has been created. This DB consists of an array with three positions, each of which holds data of the type “Struct”. The system’s limitation of supporting a maximum of three active clients simultaneously has led to the pre-definition of this array size, which is defined under the “C_IWP10F_NO” constant. For every instance of IWP10F, four main data entries need to be considered:

1. **BOX_ID** - This entry stores the Connection Box ID associated with the panel’s connection.
2. **ASSIGNED** - Indicates whether the IWP10F panel is currently assigned to an instance of “FB_HMI_PANEL_UNIFIED”.
3. **ACTIVE** - Denotes whether the IWP10F is currently connected to the system.
4. **STATION_INFO** - This is a structure of tags responsible for managing and storing the IP address readings of the IWP10F panel.

The data required to fill this DB is obtained by executing a FB within the first network of “FC_HMI”. This FB, called “FB_IWP10F” (Figure 6.8), is executed for each IWP10F integrated into the system and was specifically developed for this project, taking two input parameters: the unique device identifier and the ID read by the panel from the Connection Box.

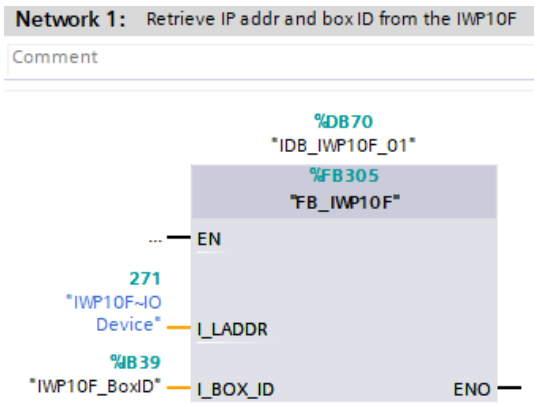


Figure 6.8: “FB_IWP10F” call on the “FC_HMI”.

To provide the first parameter, the system constant “IWP10F~IODevice” is used, which holds the value 271 as the unique identifier for the IWP10F panel. These system constants are automatically generated when an IWP10F is added from the hardware catalogue. As for the second input parameter, it receives the value from

the “IWP10F_BoxID” tag, which is of the “BYTE” data type. This tag stores the unique hexadecimal value of the Connection Box ID obtained from memory address 39, which can be retrieved by accessing the “Device view” of the desired IWP10F panel.

Upon execution, the “FB_IWP10F” uses built-in functions from the INTRALOG standard to extract the instance name of the function, which, in this case, is “IDB_IWP10F_01”. From the instance name, the instance number is derived by considering the value set in the last two characters of the name (Listing 6.1). In this case, it refers to instance number one, as it is the first IWP10F panel being evaluated. The instance number is then used to write data to the appropriate array position in “DB_IWP10F”.

```
1 IF #S_INSTANCE_NAME = '' THEN
2     #S_INSTANCE_NAME := "FC_INSTANCENAME"(GetInstanceName(0));
3
4     STRG_VAL(IN := (WSTRING_TO_STRING(MID(IN := STRING_TO_WSTRING
          (#S_INSTANCE_NAME), L := 2, P := 8))),
5               FORMAT := W#16#0000,
6               P := 1,
7               OUT => #S_INSTANCE_NUMBER);
8 END_IF;
```

Listing 6.1: Instructions on the “FB_IWP10F” to extract name and number of the instance.

Furthermore, the “FB_IWP10F” checks whether the Connection Box ID is zero, indicating that the panel is not connected. In this case, the active state is set to “FALSE”, the box ID receives 0, and the IP address of the panel is not read (execute bit is set to “FALSE”). However, if the Connection Box ID is not zero, it implies that the panel is physically connected to the system and therefore active. In such a scenario, the IP address must be retrieved, so the “GetStationInfo” block is triggered and the Box ID stored (Listing 6.2). A data type conversion is performed before storing the Connection Box ID, converting it from “BYTE” to “UINT”. This conversion is necessary because “FB_HMI_PANEL_UNIFIED” requires the variable to be of type “UINT”. Similarly, “FB_CP” requires the input “I_CBOX_ID” to be configured with the same data type.

```
1 IF (#I_BOX_ID = 0) THEN
2     "DB_IWP10F".IWP10F[#S_INSTANCE_NUMBER].ACTIVE := FALSE;
3     "DB_IWP10F".IWP10F[#S_INSTANCE_NUMBER].BOX_ID := #I_BOX_ID;
4     "DB_IWP10F".IWP10F[#S_INSTANCE_NUMBER].STATION_INFO.execute :=
          FALSE;
5 ELSE
```

```
6       "DB_IWP10F".IWP10F[#S_INSTANCE_NUMBER].ACTIVE := TRUE;
7       "DB_IWP10F".IWP10F[#S_INSTANCE_NUMBER].BOX_ID := BYTE_TO_UINT(
            IN := #I_BOX_ID);
8       "DB_IWP10F".IWP10F[#S_INSTANCE_NUMBER].STATION_INFO.execute :=
             TRUE;
9   END_IF;
```

Listing 6.2: Instructions on the "FB_IWP10F" performed based on the Connection Box ID value.

Finally, in the "FB_IWP10F", the "GetStationInfo" function, which is part of the internal library, is utilised as a multi-instance function (see Listing 6.3). This implementation ensures that the memory used for its instantiation remains within the scope of the calling FB ("FB_IWP10F"). Consequently, there is no need to create a separate IDB, making system parameterization more straightforward. Its primary objective is to read the IP address of the IWP10F panel. By setting the input parameter "REQ" to true, the block writes the IPv4 parameters (such as subnet mask and IP address) to the I/O parameter "DATA". To determine the system component from which the information should be retrieved, the input "LADDR" must receive the unique device identifier, which has been given as an input to the "FB_IWP10F".

```
1   #GetStationInfo(REQ:="DB_IWP10F".IWP10F[#S_INSTANCE_NUMBER].
        STATION_INFO.execute,
2   LADDR:=#I_LADDR,
3   MODE:=1,
4   DONE=>"DB_IWP10F".IWP10F[#S_INSTANCE_NUMBER].STATION_INFO.done,
5   BUSY=>"DB_IWP10F".IWP10F[#S_INSTANCE_NUMBER].STATION_INFO.busy,
6   ERROR=>"DB_IWP10F".IWP10F[#S_INSTANCE_NUMBER].STATION_INFO.error,
7   STATUS=>"DB_IWP10F".IWP10F[#S_INSTANCE_NUMBER].STATION_INFO.status
        ,
8   DATA:="DB_IWP10F".IWP10F[#S_INSTANCE_NUMBER].STATION_INFO.
        IP_address);
```

Listing 6.3: Instructions on the "FB_IWP10F" performed to read the panel IP address.

After the IWP10F panel has been registered in the designated DB, the next step is to establish a connection to an HMI block instance to grant the panel access to the control system. As mentioned earlier, the "FB_HMI_PANEL_UNIFIED" is designed to achieve this and should be instantiated multiple times to support the number of visualization interfaces the system requires. To create a flexible system that doesn't depend on hardcoded IP addresses for client HMI instances, and considering that the number of Unified Clients connected to the system may change frequently, a mechanism is needed to dynamically allocate Mobile WebClients

to an HMI instance. The “FB_HMI_PANEL_UNIFIED” block is responsible for implementing this procedure, accordingly to the Listing 6.4.

```
1  IF #S_STATE.TYPE.WINCC OR #S_STATE.TYPE.PANEL THEN
2      #S_CP_ID := #S_STATE.HMI_NO;
3  ELSIF #S_STATE.TYPE.MOP AND #S_MEM.IWP10F_NO = 0 THEN
4      FOR #t_loop_IWP10F := 1 TO "C_IWP10F_NO" DO
5          IF NOT "DB_IWP10F".IWP10F[#t_loop_IWP10F].ASSIGNED AND "
               DB_IWP10F".IWP10F[#t_loop_IWP10F].ACTIVE THEN
6              "DB_IWP10F".IWP10F[#t_loop_IWP10F].ASSIGNED := TRUE;
7              #S_MEM.IWP10F_NO := #t_loop_IWP10F;
8              #S_CP_ID := "DB_IWP10F".IWP10F[#t_loop_IWP10F].BOX_ID;
9              #S_HMI_IP[1] := "DB_IWP10F".IWP10F[#t_loop_IWP10F].
                   STATION_INFO.IP_address.InterfaceAddress.ADDR[1];
10 // ...
11             #S_HMI_IP[4] := "DB_IWP10F".IWP10F[#t_loop_IWP10F].
                   STATION_INFO.IP_address.InterfaceAddress.ADDR[4];
12         END_IF;
13     END_FOR;
14 END_IF;
```

Listing 6.4: Instructions to assign an IWP10F panel to an HMI instance.

When the “FB_HMI_PANEL_UNIFIED” is instantiated with the “_MOP” on the IDB name, it means that the instance is intended to serve as an interface with a Mobile WebClient, leading the “S_STATE.TYPE.MOP” variable to be set to TRUE. This condition allows entry into a FOR loop, which then goes through the indexes of the IWP10F instances that the system is intended to accommodate. The purpose of this process is to identify instances of “DB_IWP10F” that are not yet assigned to any HMI block but are actively connected to the system. Once such an instance is found, a connection is established between the IWP10F panel and the corresponding HMI block instance. The connection index, along with the Connection Box ID and the panel’s IP address, is then stored in the appropriate IDB tags. The “S_CP_ID” array is utilised to link the HMI instance to a connection point, and the “S_HMI_IP” array is used to compare the IP address read on the Unified side and match it with one of those stored on the HMI instances. This matching process allows the appropriate set of tags to be loaded for each Unified client. On the other hand, when the HMI instance is created for a fixed panel, like the MTP700 Unified Server, the “S_STATE.TYPE.PANEL” variable becomes active, and the first condition is entered. Fixed panels are continuously connected to the system and act as servers, since they have a downloaded copy of the visualization interface. In this case, the Connection Box ID takes the value of the HMI number defined on the IDB name. Additionally, there is no need to write the IP address of

these fixed panels since the Unified side does not need to match it with a block, as the IDB for it is already known.

When the IWP10F panel is unplugged from the Connection Box, the corresponding HMI instance becomes inactive. It is important to free the established connection between the IWP10F panel and the HMI block in this situation. This disconnection is achieved through a condition that checks for the transition of the HMI block to an inactive state while an IWP10F index is still in memory. If this condition is met, the assigned state from "DB_IWP10F" is removed, as can be seen from Listing 6.5.

```
1 IF #S_MEM.IWP10F_NO <> 0 THEN
2     "DB_IWP10F".IWP10F[#S_MEM.IWP10F_NO].ASSIGNED := FALSE;
3     #S_MEM.IWP10F_NO := 0;
4 END_IF;
```

Listing 6.5: Instructions to unassign an IWP10F panel from the HMI instance when it becomes incative.

In Figure 6.9, a block diagram is presented to illustrate the process previously described and aid in its comprehension.

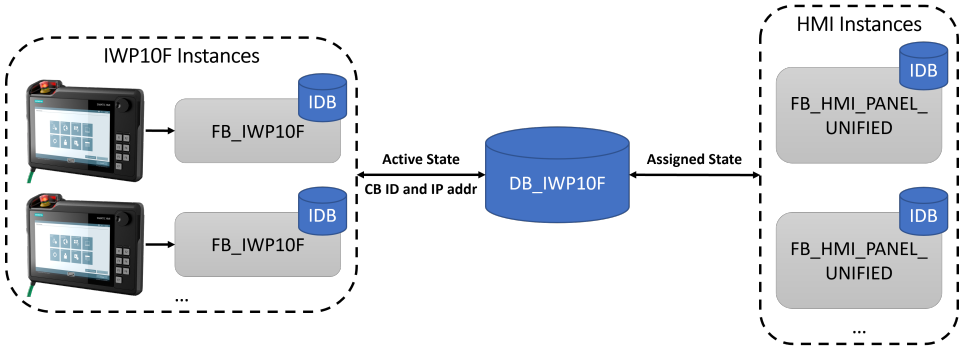


Figure 6.9: PLC blocks structure for the integration of the IWP10F Mobile WebClients into the system.

6.1.3 Establishing a Connection Between an HMI and a Control Point

In order to achieve the purposed objective of allowing the operator to control only the elements that he can physically see, the "FB_HMI_PANEL_UNIFIED" reads the Connection Box ID retrieved by the HMI panel and compares it with the corresponding tags existing on an array of CP stored in the "DB_OPERATION", accordingly to Figure 6.10. This way it is possible to identify the CP associated with the HMI instance and then, from the "DB_PARAMETER_GLOBAL", verify if the HMI should have control over a certain group of elements. Regarding this subject, the DB is filled with the configuration made on the "FC_PARAMETER_GLOBAL".

During the engineering process of the control system, multiple CP are strategically distributed on the plant and the groups that they can control are parameterised on that function. The array of CP available on the "DB_OPERATION" is written by every instance of the "FB_CP" called on the "FC_CP". For every CP existing on the plant, an instance of the FB must be called.

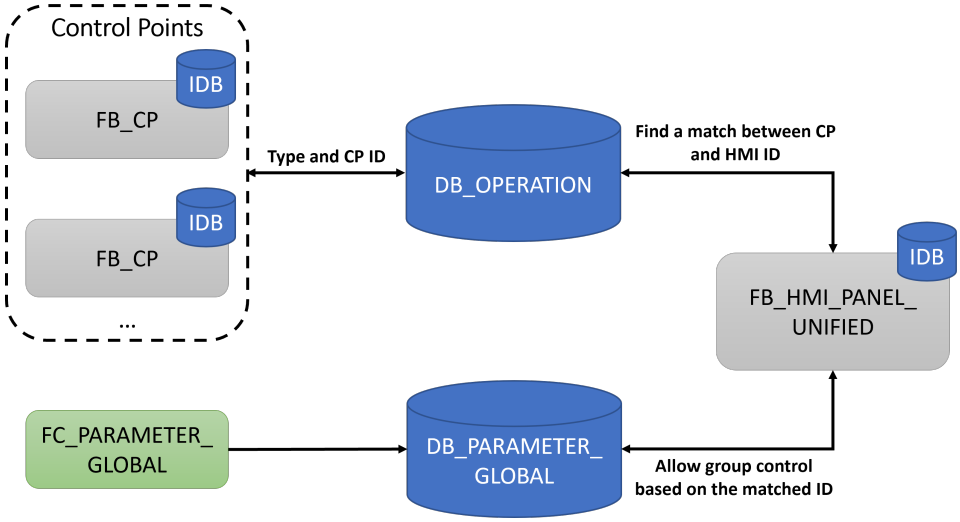


Figure 6.10: PLC blocks structure for interfacing the CP and the HMI instances.

As it is possible to see in Figure 6.11, the "FB_CP" receives two input parameters. The first, "I_TYPE", is the type of CP and can assume three different forms, expecting to receive a value between 1 and 3:

1. Connection Box.
2. Stationary Panel.
3. SCADA PC.

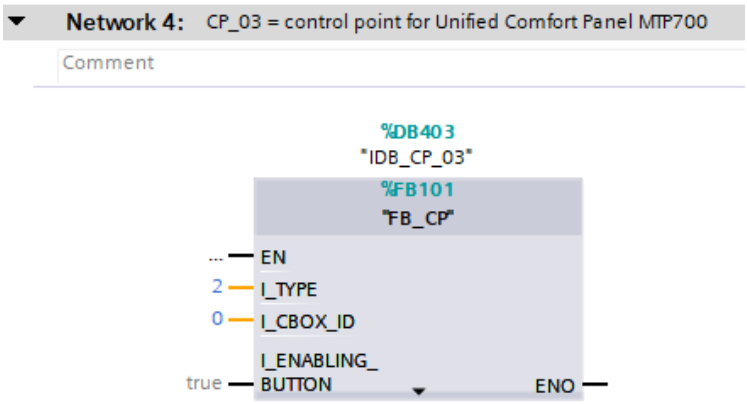


Figure 6.11: Network in the "FC_CP" for the MTP700 stationary CP.

The second parameter represents the ID, which is manually set using rotary switches on the Connection Box. If the device is not a Connection Box, this parameter should be set to 0, as the panel remains continuously connected to the system. Figure 6.11 illustrates this configuration for the CP associated with the MTP700 UCP serving as a Unified Server.

However, when performing parameterization for a Mobile WebClient like the IWP10F, acting as a Unified Client, the input parameter "I_CBOX_ID" should receive the decimal conversion of the hexadecimal value set via the rotary switches on the Connection Box. For example, if the ID on the Connection Box is set to 0xF0, then the "FB_CP" input parameter, which links the "FB_HMI_PANEL_-UNIFIED" instance to this CP, would require the value 240.

6.1.4 HMI Tag Tables

Once the PLC code development is complete, the next step involves the creation of the HMI variables, commonly known as tags. These tags are responsible for receiving data from the PLC memory areas, specifically from the IDB and DB previously discussed. They are stored within tag tables, which are established within the HMI project. These serve as structured representations of variables, facilitating the exchange of information between the HMI and other devices, such as a PLC. Apart from the tag tables linked to external content, it is also possible to create internal HMI tags. These internal tags are typically utilised within scripts and serve to store pertinent information regarding the runtime operation.

To enhance maintainability and streamline the development process, various folders were created to organise the tag tables based on their content, as represented in Figure 6.12.

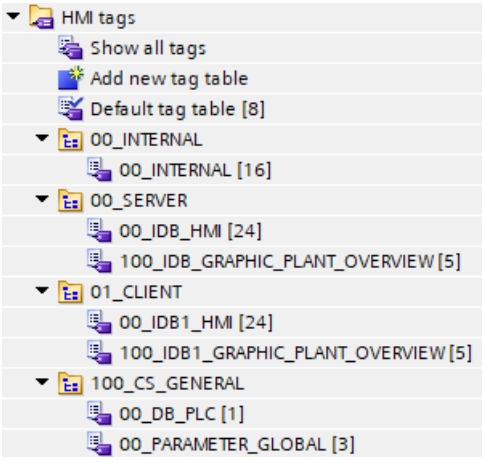


Figure 6.12: HMI project tag table structure.

Within the project, the "00_INTERNAL" folder contains the sole internal tag table. When defining each variable, the customary procedure of assigning a name

and data type was followed. Additionally, the properties of each tag were configured with a scope of "Session local". These tags are restricted to the current user session and are utilised for temporary or session-specific data. Conversely, "System-wide" tags can be accessed across multiple sessions and are suitable for storing shared or global information. By employing the "Session local" scope, the content of the Unified server and its running clients can be distinguished. This approach allows each instance to possess its own memory without necessitating the creation of multiple internal tag tables. This functionality proves beneficial, for instance, in storing the previously active screen for each main navigation option or configuring distinct layouts for individual Unified instances without impacting others.

For the server and clients' tags, individual folders are created, each containing two tag tables. The first tag table (Figure 6.13) encompasses variables derived from the IDB associated with the FB for interfacing data between the control and visualization systems, previously described. The second tag table houses tags necessary for constructing the graphic plant overview. The tag names chosen adhere to the naming conventions used in the IDB variables, with the added feature that the prefix is incremented for each predefined visualization instance. For example, the tags associated with the server, stored under "00_IDB_HMI", have the prefix "IDB". On the other hand, the tags for the first client, stored under "00_IDB1_HMI", will have the prefix "IDB1", and this pattern continues for all programmed clients. This naming convention is implemented to facilitate differentiation of tags during scripting, utilising JavaScript string manipulation instructions. The majority of these tags possess a PLC UDT, which is automatically assigned when selecting the PLC tag. This ensures consistency with the corresponding IDB data types. Once the HMI tag names and connections are established, the tag acquisition cycle can be configured. The tag acquisition cycle refers to the frequency at which the HMI system retrieves and updates data values from the PLC. The most critical data is set to be refreshed every 100 ms, while less crucial or infrequently changing data is set to either 500 ms or 1 s.

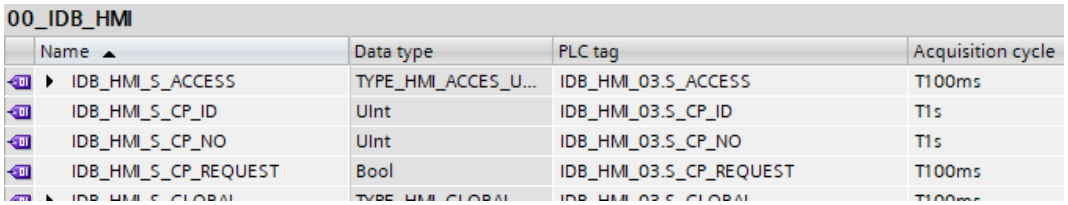


Figure 6.13: Server main tag table.

To supply the visualization interface with general information, such as the product manufacturer and retrieve certain configuration parameters, the tag table "00_-PARAMETER_GLOBAL" accesses variables from the previously mentioned DB of the same name. Furthermore, from "00_DB_PLC", it is possible to retrieve data

pertaining to the PLC performance and operational state. Since these tags are not considered critical to system operation, the acquisition cycle was set to 1 s.

6.1.5 Area Pointer Via Scripting

With the introduction of WinCC Unified, Siemens has removed the Area Pointer functionality that was previously available in the connection settings of an HMI panel. Area pointers are used to access specific data areas within the PLC. They facilitate read and write operations, enabling the PLC and panel to interact based on the evaluation of stored data. In this project, several area pointers are utilised [90]:

- **Coordination life bit** - The HMI toggles the life bit approximately once per second. By querying this bit in the control program, it is possible to determine whether a connection to the HMI device is still active.
- **Screen number** - The HMI writes the number of the currently displayed screen to the corresponding area pointer. This enables the PLC to initiate specific actions, such as executing certain program networks and consecutively display relevant data based on the active screen.
- **Job mailbox** - The job mailbox serves as a means for the PLC to transfer tasks to the HMI panel, initiating corresponding actions on the second. The first word of the job mailbox array contains the job number. The HMI evaluates the job mailbox if the job number is not equal to zero. Parameters must first be entered in the job mailbox, followed by the job number. As an example, if the PLC requests a screen change with the job number 51, the HMI device can read this value and evaluate the parameters to extract the desired screen from the PLC.

The HMI tags required for the area pointers can be found in the main client /server tag table, as illustrated in Figure 6.13. For the coordination life bit, the relevant tag is “IDB_HMI_S_RANGE_POINTER_COORDINATION”. When it comes to returning the screen number to the PLC, the appropriate interface tag is “IDB_HMI_S_RANGE_POINTER_IMAGE”. Lastly, for managing the job mailbox, the array of words labeled “IDB_HMI_S_RANGE_POINTER_CONTROL_-ORDER” should be utilised. The first position in this array contains the job number, while the second holds the job mailbox.

Due to the significance of these features in system operation, Siemens now recommends performing these tasks through scripting in the Unified environment [91].

For the cyclic toggle of the coordination life bit, a dedicated asynchronous single instruction script (Listing 6.6) is executed every 1 second. In this instruction, “LifeBit” refers to an interface tag within a faceplate that establishes the connection to the corresponding HMI tag, following the approach described in Subsection

5.2.6. Within the “FB_HMI_PANEL_UNIFIED”, the continuous toggling of the life bit ensures the active status of the HMI instance, allowing it to access the control program. If, at any point, the life bit ceases to toggle, the HMI instance becomes inactive, resulting in the clearing of accesses to the global DB and the control program. This also blocks any manual operations on the system that were previously in progress. Furthermore, the PLC will request the visualization instance to switch to the home screen, which will be discussed later in this document. This procedure is executed for safety reasons. In the event of an HMI-related fault or if the HMI device is unplugged, the control program prevents any unsupervised operations on the system.

```
1 HMIRuntime.Tags.SysFct.InvertBitInTag("LifeBit", 0)
```

Listing 6.6: Instruction to cyclically toggle the coordination life bit.

To notify the PLC about the active screen on the visualization interface, the “IDB_HMI_S_RANGE_POINTER_IMAGE” HMI tag needs to be written when a screen is loaded. This task is accomplished using the following instruction (Listing 6.7):

```
1 Tags(idbName+"_HMI_S_RANGE_POINTER_IMAGE").Write(Screen.
      ScreenNumber)
```

Listing 6.7: Instruction to inform the PLC about the number of the currently active screen on the visualization.

The provided instruction is executed during the loading event of the screen accompanied by another portion of the script that is omitted and will be discussed later. For now, only the process of writing the area pointer is relevant. The instruction reveals that the screen number is obtained using the command `Screen.ScreenNumber`. This command accesses the configuration properties of the screen and reads the value (0 by default) that the developer has pre-defined as the screen number (refer to Figure 6.14). To facilitate the development process, the number of screens should correspond to the prefix found in their names. In the PLC, a numeric constant is defined for each screen. Subsequently, in the “FC_HMI_PIC_UNIFIED”, the function “FC_HMI_PIC_UNIFIED” is invoked and provided with the current screen number. By comparing the area pointer tag with the defined constants, the program can determine the currently active screen and execute specific instructions based on this information.

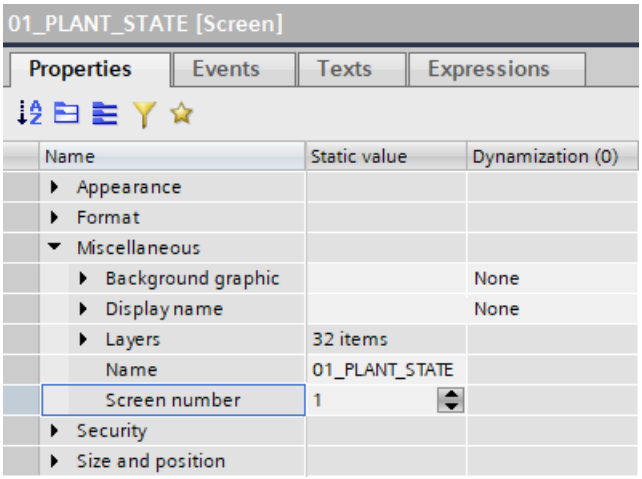


Figure 6.14: Screen number property definition.

In order to execute the control order operation, a script (Listing 6.8) is triggered when the job number tag, which occupies the first position in the array "IDB_-HMI_S_RANGE_POINTER_CONTROL_ORDER", undergoes a change. If the PLC writes a new job number, the script verifies its value and takes appropriate action. This specific project involves three different job numbers, specified as PLC constants, that need to be considered. When the PLC requests a new screen, the value 51 is written in the first byte of the job number tag. To assess this request, the tag is masked to extract the first byte value for validation purposes. Following this instruction, the job mailbox (located at the second position of the control order array) is read, and its content is copied to another tag, which will be utilised to initiate the screen change request. The other two important job numbers are 24 and 23. The value 24 indicates a request from the PLC to log off the current HMI user, while the value 23 triggers the appearance of the user login dialog. These features are valuable in scenarios like the one described earlier, where the HMI makes the transition to an inactive state, and they also help control access privileges over the system.

```
1  TB_Script_Text_OnPropertyChanged(item, value) {
2
3    let controlOrder0 = Tags("ControlOrder[0]");
4    let controlOrderRequest = controlOrder0.Read();
5
6    let controlOrder1 = Tags("ControlOrder[1]");
7    let controlOrderContent = controlOrder1.Read();
8
9    let firstByte = controlOrderRequest & 0xFF;
10
11   if (firstByte == 51 && controlOrderContent != 0) {
12     let requestedScreen = Tags("RequestedScreen");
```

```
13      requestedScreen.Write(controlOrderContent);
14      controlOrder1.Write(0);
15
16    } else if (controlOrderRequest == 24) {
17      let logOFF = Tags("LogIN_OFF");
18      logOFF.Write(1);
19
20    } else if (controlOrderRequest == 23) {
21      let logOFF = Tags("LogIN_OFF");
22      logOFF.Write(2);
23    }
24  }
```

Listing 6.8: Script to handle the control order area pointer.

6.1.6 Managing Multiple Unified Instances

As previously stated, the Unified server will store all the necessary tags, not only for its instance but also for the clients that will connect to it. For this reason, whenever the need to use a PLC tag arises, a mechanism is needed to differentiate the multiple Unified instances in order to connect the appropriate tags for the faceplates that will constitute the screens. To perform this operation, the "RetrieveIDB" function, stored under the global module, is used.

Initially, this function starts by having some constants defined (Listing 6.9). The "localMachineName" receives the IP of the Unified instance that called the function by reading the system tag "@LocalMachineName". As for the "serverName" constant, the string "Controlroom Client" is written into it since this is the name that is found on the system tag "@LocalMachineName" when the connection is the server. Additionally, the "simulationMachineName" receives the name of the PC that might want to run the interface in simulation mode, usually "localhost". This is relevant for training or development purposes. Lastly, regarding the definition of constants, it is possible to set under "maxClients" the number of allowed WebClients that can simultaneously access the visualization system, which for UCP should not be higher than three.

```
1  const localMachineName = Tags("@LocalMachineName").Read();
2  const serverName = "Controlroom Client";
3  const simulationMachineName = "localhost";
4  const maxClients = 1;
```

Listing 6.9: Constants definition on the "RetrieveIDB" function.

After this initial definition of variables, multiple conditions are verified in order to check what type of instance called the function and therefore return the appropriate

prefix to be employed during the tag table selection. Meaning that if the connection is to the server, the function return will be “IDB”, and if it is, for example, the first client, it should be “IDB1”, as previously explained.

The first condition checks if there are any clients allowed for the server and if the instance is effectively the server or eventually a PC executing the interface in simulation mode, leading to the appropriate tag table to use being the server, so the return should be “IDB” (Listing 6.10).

```
1 if ((maxClients == 0 && localMachineName == serverName) ||
      localMachineName == serverName || localMachineName ==
      simulationMachineName) {
2     return "IDB";
3 }
```

Listing 6.10: Conditions on the “RetrieveIDB” function to validate that the Unified instance is the server.

If the first condition was not validated, it means that the Unified instance that called the function should not receive the server tag table prefix, so it could be a client. In order to check this, a for loop is conducted in order to iterate through the number of allowed clients. For each client, the IP address is split into four parts and compared to the IP stored on the IDB of the PLC that are used to accommodate the clients. When the “checkClientIP” variable is true, it means that the IDB that should be used for this client instance has been found, and therefore the tag table to use should be the concatenation of the “IDB” string with the number of the client, stored on the loop index, as presented in Listing 6.11.

```
1   for (let i = 1; i <= maxClients; i++) {
2
3     let parts = localMachineName.split(".");
4
5     let checkClientIP =
6         parts[0] == Tags("IDB"+i+"_HMI_S_HMI_IP[0]").Read() &&
7         parts[1] == Tags("IDB"+i+"_HMI_S_HMI_IP[1]").Read() &&
8         parts[2] == Tags("IDB"+i+"_HMI_S_HMI_IP[2]").Read() &&
9         parts[3] == Tags("IDB"+i+"_HMI_S_HMI_IP[3]").Read();
10
11    if (checkClientIP) {
12      return "IDB"+i;
13    }
14  }
```

Listing 6.11: Conditions on the “RetrieveIDB” function to validate that the Unified instance is a client.

Lastly, if the connection reveals to be something that is not contemplated by the system, an empty string is returned, causing the system to not be able to connect to any valid tag table and the connection to the PLC to not be established. This is critical for security reasons, because if someone with network access attempts to access the server and its machine IP is not recognised by the system, it will be unable to do so.

The "RetrieveIDB" function is executed once, during the loading event of the Unified Runtime start screen, and the return value is stored in an internal HMI tag using: `Tags("internal_IDBName").Write(RetrieveIDB())`. This tag is configured to have a "Session local" scope; as a result, each instance will possess its own unique value.

Subsequently, whenever it becomes necessary to evaluate the Unified instance for handling PLC tags, the HMI tag can be read, returning the tag table prefix required to perform the desired instructions.

Throughout this document, this procedure is evident on numerous occasions. To elucidate its working principle, Listing 6.12 illustrates the script executed during the loading event of the elements "Detail Overview" screen, which will be thoroughly explored later in this document. Initially, the previously retrieved IDB prefix is read from the HMI tag. This prefix is then utilised to inform the PLC about the presently active process screen and establish the correct interface tags' connection to the designated faceplate containers: "FC_Footer", "FC_Detail" and "FC_Manual". These faceplates also require the interface to internal HMI tags, but since these have a "Session local" scope, they can be written during the process engineering into the interface properties of the containers, freeing the screen loading events from unnecessary lines of code and making their execution quicker.

```
1  export function _1_SE_DETAIL_OnLoaded(item) {
2
3    let idbName = Tags("internal_IDBName").Read();
4
5    Tags(idbName+"_HMI_S_RANGE_POINTER_IMAGE").Write(Screen.
         ScreenNumber);
6    Tags("internal_ElementsPreviousScreen").Write(Screen.Name);
7
8    Screen.FindItem("~/SW_NavigationHandle").Visible = true;
9
10   Screen.Items("FC_Footer").Properties.Operation.Tag = idbName+"
         _HMI_S_SE_OPERATION";
11   Screen.Items("FC_Detail").Properties.Operation.Tag = idbName+"
         _HMI_S_SE_OPERATION";
12   Screen.Items("FC_Detail").Properties.CurrentScreen.Tag = idbName
         +"_HMI_S_RANGE_POINTER_IMAGE";
```

```
13    Screen.Items("FC_Manual").Properties.Operation.Tag = idbName+"
          _HMI_S_SE_OPERATION";
14  }
```

Listing 6.12: Script executed on the loading event of the elements "Detail Overview" screen.

6.2 Visualization Interface

In this section, the focus is on exploring the development achieved using the WinCC Unified environment to create a new visualization system for controlling conveyor systems in the INTRALOG standard. The section begins by presenting the implementation of the previously conceptualised screen layout. Following that, the majority of the developed screens for the interface are introduced, accompanied by explanations of their main objectives and functionalities. To provide a more detailed insight, the faceplates and the scripts developed to construct them are also mentioned. For ease of organisation, the content of the screens will be mostly presented in the order of the categories where they are inserted, based on those presented in the menu bar.

6.2.1 Screen Layout

The implementation of the screen layout described in Subsection 5.2.1 will now be addressed within the Unified development environment. The layout is created in the screen "0_CS_ScreenLayout" using several screen windows from the toolbox, as illustrated in Figure 6.15.

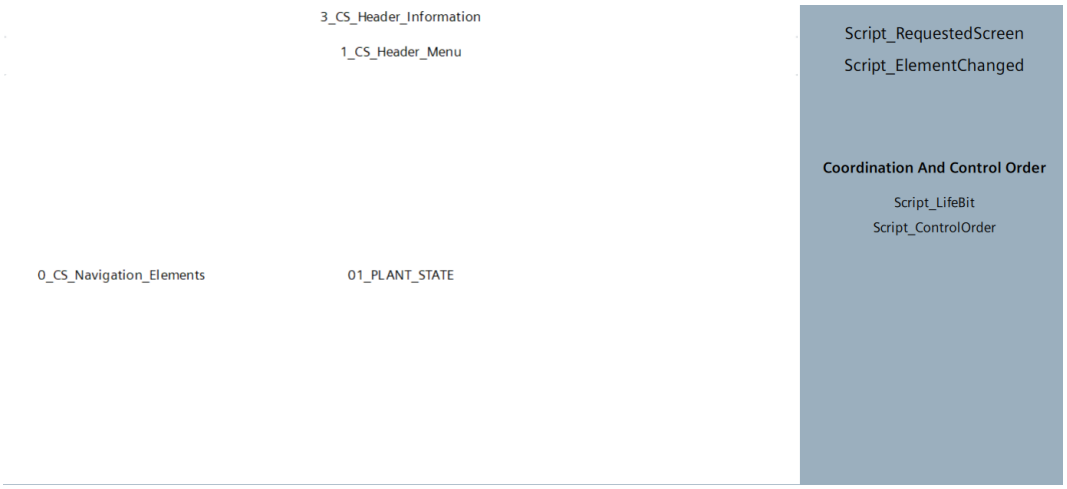


Figure 6.15: Screen layout on the Unified project.

Following the definition of the layout, the screen windows are parameterised with their corresponding screens. For instance, in the upper portion of the layout, the screen window related to the information bar ("SW_Info") holds the screen "3_CS_Header_Information", which will be presented in Subsection 6.2.1.

Based on Figure 6.16, it is evident that the various screen windows have been positioned on different layers, enabling the configuration of the layout. The higher the layer number, the closer the object appears to the foreground, as illustrated in Figure 5.2.

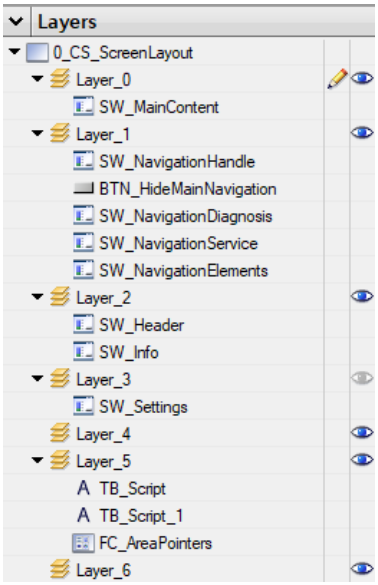


Figure 6.16: Layers of the screen layout on the Unified project.

The primary content window, initially housing the "01_PLANT_STATE" screen, is positioned in the background. This placement ensures that navigation elements like the navigation handle or side navigation menu are displayed above the process screen's content, ensuring continuous accessibility to navigation options. The main content screen window is responsible for presenting the process screens, which act as interfaces for controlling and supervising the conveyor technology. To display the selected content by the operator, the global module function named "ChangeMainContentScreen" (presented in Listing A.1, available in Appendix A) is executed in order to change the process screen.

Moving up a layer, Layer 1 focuses on the side navigation components. Positioned at the lower section is the screen window containing the navigation handle, followed by the overlapping content. To conceal the side navigation when it is open, a button covers the entire screen and is only accessible when the side navigation is visible. To make this button imperceptible, its appearance properties are adjusted to render it invisible. Three screen windows for the navigation screens are positioned and overlapped in the layout, without any specific order. As mentioned, the content

of the side navigation varies based on the process screen that the operator is interacting with. Consequently, when a specific navigation screen is visible, the others are hidden through scripting. In theory, a single screen window could handle multiple navigation screens using something similar to the "ChangeMainContentScreen" function. However, after conducting performance tests, it was observed that dividing the navigation into separate screen windows resulted in fewer lines of scripting code throughout the entire project, leading to improved overall interface performance. Since the navigation faceplates consistently maintain the correct content in the background, they are made visible as needed.

The information and menu bars can be found on Layer 2. Because they contain critical information for the entire system and are the primary means of navigation, they should appear in the foreground of the interface and be accessible at all times. There is an exception to this rule, which is why they do not assume the higher layer value on this layout. Because the settings screen is intended to be used briefly and in some cases open content that requires the use of the entire screen space, the "SW_Settings" , on Layer 3, is the last in terms of visualization content.

Layer number 4 is left empty for any future additions to the layout and also to create a separation to Layer 5. The latter holds three text boxes and one faceplate container, that, as can be seen in Figure 6.15, are positioned outside of the screen, so they will not be visible to the interface user. These screen components are merely used to trigger scripts.

In Unified, the content of a tag can be displayed in a text box, and whenever the value of the tag changes, a script can be triggered. These scripts, placed on the start screen, are designed to be universally accessible throughout the interface. The start screen is loaded when the runtime begins and remains loaded throughout the execution, ensuring constant accessibility and execution of the scripts. This technique is primarily used to trigger scripts for two main reasons. Firstly, in the current version of Unified, it is not possible to perform certain actions, such as changing the screen of a specific window, from within a faceplate. Since most screen content will be developed using faceplates to accommodate multiple instances, this technique is essential. Secondly, this approach is chosen due to limitations with Scheduled tasks. Although Scheduled tasks can trigger events based on time or tag changes, they do not support instructions from `HMIRuntime.UI`, making it impossible to change screens using this method. Additionally, using Scheduled tasks for event triggering based on tags is not optimal in the context of multiple clients, as it would require parameterizations to be adjusted when the number of clients changes between projects. One of the main goals of the INTRALOG standard is to keep parameterizations to a minimum as well as the need to make changes to the base project.

To illustrate the usage of the text box method for triggering a script, consider the

"TB_Script". The "Text" property was dynamized with the HMI tag "internal_-ScreenRequestedFromFaceplate" (Figure 6.17). Whenever the tag value changes, the script dynamization is triggered. Upon execution of the script, the tag value is checked, and if it is non-zero, it indicates that a screen change request has been initiated from a faceplate. Consequently, the "ChangeScreenRequest" function (presented in Listing A.2, available in Appendix A), stored in the global module, is invoked.

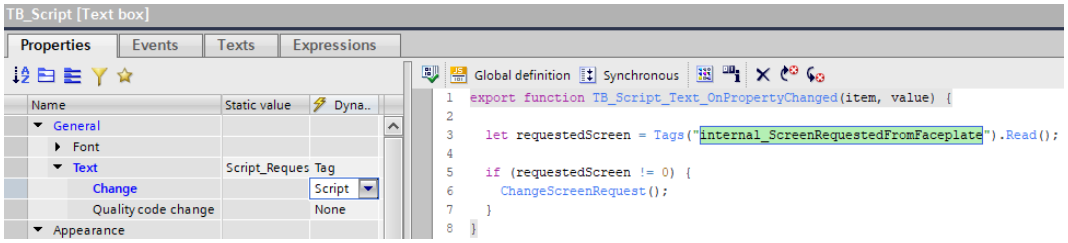


Figure 6.17: Script to trigger the change screen request from a faceplate.

The second text box follows the same procedure for triggering the script as the previous one, but in this case, it utilises the "internal_SignalElementChanged" tag and the global module function called "ElementChanged" (shown in Listing A.3, available in Appendix A).

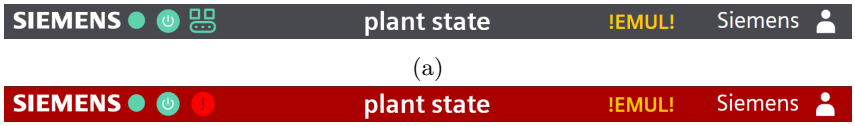
When the user selects a particular plant element, a specific set of control and supervision functionalities become accessible to the operator through the element's side navigation. To determine the available functionalities, the HMI FB provides the "IDB_HMI_S_SE_OPERATION.OVERVIEW.USABILITY" HMI tag. This tag, defined as a word data type, contains bits that indicate the supported functionalities for each element. Whenever the value of the "Usability" tag changes, it signifies a change in the selected element. Therefore, it is necessary to verify if the current screen being used is still valid or if it is no longer supported by the newly selected element. If it is invalid, it needs to be changed accordingly. To accomplish this, the "ElementChanged" global module function is utilised, and it is triggered by the change in the state of the "Usability" tag. The verification of this change is performed within the faceplate dedicated to the elements' side navigation, as will be explored in subsequent sections.

Finally, a faceplate container is employed to hold two additional text boxes responsible for handling the scripting related to coordination and control orders, as explained in Subsection 6.1.5. The interface tags used by this faceplate are written during the loading event of the start screen, as shown in Listing A.9, available in Appendix A. Additionally, the final instruction displayed on this script restricts the user from adjusting the zoom factor of the visualization interface. This limitation is intentional, as the interface has been specifically designed to function optimally on the

designated panels, and user intervention to modify the zoom is unnecessary. The appropriate adjustments have already been incorporated during the engineering phase. Allowing users to change the zoom factor could potentially lead to issues, as there is currently no designated button to revert to the original value without restarting the Unified instance. Moreover, unintended zooming might occur, especially when users are wearing working gloves, which could lead to undesired outcomes.

Information Bar

The following figures illustrate the information bar displayed during runtime. To showcase the potential dynamic changes, the first case exhibits the bar when the plant is operational, without any faults, and with the elements functioning in automatic mode (Figure 6.18a). In the second case, the appearance is altered due to a fault in the plant (Figure 6.18b).



(a)

(b)

Figure 6.18: Information bar during runtime: (a) Plant ON and automatic mode and (b) Plant ON with active fault.

Regarding the user login button, positioned on the right-hand side, its behaviour is as follows: when pressed, a script (presented in Listing A.10, available in Appendix A) checks if a user is currently logged in. If there is no logged-in user, the login dialog box is displayed. However, if a user is already logged in and this button is pressed, the user is logged off.

During runtime, the name of the logged in user, in this case "Siemens", is dynamically updated when the user changes. This is possible by linking the text property of the button to the system tag "@UserName". Whenever the value of this tag changes, a script (shown in Listing A.11, available in Appendix A) is triggered. The script retrieves the appropriate tag table based on the HMI instance, reads the name of the currently logged-in user, and writes it to the appropriate IDB of the PLC. If no user is logged in, the PLC is notified to handle the situation accordingly, such as requesting a switch to the home screen on the HMI. Additionally, the function responsible for hiding the side navigation is called because, on the home screen, the side navigation should not be visible.

Over the information bar, a "Touch area" component named "TA_ShowSettings" is positioned. This component, available on the toolbox, enables the detection of gestures and triggers corresponding actions when such events are detected.

Similar to the familiar functionality found on smartphones and other mobile devices, a swipe down gesture on the information bar presents the user with a settings screen. Additionally, as the information bar can also be positioned at the bottom of the screen, the option of a swipe up gesture is available. When either of these preconfigured gestures is detected, the screen window called "SW_Settings" (as shown in Figure 6.16) becomes visible, displaying the settings screen. This is achieved by the script provided in Listing A.12, available in Appendix A.

To present HMI instance-specific data, a faceplate is utilised. The interface tags of this faceplate are written during the loading event of the information bar screen, following a procedure similar to the one described in Listing 6.12. The faceplate handles various aspects of the information bar's appearance and functionality. Firstly, on the left side of the information bar, a circle indicator is displayed to signify the active status of the HMI instance and its communication with the PLC. On the right side of the circular indicator, two graphic views are positioned. The first view indicates whether the plant is in an ON or OFF state, while the second whether the system is operating in automatic mode or if any elements or groups are in manual mode. If a fault occurs or the emergency button is pressed, these graphics will update accordingly.

The faceplate is also in charge of displaying the name of the currently active process screen, which may be seen in the middle of the information bar. This is accomplished by reading the value of the active screen and comparing it to the indexes of a text list called "Current_Screen_Name", which stores the names of all screens. By retrieving the appropriate text from this list (as shown in Listing A.13, available in Appendix A), the faceplate ensures that the correct screen name is displayed.

Regarding the overall operational status of the plant, the background colour of the information bar is determined by the "IDB_HMI_S_GLOBAL.COLLECTIVE_-FAULT" tag. This tag indicates whether the system is experiencing a fault or if an emergency button has been pressed. When the tag is in a true state, the background colour of the information bar changes to dark red.

Finally, the yellow text "!EMUL!" indicates that the PLC is operating in emulation mode, which is utilised for training or testing purposes as explained in Section 6.3.1.

Menu Bar

The menu bar (Figure 6.19) serves the purpose of offering the interface user access to the first level of navigation. It consists of six buttons, each designed to load a specific process screen into the main content screen window. Similar to the behaviour of the information bar, when a fault or emergency stop button press is detected, the background colour of the menu bar changes to dark red. Furthermore, if a fault

is detected, the button exhibits a blinking effect with a red accent colour along its borders.

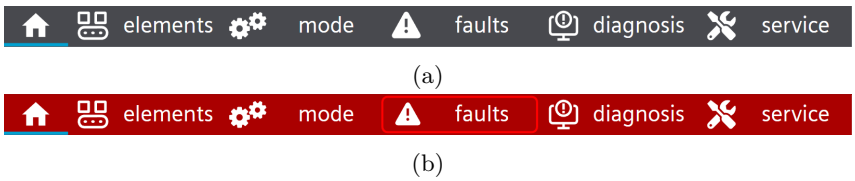


Figure 6.19: Horizontal menu bar during runtime: (a) Without active fault (b) With active fault.

To manage instance-specific dynamic elements in the HMI, particularly the blue indicator beneath each button, a faceplate is utilised. Each button is associated with a blue line, which is displayed based on the currently active process screen.

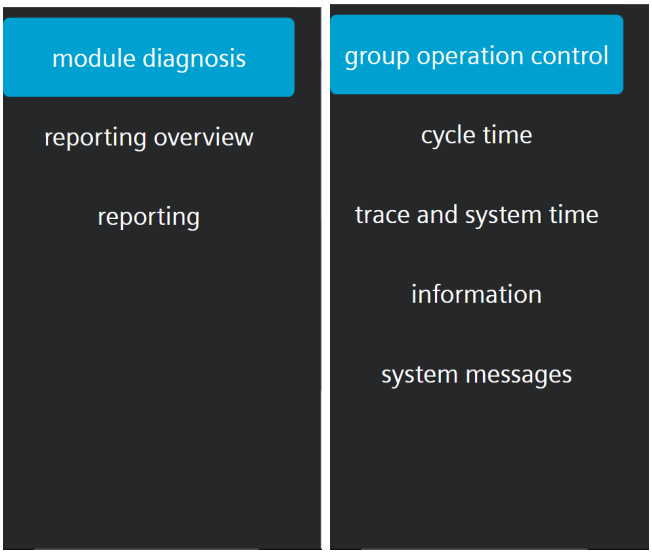
For categories with a second level of navigation, the initial button press loads a default screen. Subsequently, when the user navigates to the second level within a category, the name of the screen is stored internally. Consequently, when the same category is pressed again, the previously used process screen is loaded. This streamlines the workflow and minimises the number of button presses required to reach the desired screen. To illustrate this concept, Listing A.14 (available in Appendix A) provides the script executed when the “elements” category is pressed. In each category, the first action taken when a button is pressed is to hide the side navigation, through the global module function called “MainNavigationHide” (presented in Listing A.4, available in Appendix A). This is done because, as a principle, the sub-navigation of a specific category should only be accessible when one of the corresponding process screens is being used.

In Figure 6.20, an alternative version of the menu bar is shown, presented in a vertical format. It is employed when the user selects a different layout in the settings screen. The vertical menu bar shares the same properties as the horizontal bar previously described. However, due to space limitations, the names of each category are not displayed.

Figure 6.20: Vertical menu bar.

Navigation

A side navigation screen was developed for the elements, diagnosis, and service categories. Each of these screens features a faceplate that houses the navigation buttons. The diagnosis and service navigation are straightforward since they offer static options that can be predefined during the engineering process (Figure 6.21).



(a) (b)

Figure 6.21: Side navigation: (a) Diagnosis (b) Service.

In contrast, the elements navigation is more complex as it should provide dynamic options that are intrinsically related to the selected element, as explained previously. To achieve this functionality, a script is triggered within the faceplate whenever the usability tag changes. This script is responsible for managing the visibility of each navigation entry. In order to associate each bit of the usability word with the corresponding entries, a naming convention has been established for the screen components. For example, in the case of the "Multicontrol" screen, which does not rely on any bit of the usability word, the entry component is named "TB_-Entry". On the other hand, the "Control" entry depends on bit 0, hence its name is "TB_Entry_0". The suffix in the name of each entry is directly associated with the corresponding bit of the usability word. When a usability bit is associated with multiple navigation entries, the same ideology is applied. For instance, there are three data tracking screens available, and they correspond to bit 7. Therefore, the entry names for these screens are "TB_Entry_7", "TB_Entry_7_1", and "TB_-Entry_7_2" (Figure 6.22).

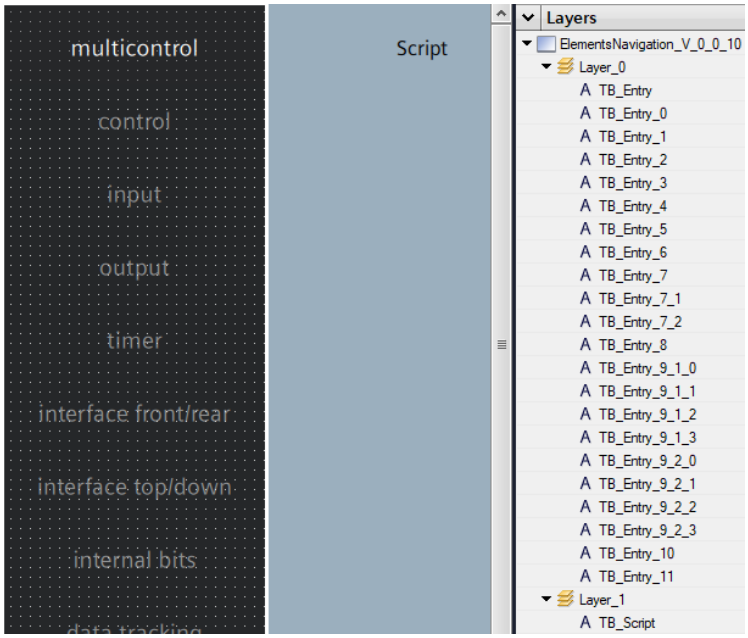


Figure 6.22: Faceplate to handle the entries on the elements navigation.

To align with the new approach of using a side navigation to provide users with the functionalities inherent to each single element, it made sense to incorporate the options of the "Special" screens into the navigation itself. In the old interface, on the elements overview screen, there was a "special images" option available. By selecting this option, the screen would change to another overview screen with a specific set of options based on the selected element. With the new visualization interface, these "Special" functionalities are now accessible through the side navigation, and the list of entries dynamically adjusts based on the number of available options to allow quicker access to these functionalities.

Once the entries on the faceplate have been named and positioned, a text box located outside the visible area of the faceplate serves to trigger the script (presented in Listing A.15, available in Appendix A) when the element's usability is changed. To start with, two variables are initialised. The "screenHeight" variable stores the height of the list after all the relevant entries are made visible. The initial value is set to 68, considering that each entry occupies 60 pixels and the "multicontrol" entry is visible by default. Additionally, the first entry starts at 8 pixels from the top for visual appeal. The second variable, "entryHeight", is a constant representing the entry height, ensuring that the code remains portable and easily adaptable if the height is modified in the future. Next, the value of the usability word is retrieved and stored in the "overviewUsability" variable. Afterwards, the interface tag "ElementChanged" is set to one, which will be connected to the "internal_-SignalElementChanged" tag. This setting triggers the execution of the global module function "ElementChanged", as explained in the Subsection 5.2.1. Subsequently, a

loop is utilised to iterate through each bit number of the usability word. Within this loop, if the current bit being evaluated is set to true, the corresponding entry is made visible and positioned by setting the "top" property to the value stored in the "screenHeight" variable. This variable is updated each time a new entry becomes visible (Listing 6.13). In case the usability bit is set to false, the entry's visibility property is also set to false, restricting navigation to that specific process screen.

```
1 Faceplate.Items("TB_Entry_" + i).Visible = true;
2 Faceplate.Items("TB_Entry_" + i).Top = screenHeight;
3 screenHeight = screenHeight + entryHeight;
```

Listing 6.13: Instructions executed when elements navigation entry usability is available.

At the end of the script, the value of "screenHeight" is written to the faceplate interface tag connected to "internal_ElementsNavigationHeight." This allows the value to be accessed externally from the faceplate and utilised to adjust the height of the elements navigation screen. This adjustment ensures that there is no unnecessary empty scrollable space at the end of the navigation entries list. For further details, and a complete version of the script, refer to Appendix A.

Each entry is created using a text box, and apart from the properties that are modified through scripting, two additional properties are manually defined. The background colour is set to depend on the tag that indicates the currently active process screen, ensuring that the background appears blue when the corresponding screen is active. Furthermore, for security purposes, the "operator control - allow" property is configured to restrict access to the entry only if the corresponding bit from the usability word is active. Although the script guarantees that the entry will only be visible when appropriate, configuring this property adds an extra layer of security. Lastly, when an entry is pressed, the desired process screen is requested by writing its value to the "internal_ScreenRequestedFromFaceplate" through the interface tag.

Some details concerning the screen that will use the elements navigation faceplate should be provided. One of them is the fact that the "internal_ElementsNavigationHeight" tag dynamically adjusts the screen height. In Unified, when a screen element is positioned within the screen area but is bigger than the screen itself, a scroll bar shows during runtime. This is very beneficial because the faceplate for the entries can have up to 1328 pixels of height and the screen only has 480 (Figure 6.23), so when the entries list is bigger than the screen, the desired scroll functionality on the side navigation is possible.

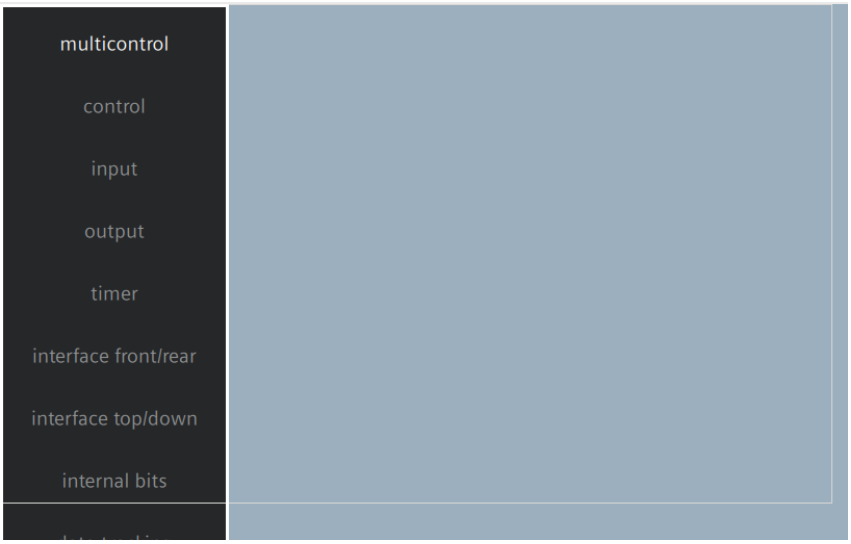
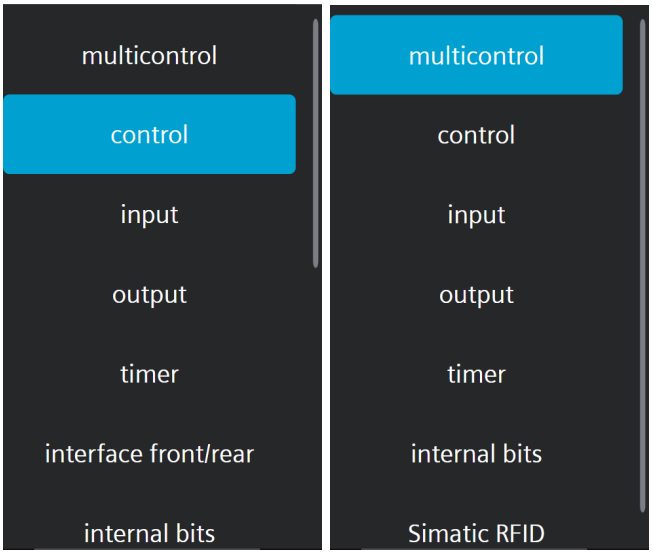


Figure 6.23: Elements navigation screen in engineering.

To conclude the discussion on the elements navigation, two examples are presented in Figure 6.24. In the first, the currently active element is a conveyor, resulting in a large number of entries on the side navigation, as perceptible by the extension of the vertical slider. Conversely, in the second figure, the selected element is a SIMATIC RFID scanner, resulting in a dynamically adjusted navigation list with only seven process screens available.



(a) (b)

Figure 6.24: Elements side navigation: (a) Conveyor (b) SIMATIC RFID.

To access the side navigation, a navigation handle is positioned on one of the bottom corners of the interface, based on the operator's preferences. This handle

serves as a button that invokes the global module function “MainNavigationVisible” (refer to Listing A.5, available in Appendix A) when pressed, displaying the appropriate navigation based on the currently active process screen. The handle is visually represented by an icon in the style of a “hamburger” menu (Figure 6.25). This icon choice is common in modern applications and is also used in projects utilising the HMI Template Suite.

Figure 6.25: Navigation handle.

6.2.2 Settings

By using the gesture area positioned over the information bar, the user can access the settings screen (Figure 6.26). This screen aims to provide quick access to panel settings, as well as giving access to features that are not particularly related to the process of controlling the conveyor system. Below the time and date, a white region denotes a area destined to the position of quick access functionalities.

Figure 6.26: Settings screen.

The first option is related to the currently logged-in user; it not only gives the name of the current user but also, when pressed, provides access to the login dialogue pop-up.

On the right, a “cleaning screen” button appears. This feature will show many screen components that cannot be pressed for a predetermined amount of time; in this case, the value was set to five seconds for testing purposes. This screen’s goal is to allow users to manually clean the panel screen without mistakenly pushing any buttons. When the quick access button is pressed, a script is executed, displaying

the appropriate screen components for the cleaning screen. When the time specified for the timer runs outs, a callback function is triggered, returning the user to the default layout of the settings screen (Listing A.16, available in Appendix A).

While pressing the "lamp test" button, the signal lamps of the CP that the HMI instance is controlling are turned on. This serves to perform the validation that the lamps are working, for safety purposes.

Following that, pressing the "open user manual" button opens a PDF and automatically changes it to the appropriate page based on the currently active screen number.

The last button is the "show panel configurations". When pressed, it puts the runtime in the background and gives access to the UCP operating system, where it is possible to setup network parameters, reboot the panel, open applications, among others. This feature is protected by an authorised log in, since it should not be accessed by common operators. By default, the UCP displays a bar on the bottom of the screen that allows access to the operating system. However in the final implementation this feature must be disabled and the access to the operating system features should only be made by the quick access button on the settings screen, by authorised users.

The second row of the quick access region was left empty because the base project doesn't require any additional features. However, this space can be used to implement new shortcuts that can be customised for each customer project.

Outside the white area, on the right of the screen, an option to change the system language is provided. In the past, on the WinCC Advanced project, to change the system language the user had to press a country flag on the information bar, in order to cycle through the available languages. On the new solution, the PLC code was slightly changed in order to accommodate the fact that now all the languages are listed and the user can simply choose the preferred one.

Near the bottom of the screen, a slider is used to set the display brightness, using the `HMIRuntime.Device.SysFct.SetBrightness(value)` instruction, where value will assume the "internal_DisplayBrightness" tag that is updated whenever the slider is used.

Occupying the bottom portion of the screen, a touch area component is positioned to detect the swipe up gesture, concealing the settings screen.

Lastly, the "panel settings" button will change the settings screen to the one displayed in Figure 6.27.

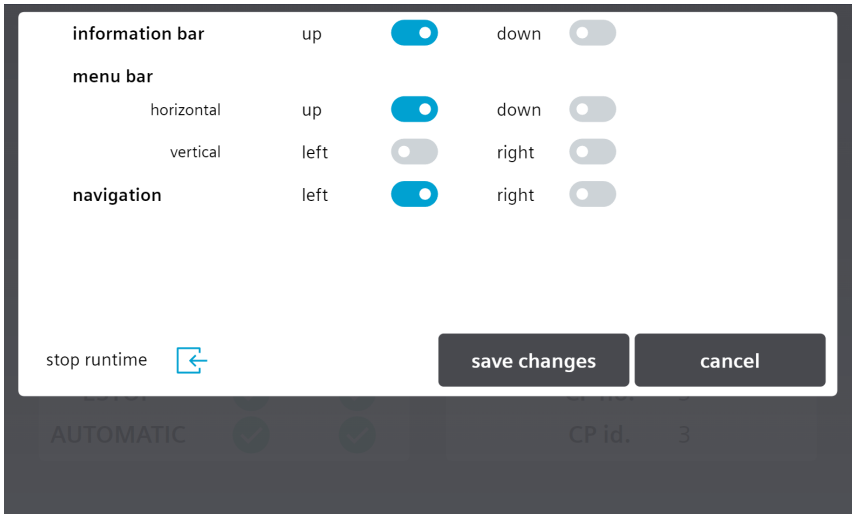


Figure 6.27: Panel settings screen.

By the usage of switches, the user can customise the screen layout, like setting the position of the information and menu bar, as well as the side navigation and handle. The switches will change the value of the "internal_InfoBarPosition", "internal_-MenuBarPosition" and "internal_NavigationPosition" tags. Then, when the "save changes" button is pressed, these variables are passed as parameters to the global module function "ChangeLayout" (refer to Listing A.6 for further details), which will adapt the layout based on the user choices. If, at any point, the user wants to leave this screen and return to the main settings screen, the "cancel" button must be pressed.

Finally, on this screen, the "stop runtime" button available on the left bottom corner will open a pop-up with the option to stop the runtime (Figure 6.28). This feature can only be performed by an authorised user.

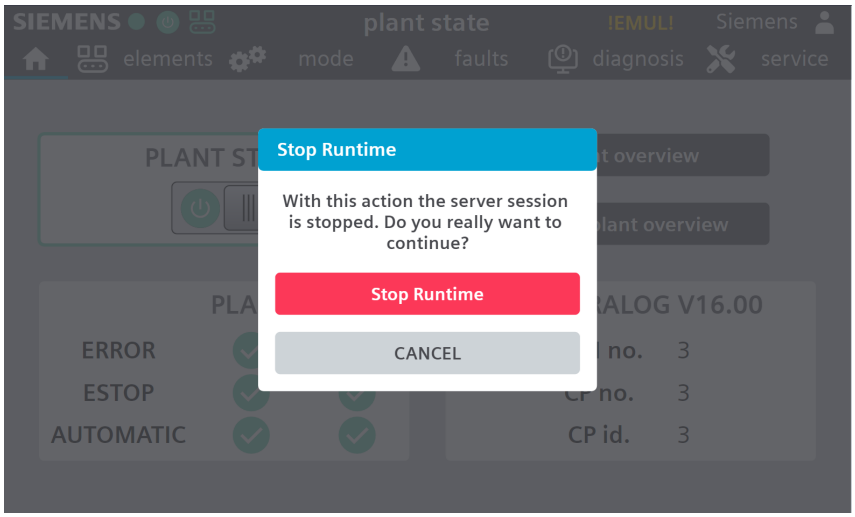
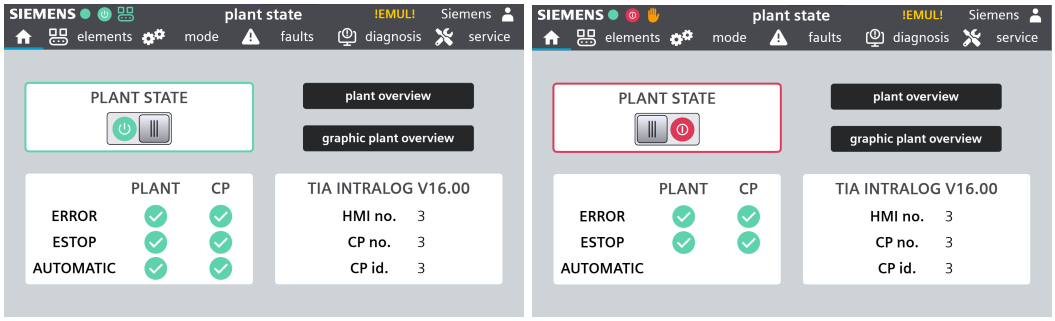


Figure 6.28: Stop runtime pop-up.

6.2.3 Plant State

The initial screen loaded in the visualization system is the "Plant State" (Figure 6.29). It is also the screen that the PLC requests the HMI to switch to when the current user logs off. Consequently, it serves as the home screen and is represented by the first button on the menu bar.



(a) (b)

Figure 6.29: Plant state: (a) ON (b) OFF.

Upon first glance, the screen content can be divided into three distinct areas, along with two additional buttons.

The first area provides the user with information about the plant's ON/OFF state. To visualise and control this state, a switch is used. The border surrounding this content area also provides visual feedback about the state. A static green border indicates that the plant is ON, while a slowly blinking red border indicates that the plant is OFF. Additionally, when the plant transitions from one state to another, the background of the content area is animated to reflect the change, as the process is not instantaneous. During the transition to an ON state, the background blinks green and during the transition to an OFF state, the background blinks red.

Beneath the first content area, there is another section designed to provide the user with a quick assessment of the plant and CP status. This section utilises two separate SVG elements to inform the operator about errors, emergency stops, and the process execution mode in automatic mode. Under the "Plant" column, the status pertains to all the groups of elements in the plant. On the other hand, under the "CP" column, the information provided specifically relates to the groups that the HMI can control.

The final content area briefly mentions the INTRALOG standard and the system version. Below that, three fields provide information about the HMI and CP. The "HMI no." field displays the number of the IDB on the PLC that is supplying data to the Unified instance. Below that, the "CP no." indicates the number of the CP that the HMI is controlling. Lastly, "CP id." represents the number identifier of the Connection Box to which the Mobile WebClient is connected. This number is crucial for the PLC to determine which CP this visualization instance should control. In

the case of a fixed HMI panel, such as the MTP700 UCP, a Connection Box is not required. Therefore, the "CP id." field will be the same as the "CP no." field.

Finally, on this process screen, there are two navigation buttons available. These lead to screens that provide two different types of overview for the groups of elements, as demonstrated below.

Plant Overview

The "Plant Overview" screen (Figure 6.30) provides a comprehensive display of the plant's elements, offering both visual and textual representations of their operational status. For instance, referring to the illustration, Group 1 is operating in automatic mode, while elements "01RF05" and "03RF01" are experiencing faults. Additionally, Group 3 has been instructed to halt and is now in a defined position. On the other hand, the elements of Group 2 are operating manually. These representations are the most commonly encountered, although others, such as indicating that the plant is turned off or that a specific element has been instructed to stop to transition the plant to an off state, are also taken into account. To navigate through the available elements, the user can utilise scroll buttons located on either side of the white content area. These buttons allow the user to browse through all the elements based on the applied filter.

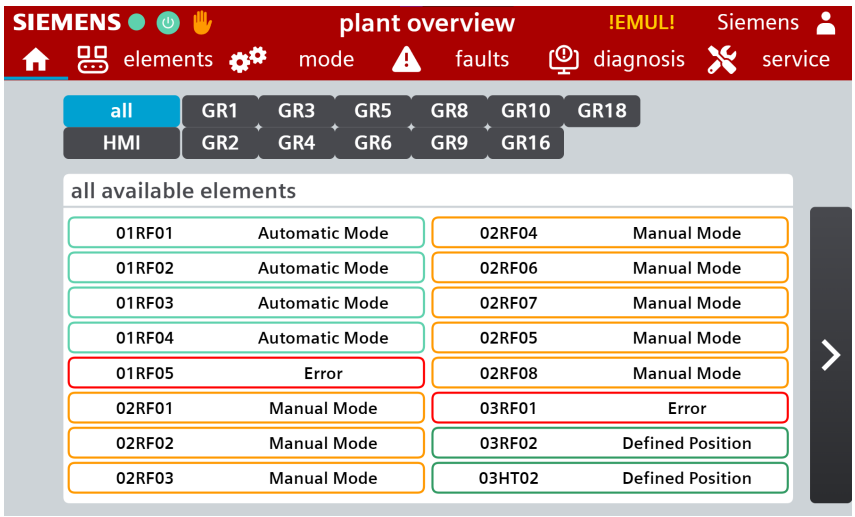


Figure 6.30: "Plant overview" screen.

The top buttons can be employed to filter the elements by group. Each group that the CP controls is associated with a button. When a particular group, for example, Group 1, is selected, only its elements will be displayed in the content area. Simultaneously, the content header will be updated to reflect the chosen group number and name.

This screen also allows users to directly access the elements screen of a specific element. When a user presses an element from the content area, a request is made to switch the process screen to the most recently visited elements screen. Additionally, at the bottom of the screen, a touch area is positioned. If a swipe-up gesture is detected, the process screen will be changed to the last accessed elements screen. Although this functionality serves the same purpose as pressing the "elements" button on the menu bar, having quick access at the top and bottom of the screen can be convenient and intuitive. This is particularly useful and will be explained further later on, as the elements screens also offer a similar quick access gesture.

Graphic Plant Overview

In addition to the "Plant Overview" screen, there is an option to create a graphical representation of the plant. However, this requires manual planning and creation of each screen because there is no automatic generation available due to the plant layout varying according to the customer's requirements. The standard offers an emulation mode, and creating base layouts for the model project proves beneficial for training purposes or validating the development through simulation. It's important to note that the functionalities in the content areas, such as placing or taking a pallet or starting conveyor movement, are designed solely for simulation and won't be provided to customers. To demonstrate the capabilities of Unified and its dynamizable properties, two graphic representations were developed for groups one and five (Figure 6.31).

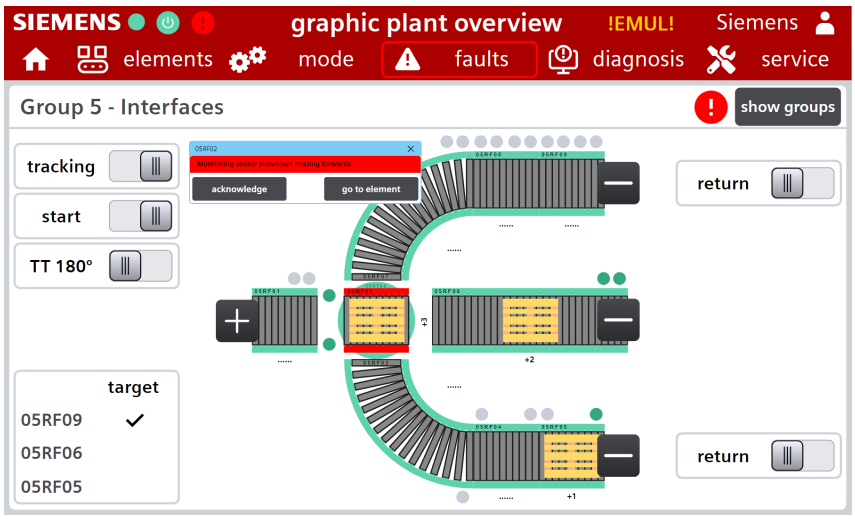


Figure 6.31: "Graphic plant overview" screen example.

Each screen in the "Graphic Plant Overview" features a content area with the group number, name, and an error indicator/acknowledge button on the header regarding the entire plant. A show/hide groups button is also available, allowing

the user to select another group and change the representation. This feature helps save screen space by not continuously displaying the group selection row.

For each group, a manual construction of the plant representation is performed, utilising conveyor SVG, circular indicators for sensors, and text boxes for data tracking.

When the user presses a conveyor, the interface automatically switches to the last-accessed elements process screen with the selected element already loaded. However, if there's a fault with the conveyor, a pop-up appears with the specific error related to the element, granting the operator the option to acknowledge the fault or go to the last element's process screen. This functionality is achieved using the script presented in Listing A.17 on the press event of the SVG.

Multiple pop-ups can be simultaneously opened if more than one conveyor is at fault. For this reason, when the pop-up faceplate is loaded, the selected element label is stored on a local faceplate tag, not only to apply it to the alarm control filter to display an element-specific fault, but also to make it possible to press the "acknowledge" and "go to element" buttons without applying these instructions to the last element pressed; this is achieved through the script presented in Listing A.18, available in Appendix A).

6.2.4 Elements

The following Sub-subsections cover the development regarding the construction of the screens dedicated to the interaction with the elements that make up the conveyor system.

Detail Faceplate

A "Detail" faceplate (Figure 6.32) was created to provide the user with a graphical representation of the element with which he is currently interacting.

In addition to the graphical representation, the faceplate provides the operator with a condensed view of the element operation parameters, which can be evaluated more thoroughly by accessing multiple different screens. This reduces the number of process screen changes, while providing a more complete evaluation of the working state of the single element through most screens.

In contrast to the old visualization interface, where this representation was far less complete and only available on one screen, the "Detail" view will be available on most process screens under the "elements" category.

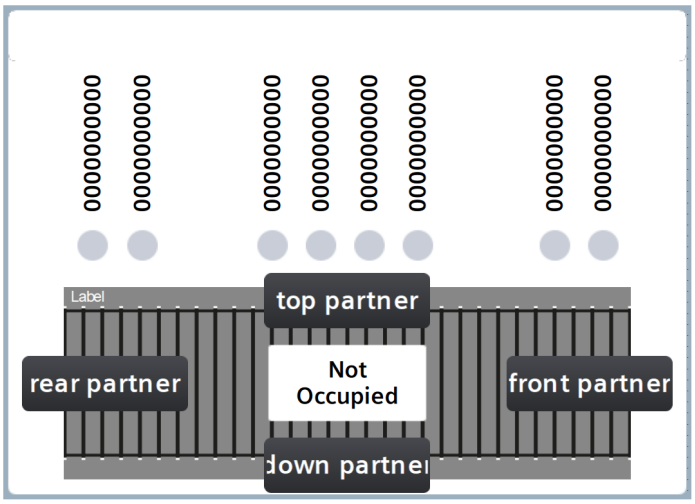


Figure 6.32: "Detail" faceplate.

The content of the faceplate can be divided into several key components:

- **Dynamic SVG** - A graphic representation of the conveyor, providing the user with several informational points through the use of colours to represent states and outputs, particularly the conveyor operation mode and the drive running state.
- **Occupied state** - Use of a text box over the conveyor, with an Occupied or Not Occupied message in order to identify if the conveyor is currently carrying a load.
- **Partners buttons** - Positioned over the conveyor SVG, it serves to inform the operator about the labels of partners, which are conveyors connected to the one being analysed.
- **Inputs** - Display multiple input states through a textual label and a circular indicator that changes colour to inform about the active or not-active state of a signal.
- **Fault bar** - On the top portion of the faceplate, an "Alarm Control" screen component is positioned and visible when the single element is at fault. The properties of the alarm control were tweaked in order for it to display only one line without any columns. This allows the fault bar to provide a single element specific alarm message in the format of a header.

In the case of a single element that does not have a detail screen, such as a scanner, all of the above contents are disregarded, and the faceplate will only present a text box with the name of the element in a central position.

In order to have a unique faceplate that contemplates all types of elements, the content from each type was placed on a different layer of the screen in the engineering environment. Then, by using a script (presented in Listing A.19, available in Appendix A) that evaluates the type of the currently active element, it is possible to manage the visibility of each layer. During runtime, the user will only view the content that is inherent to the selected element. So, this means that when a specific type of element is active, only the layer that has the screen components referring to it is visible. The script that manages this faceplate starts by reading the label of the current element in order to apply a filter to the alarm control, ensuring that the error shown on the fault is specific to the selected element.

Then the tags that provided information about the type of the element are read and stored in variables that are used on conditions to evaluate what screen layers must be visible at a time in order to provide the correct content based on the active element. As an example, in the first condition, the detail usability bit is tested in order to verify if the element supports a graphical representation. On the second, it is evaluated if the element is a roller conveyor; if it is, "Layer_1" is made visible since it's the one with the content dedicated to this type of element. Lastly, the script is triggered whenever one of the PLC tags experiences a change in its value.

To close the topic of the "Detail" faceplate, it's important to mention the dynamic capabilities of the SVG used to graphically represent the conveyors. On the "Interface" properties of the SVG (Figure 6.33), some parameters are dynamized through tags and expressions. For example, with the HMI tag linked to the "RollerColor" property, it is possible to make the rollers of the conveyor flash green when the drive is running.

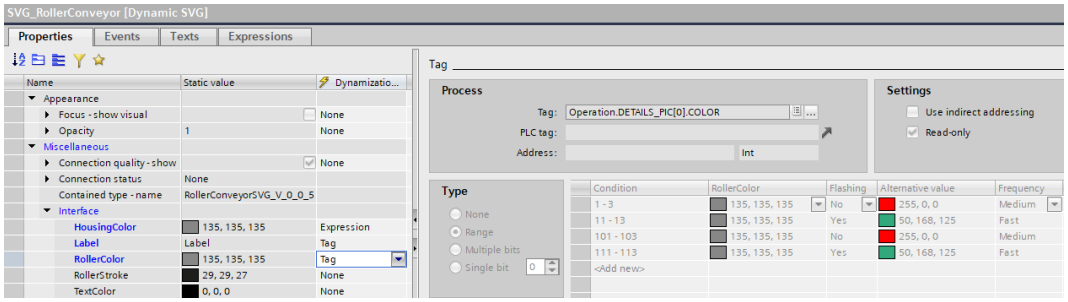


Figure 6.33: Roller conveyor SVG properties.

Furthermore, the colour of the conveyor housing is determined through a set of conditions defined under "Expressions" (Figure 6.34). There are three conditions set for the roller conveyors. The first condition activates when the conveyor is at fault, causing the housing to flash red at medium speed. The other two conditions apply when the conveyor is not faulty; in this case, the housing colour is either set to orange or green, depending on the operation mode of the element.

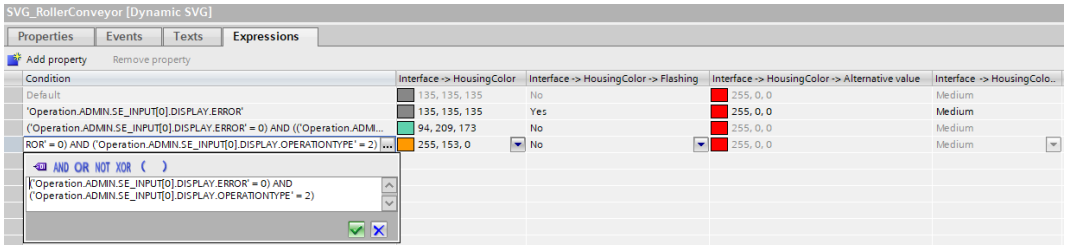


Figure 6.34: Roller conveyor SVG expressions.

Default Footer Faceplate

In order to achieve a uniform look on all the element screens, the footer used on most of them is shared through the use of a faceplate (Figure 6.35). The content area provides the user, in a central area, with an I/O field from which the operator can not only see the label of the currently selected element but also, by pressing this zone, change the element by using the keyboard of the HMI. Furthermore, on the extremes of the I/O fields, two scroll buttons are positioned in order for the user to go through the available elements. After the desired element is currently active, near the left corner, the user has access to five parameters. On the top, the name of the FB used to parameterise the element, presenting an abbreviation of some of its characteristics. Then, on the bottom, three icons are provided. On the first, the indication of the release state of the element is signalled by the black or green representation of a conveyor. A black conveyor means that the conveyor has no release and therefore cannot operate freely. The second icon, in the middle, informs the operator about the error state of the single element. If a fault is active, the user can press this icon to acknowledge it. Lastly, the third icon tells about the operational state of the element, contemplating a representation for: automatic, manual, no state, and emergency stop.

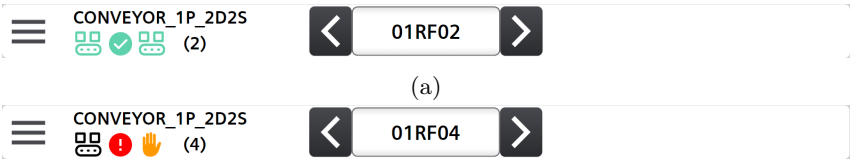


Figure 6.35: Elements default footer: (a) Element with release, no error and in automatic mode (b) Element with no release, error and in manual mode.

Lastly, overlapping, near the bottom, the element label, a touch area screen component was positioned in order to provide the operator with quick access to the plant overview. As perceivable from Listing A.20 (available in Appendix A), when a swipe-up gesture is detected the process screen is automatically changed to the

"Plant Overview" screen. This will provide the user with a list of all available elements, and when one is pressed, the process screen is changed back to the previously accessed with the newly selected element already loaded.

Control and Internal Bits

Since the representation of the "Control" and "Internal Bits" content is similar, a single faceplate was developed to handle both. The faceplate represents a content area that holds six entries and accommodates scroll buttons on the bottom to go through them, in case more than six are available. Both screens will use this faceplate for output and input purposes, so each entry will have on the right a button that will not only serve to determine the entry state but also to toggle its value if possible. In addition to this button, the entries areas are composed by the entry name itself, provided by a PLC tag, and a border delimiting and indicating the active or inactive state of the parameter. In this case, as explained before, since these are system or user-controllable states, the blue accent colour is used to represent the active state. As can be seen from the faceplate engineering (Figure 6.36), the buttons are represented at a lower opacity level because their control capabilities are dependent on the state of another PLC tag. This means that if the operator is not able to control the parameter, the pressing capabilities of the buttons are suppressed. Furthermore, during runtime, the opacity of the screen component is also reduced to inform the user that it is not possible to interact with it.

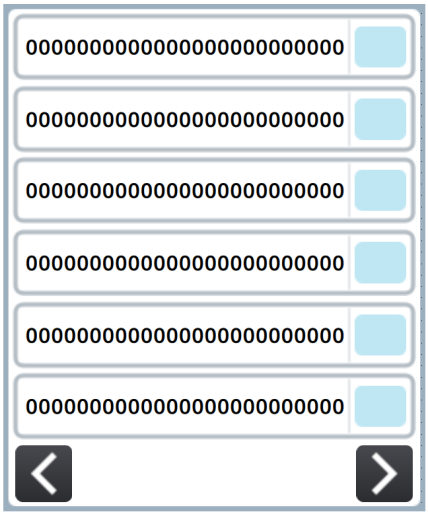
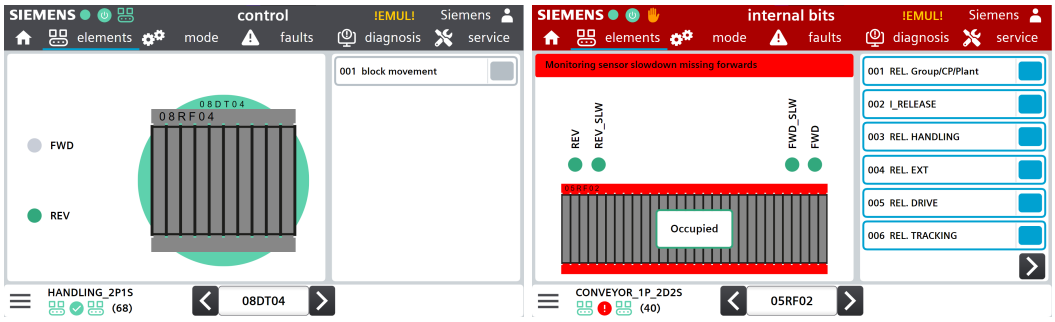


Figure 6.36: "Control" and "Internal Bits" faceplate.

The "Control" screen (Figure 6.37a) is used to enable or disable certain pre-parameterised functions. On this screen, the operator can usually find the option to block the movement of the element, like restricting a conveyor from transferring or taking a pallet or container. When the single element is a scanner, for example,

it's possible to disable it. In the case of the turn table presented in the figure, the "block movement" is inactive; by pressing the button near the tag entry, the user would enable this functionality, which would be represented by the button and the border turning blue.

Concerning the "Internal Bits" screen (Figure 6.37b), it provides access into the internal parameters of a single element. A key objective of this screen is to facilitate the evaluation of the "release" bits, as this state is influenced by various factors, such as higher-level control parameters, interface connections with top and bottom partners, motor drive status, and more. By examining the release bits, one can determine if the element is available for operation without any restrictions. In the showcased scenario, all the release bits are in a true state, indicating that the element can operate without limitations. In certain configurations, regarding blocks and plant setups, specific states such as "occupied" can be modified directly from this screen.



(a) (b)

Figure 6.37: (a) "Control" screen (b) "Internal Bits" screen.

Inputs and Outputs

Using the approach mentioned earlier, the "Inputs" and "Outputs" screens share a faceplate (Figure 6.38) to show I/O status. This faceplate has space for eight entries, each displaying a signal's name and a circular icon for its state. Since these signals are not user-controlled, the entry area is smaller compared to the "Control" and "Internal Bits" faceplates. The circular indicator uses dark green for active state and border delineation.

Furthermore, at the area's bottom, three buttons are present. Two are at the edges, allowing scrolling through entries if more than eight exist. In the middle, a switch icon toggles between descriptive and parameterised names for input entries. Descriptive naming is the default as it's more intuitive.

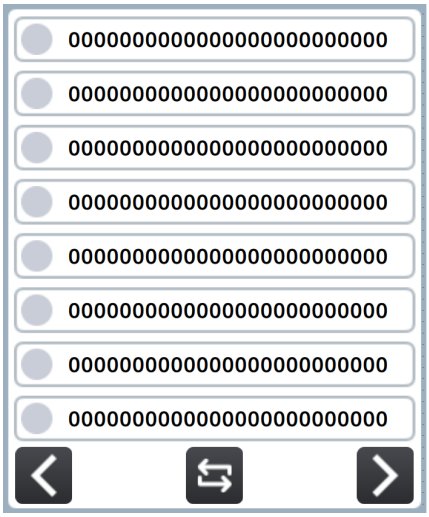
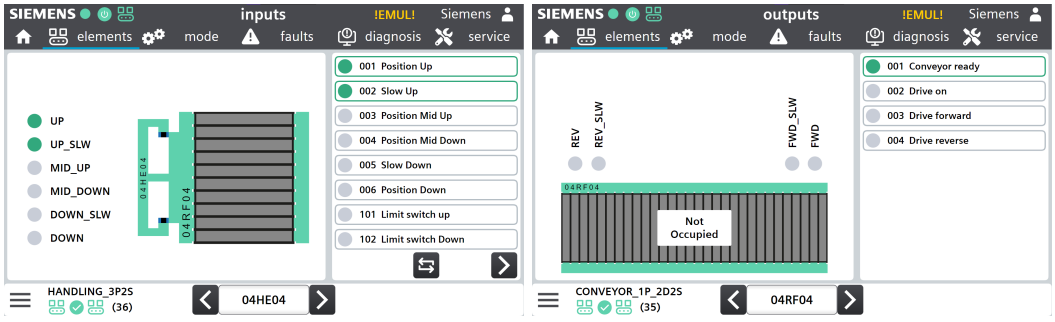


Figure 6.38: “Inputs” and “Outputs” faceplate.

The “Inputs” screen (Figure 6.39a) shows a complete list of input signals for the chosen single element. It includes signals from the element’s detailed view, often with more descriptive names. This list might even have extra sensors not visible in the detailed view. For instance, the illustrated lifting table has six position signals, but the inputs list could have more signals that aren’t shown.

Regarding the “Outputs” screen (Figure 6.39b), it mainly shows signals from the drive that controls the conveyor’s motion. These signals indicate the drive’s status and the conveyor’s direction. For a scanner element, the “Outputs” may reveal if the scanner is ready to read, if data is available, or if a reading request is done.



(a) (b)

Figure 6.39: (a) “Inputs” screen (b) “Outputs” screen.

Interface

To analyse the signals exchanged between a conveyor and its partners, two process screens are available. One screen is dedicated to evaluating the states of the front/rear partner interface, which is typical for conventional conveyors. The other focuses on the top-down interface, primarily used for elements that can have partners

positioned above or below, such as a lifter or a rotating table. These new interface screens have undergone visual redesign and offer additional functionalities compared to the old visualization solution.

One notable improvement is the integration of the detail faceplate in the middle of the screen, using the nested faceplate capabilities of Unified. By pressing the labels of the partners, the selected element is automatically loaded and, if necessary, the process screen is changed accordingly. Furthermore, if the elements do not have partners, the screen components designated for them are automatically hidden, streamlining the evaluation of the screen content.

On the "Conveyor Interface" (also referred to as "Interface Front/Rear") screen (Figure 6.40), the operator can assess the coupling signals between the element selected in the footer and the partners automatically loaded on the available content areas on each side. Below the graphical representation of the selected element, an arrow with a text box indicates the moving direction of the conveyor. If the conveyor supports movement in both directions, scroll buttons within the content area are visible, allowing the operator to change the movement direction and evaluate the interface signals accordingly.

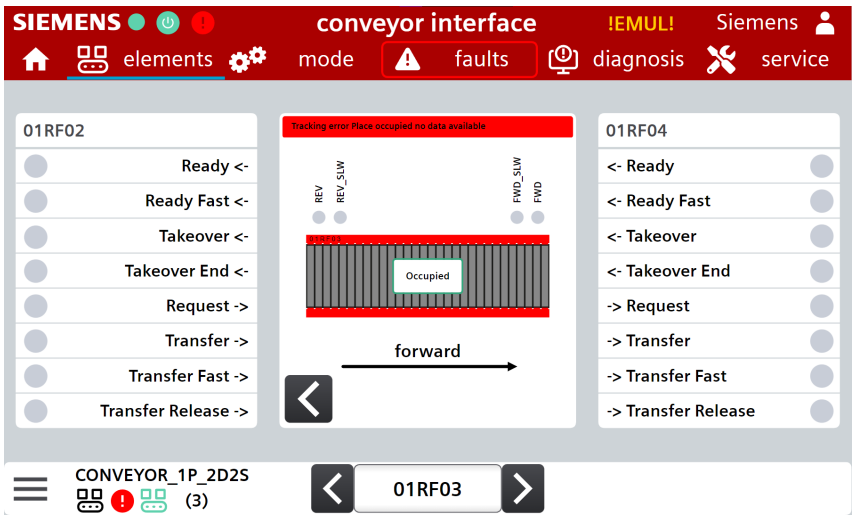


Figure 6.40: "Conveyor Interface" screen.

Regarding the "Handling Interface" (often referred to as "Interface Top/Down") screen (Figure 6.41), the working principle is similar to what was previously described. However, in this case, the partners are positioned at the top and bottom of the selected element. For instance, consider the element "03HT02," which functions as a vertical lifter responsible for managing two roller conveyors. Specifically, it handles the "03RF03" conveyor at the top and the "03RF02" at the bottom. If the selected element was a turntable, only a top partner would be visible since, based on the physical structure of this type of element, a bottom partner does not exist.

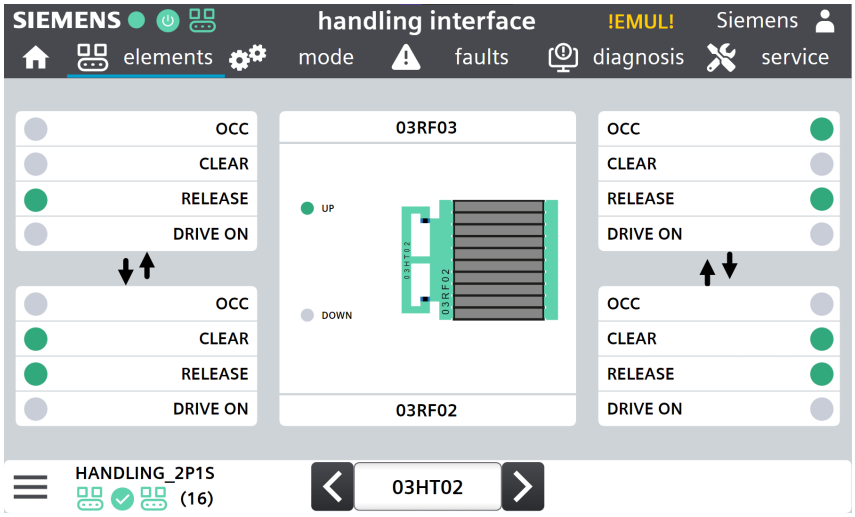


Figure 6.41: "Handling Interface" screen.

Detail Overview

In the old visualization interface, the "Detail Overview" screen was the only one to contemplate a graphical representation of the conveyor selected for operation, and therefore it was from this screen that the operation in manual mode of the single element was possible. Though the "Detail" faceplate is now accessible through multiple screens, the ideology behind the "Detail Overview" screen remains the same, as depicted in Figure 6.42.

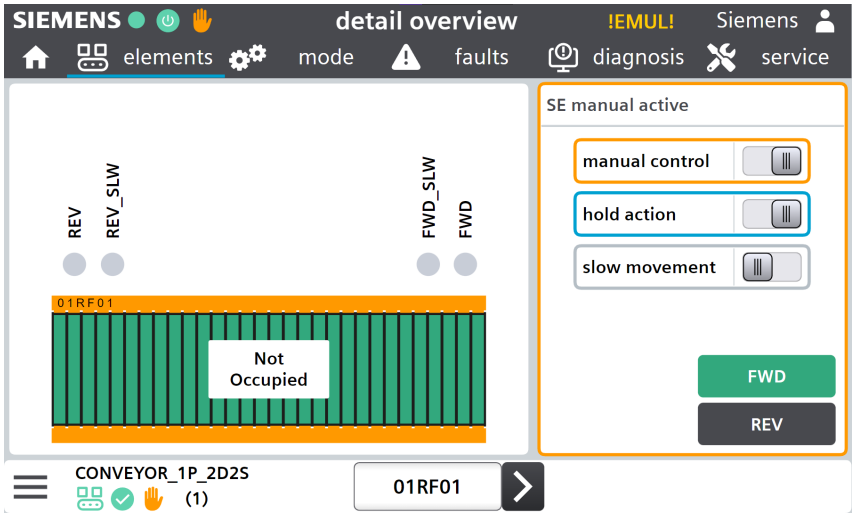
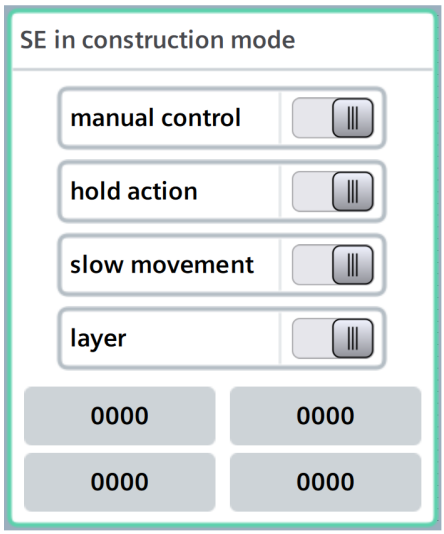


Figure 6.42: "Detail Overview" screen.

In addition to the default elements footer and the content area dedicated to the graphical representation of the conveyor, on the right side of the screen, the user has access to a faceplate (Figure 6.43a) that constitutes another content area dedicated to the operation of the single element. Surrounding the content area, a border

changes colour to visually represent the state of the element, either green or orange for the operation mode, or red for errors or warnings that the operator must consider carefully. The header represents the same states as the border but through a more detailed textual message. In addition to informing about the operational state, the header can hold multiple error or warning messages that are defined under a text list (Figure 6.43b). The text is retrieved from it by means of the entry index that is chosen by a PLC tag; the script used for this functionality uses the same logic as the one stated in Listing A.13.



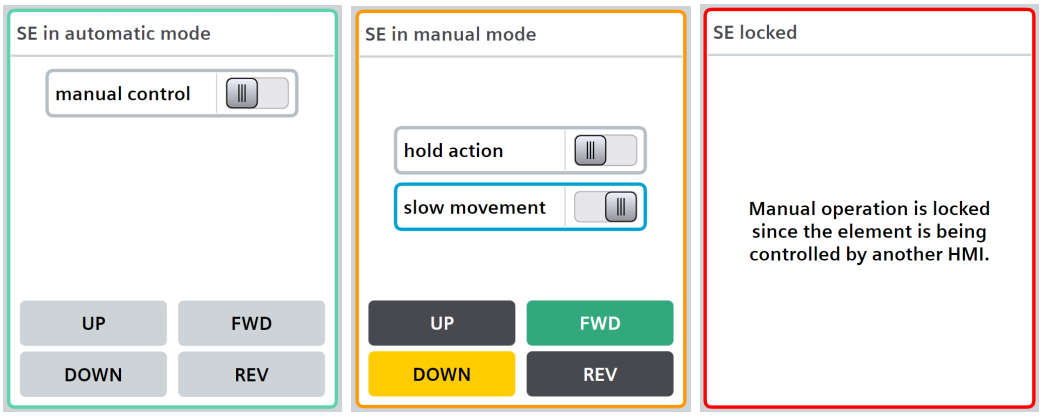
(a)

Bit num...	Text
0	SE not found
1	group not found
2	part not found
3	type not found
4	operation restricted
5	SE locked
6	part not found
7	no e-stop release!
8	SE not in manual mode
9	SE manual active
10	text-offset must <> 0
11	number not found
12	position-offset 0!
13	SE has no space!
14	DS not occupied
15	plant OFF
16	SE in manual mode
17	SE in automatic mode
18	SE in construction mode
19	no release

(b)

Figure 6.43: Detail overview faceplate: (a) Faceplate (b) Header text list.

When the element operates in automatic mode (Figure 6.44a), users can switch to manual control. This activates control options and movement buttons that are visible but disabled in automatic mode. In contrast, if the element already operates in manual mode (Figure 6.44b), and it belongs to a manual-operating group, the operator can directly use the controls. Among a diverse list of states, like emergency stop or no release, one of the most relevant for this project is “SE locked” (Figure 6.44c). This means that the single element is currently being controlled by another HMI; therefore, manual operation is restricted for safety reasons.



(a) (b) (c)

Figure 6.44: Detail overview element operation: (a) Automatic mode (b) Part of group in manual mode (c) Operation locked.

Besides the manual mode switch, the user has three more switches. The "hold action" switch maintains movement without continuous button pressing. "Slow movement" allows slower execution for better control. The "layer" switch swaps movement sets if more than four buttons exist. If fewer than four movements are supported, the last switch hides automatically.

Towards the bottom of the content area, the operator will encounter a maximum of four buttons specifically designed for operating the conveyor. In cases where only two buttons are available, they are strategically positioned near the screen's edge to enhance accessibility, particularly on hand-held devices. The background colour of these buttons serves a meaningful purpose and conveys the following information:

- **Light grey** - The button is not usable as the element is not in manual mode.
- **Dark grey** - The button can be operated.
- **Dark green** - Movement is currently in execution, and the drive is running.
- **Yellow** - The end position has been reached.

Data Tracking

The process of data tracking is a critical aspect of the conveying system. Mishandling or incorrect manipulation of data can lead to severe consequences, causing system failures and unwanted downtime. While data tracking is automated, there are instances where operators or commissioners need to intervene, either to correct tracking issues or conduct tests during system deployment. Hence, it is crucial to have a clear interface for performing these operations.

In the previous visualization system, the data tracking screen was challenging to read and interact with, often causing difficulties for operators to adapt to it. As a result, errors in data manipulation would occasionally occur.

In essence, data tracking involves maintaining a record of individual characteristics of the goods being conveyed, such as their unique identifier, weight, source, and destination. As goods flow through different types of conveyors, their data must accompany them throughout the process. Conveyors are assigned a specific count of places, denoting the goods they can hold concurrently. These place settings are configured individually for each conveyor. The INTRALOG standard accommodates conveyors with a range of one to six places. As depicted in Figure 6.45, each conveyor has a specific number of places, with the second conveyor having four places.

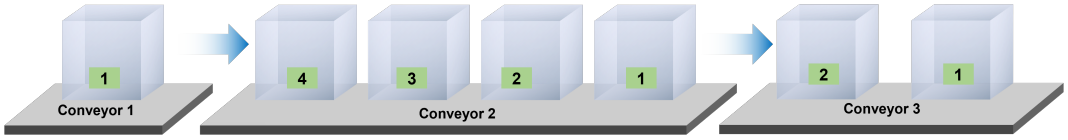


Figure 6.45: Conveyor data tracking places representation.

Given the significance of this data, a virtual "infeed place" is created for each conveyor. During transfer, data is initially copied to these infeed places. Once the actual target places are free of data, the information is transferred to their respective target places. This intermediate step helps minimise errors caused by potential transport overlaps. The tracking information of the places' data is automatically stored in the "DB_TRACKING" in the order of the elements used to construct the system.

Figure 6.46 showcases the primary interface for data tracking.

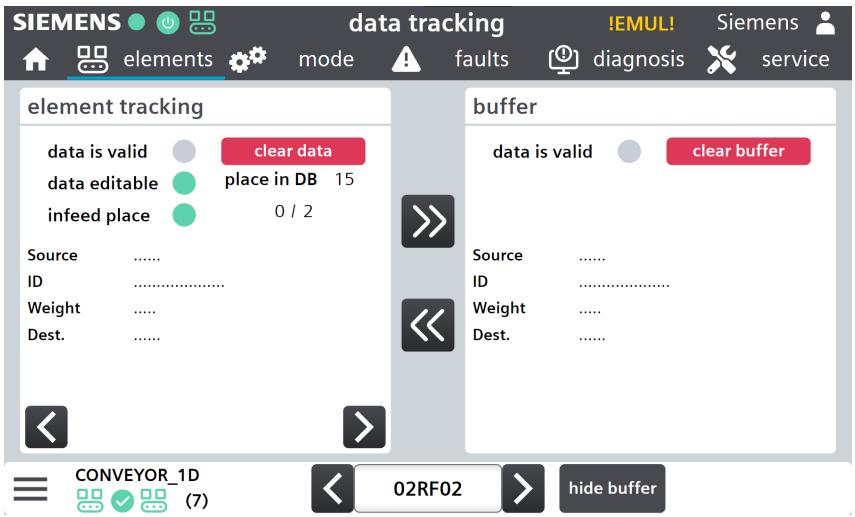


Figure 6.46: "Data Tracking" screen.

The layout consists of two distinct sections. On the left side, the selected element's data tracking is displayed, along with additional details in the upper section. Within this area, three circular indicators are present. The first two informs the

user about the current state of the data written in the designated place, while the other indicates whether the current place in the database is an “infeed place”. Furthermore, the upper portion provides information about the place number within the database and the total number of available places on the conveyor. For instance, “DB_TRACKING” Place 15, or Place 0 out of the 3 available on the conveyor, is an infeed place. The operator has the option to clear the data, but this action necessitates authorisation from a logged-in user. Below the tracking parameters, which can range up to six, two scroll buttons enable the user to navigate through the places. The second section of the interface is exclusively dedicated to the “buffer”. Whenever data requires editing or copying to another place, the user must transfer it to the buffer using the double arrow button positioned at the centre of the screen. Subsequently, using the keyboard, the user can manipulate the data and copy it to the desired destination. Considering that the buffer does not necessitate continuous access or usage by unauthorised users, the footer incorporates a button to toggle its visibility.

Since the data tracking of goods flows from conveyor to conveyor, having an interface that facilitates the evaluation of this exchange of data between one conveyor and its partners can be useful. For this reason, the “Partners Data Tracking” screen (Figure 6.47) was developed from scratch for the new visualization interface.

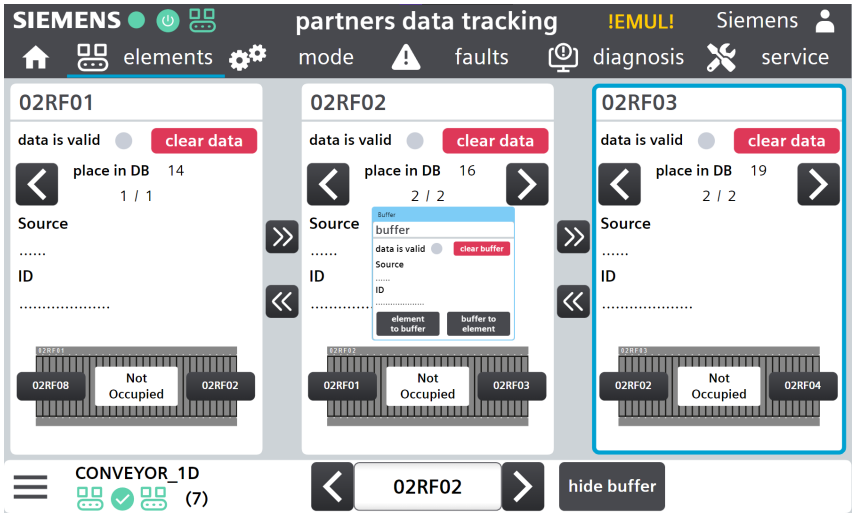


Figure 6.47: “Partners Data Tracking” screen.

The user will be generically provided with three content areas: one for the conveyor selected from the footer and another two dedicated to its rear and front partners. For each conveyor, a condensed version of the content from the main data tracking screen is provided. Below the indication of the validity of the data and a button to clear it, the numerical information about the place being evaluated is shown, accompanied by scroll buttons to scroll to those existing for each conveyor. Then, due to space concerns, only two parameters of data tracking for the selected

place are presented; for this screen, “Source” and “ID” were chosen. At the bottom, a graphical representation of the conveyor is given with an indication of the occupied state and buttons to allow a quick loading of the partners in the main content area.

In order to move data between the places of distinct conveyors, the double arrow buttons between the content areas can be used.

Furthermore, following the same approach as the main data tracking screen, access to the “buffer” is also provided, but since the space available on the screen is limited, it is provided in the form of a pop-up. In addition to the data tracking parameters, the “buffer” offers two buttons to allow data to be moved to and from it to the elements. In order to choose which element this flow of data refers to, the user can press over the content area of each element in order to select it, making its border turn blue to highlight the fact that the interaction with the buffer is being performed with that element.

When dealing with conveyors featuring multiple places, relying solely on the aforementioned screens for data tracking evaluation might not be the most effective approach. Furthermore, these screens don’t offer simultaneous access to the contents of all places. To address this limitation, a new screen called “Places Data Tracking” (Figure 6.48) was developed, allowing users to conveniently view the content of each place for the selected conveyor from the footer.

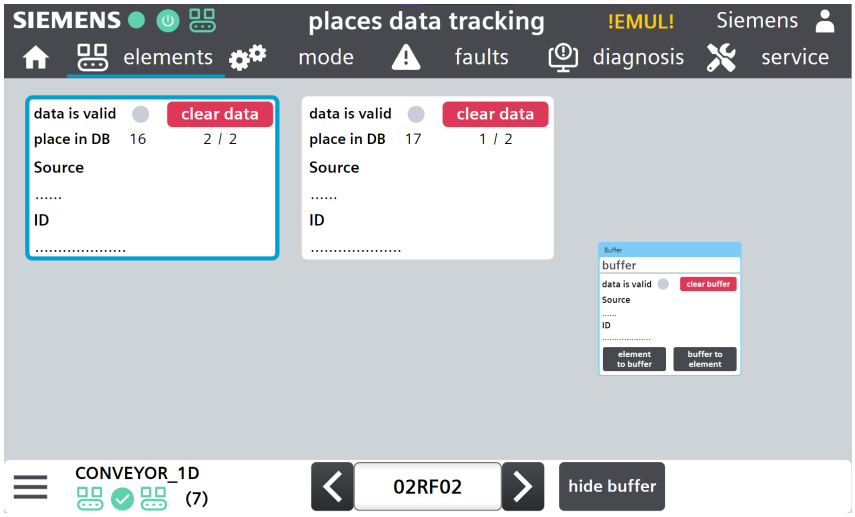


Figure 6.48: “Places Data Tracking” screen.

Considering that the standard supports conveyors with up to six places, the screen layout was optimised to accommodate these places, presenting them in the order of their indexes in the DB where places data is stored. Consequently, the infeed places are not displayed on this screen. For moving data between the places, users can utilise the “buffer”, which operates on the same principle as the “Partners Data Tracking” screen, as depicted in Figure 6.47.

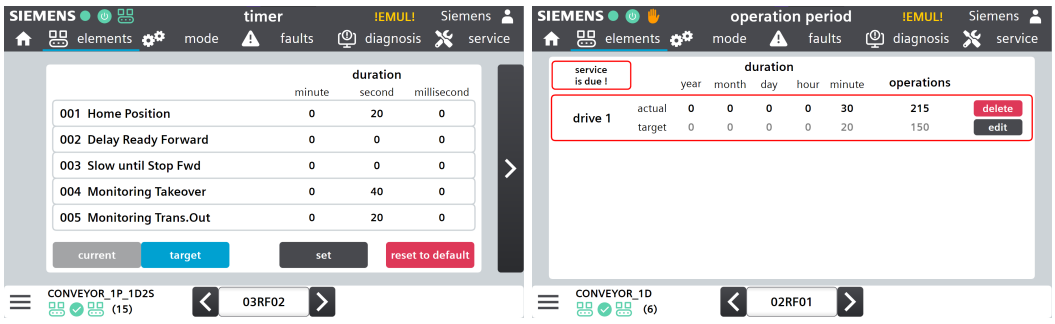
The data tracking screens showcase a modified version of the default footer found on the elements screens, displaying a button to toggle the buffer visibility. Triggered upon pressing it, this button on the last two presented screens allows the buffer's display through a faceplate, which opens as a pop-up. As seen in Listing A.21, an initial Unified inbuilt instruction is performed to determine if a pop-up has already been opened. This data is then used to inform the PLC about the state, allowing for appropriate processing of the buffer data. If the pop-up is not open, the `OpenFaceplateInPopup` instruction is executed to display the pop-up. The "WindowFlags" are used to control certain properties of the pop-up, enabling user interaction such as moving and resizing. Additionally, a border and header are shown to aid in identifying the pop-up on the screen.

Moreover, depending on the current state of the pop-up, the text of the button located in the footer is dynamically adjusted. This evaluation occurs during the loading event of the screens, and also when the button is pressed, ensuring the button accurately reflects the pop-up's visibility status.

A concise explanation of the operational principles behind the PLC code created to implement these new screens can be found in Appendix B.

Timer and Operation Period

Unlike the previously presented screens, the "Timer" and "Operation Period" (Figure 6.49) underwent minimal changes and did not receive any new features compared to the old visualization interface. The focus was primarily on redesigning the screens to improve readability and user interaction.



(a) (b)

Figure 6.49: (a) "Timer" screen (b) "Operation Period" screen.

The first screen allows users to set time values for specific operations related to the selected element. Since these parameters are crucial for the system's proper functioning, only authorised users can modify the target value or use the "set" and "reset to default" buttons. The screen provides both a "target" view for parameterisation and a "current" view to display the currently defined values.

As for the “Operation Period” screen, it provides information about the duration and number of operations performed by a drive. This data is displayed in the “duration” and “operations” columns, respectively. Additionally, users can set target values for these parameters to evaluate how long the drive can operate before requiring service. The screen allows retrieval of data for up to four drives associated with the selected element. If the prescribed running time or number of operations is exceeded, a “service is due !” message is shown, accompanied by a flashing red border surrounding the drive that needs maintenance. Each drive is accompanied by two buttons. The “delete” clears the current running time and number of operations for the drive. When the “edit” option is activated (indicated by a blue background), users can modify the target values. Otherwise, the values are displayed in an output format only. To ensure data integrity and prevent unauthorised access, both the “delete” and “edit” buttons require an authorised user to log in before performing any actions.

Special

The “Special” screens are automatically generated based on the parameters configured for each single element and are provided with data from the respective FB parameterised in the PLC. It serves to provide the user with relevant information regarding the interfacing of the single-element internal control capabilities and the overall control system for the conveying technology. As an example, if the element is a conveyor, it will most likely have a “Special” screen in order to view and set the parameters of the controlling drive based on the FB used for its configuration.

Since there are multiple types of single elements used on the conveyor system, and creating a specific “Special” screen for each would be a tedious process, depending on the type of element selected, the PLC automatically generates multiple strings holding labels or data, as well as boolean and numerical control parameters targeted for constructing the screens on the visualization system.

In order to leverage these PLC tags, an unique faceplate (Figure 6.50) is used to generate all types of “Special” screens. The faceplate contemplates all available PLC tags, targeted for the “special” content, that are linked to multiple text boxes, I/O fields and buttons that are strategically positioned on the screen in order to construct a coherent interface. Combining these tags and the dynamization options provided by Unified, it’s possible to control multiple properties of the screen components, like their visibility, size, and weight of the text font, in order to achieve the desired screen layout.

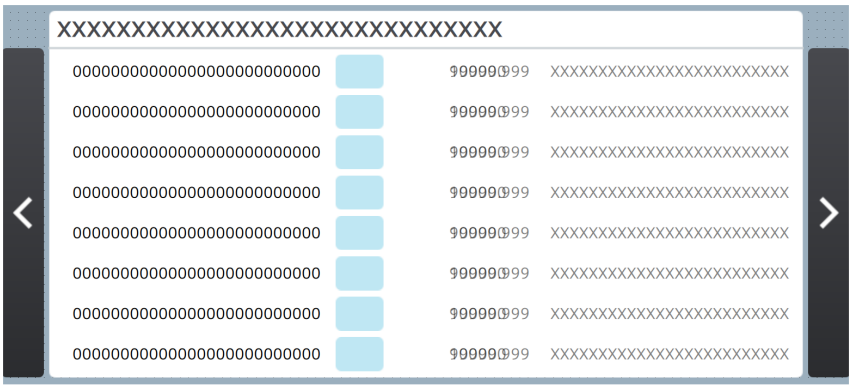
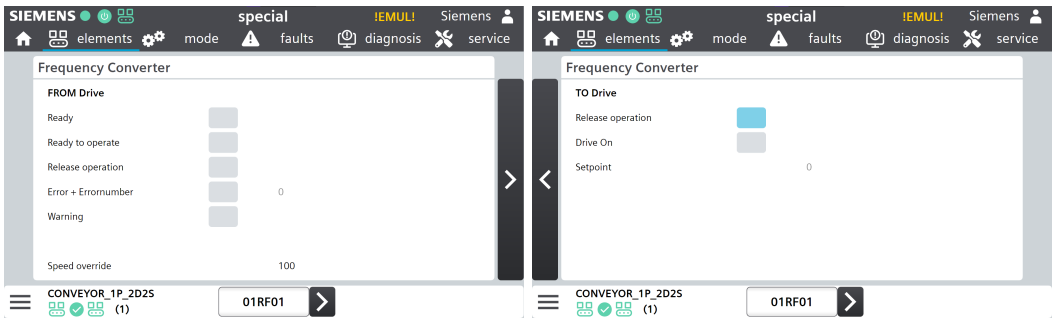


Figure 6.50: “Special” screen faceplate.

As an example of what is presented to the operator during runtime, based on the faceplate previously described, Figure 6.51 shows the “Special” screen called “Frequency Converter” that allows an interface to the drive controlling the single element “01RF01”. Using the scroll buttons, the user has access to an interface for the parameters sent from the drive and the parameters that are sent from the PLC to the drive.



(a) (b)

Figure 6.51: “Special” screen conveyor drive: (a) From Drive (b) To Drive.

Multicontrol

The “Multicontrol” screen (Figure 6.52) enables users to operate multiple single elements concurrently.

Each page accommodates up to eight content areas assigned to single elements that the current CP is authorised to manage. For each element, up to eight distinct movements are provided and can be selected.

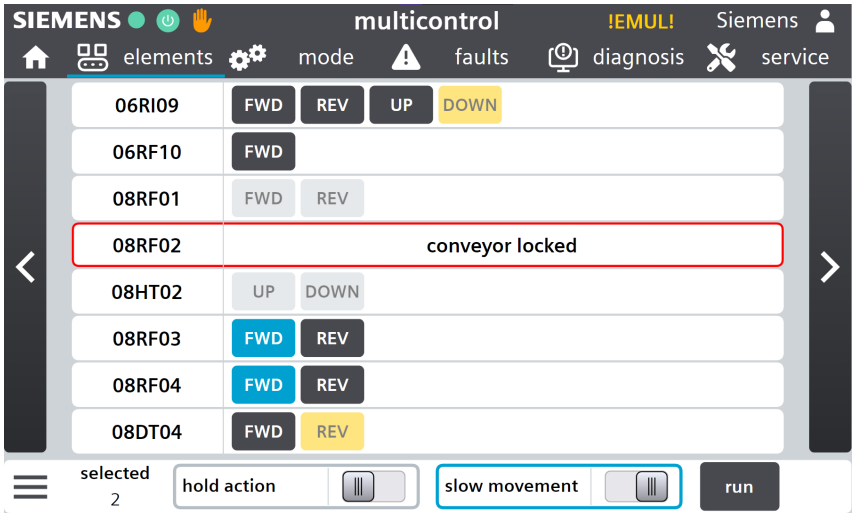


Figure 6.52: "Multicontrol" screen.

To choose the available movements for each element, it must be operating in manual mode, either individually or as part of a group in manual mode. If the element is not in manual mode, the movement buttons are deactivated and displayed in a light grey colour, indicating that they cannot be pressed. Conversely, when the element is in manual mode, the buttons appear in dark grey, indicating that they are selectable, and the corresponding movements can be executed.

To initiate the movements, the user can select the desired ones for each element, causing the buttons to turn blue. Then, by pressing the "run" button located in the bottom right corner, all the selected movements are triggered simultaneously. While pressed, the "run" button assumes a blue background. If certain movements cannot be performed due to reaching the end position, the corresponding button turns yellow and becomes unresponsive.

In the case where an element is already under the control of another operator in a different HMI, a "conveyor locked" message is displayed over the buttons for that particular element. This safety measure prevents two users from operating the same element simultaneously.

Additionally, at the screen's footer, a counter keeps track of the number of selected buttons, helping the user monitor their instructions. Apart from the "run" button, the operator has the option to use the "hold action" feature, allowing the selected movements to execute continuously without the need to keep pressing the "run" button. Furthermore, the operator can choose the "slow movement" option, which executes the operations at a reduced velocity for enhanced controllability and safety.

Manual Operation

The "Manual Operation" screen (Figure 6.53) empowers the user to control multiple elements simultaneously in manual mode. This screen shares similar content with the "Detail Overview" screen (Figure 6.42) but for two distinct single elements, enabling multiple operations concurrently. It is particularly beneficial for scenarios like transferring pallets between two conveyors.

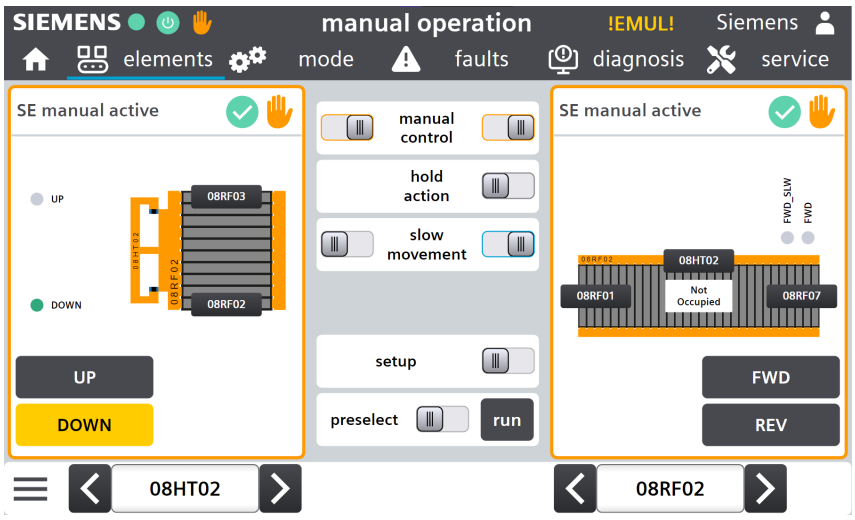


Figure 6.53: "Manual Operation" screen.

Compared to the old interface, this screen has undergone various upgrades, enhancing the user experience and functional capabilities. Like other screens under the "elements" category, it provides centralised access to crucial components of conveyor operation, including errors, input and output states, and more, through a "detail" representation of each element.

Within the graphical representation of the conveyor over the SVG, a set of buttons is strategically positioned and labelled with the partners of the conveyor. These buttons serve not only as output information but also enable quick loading of a partner into the opposite content area by pressing the corresponding label. This eliminates the need for pressing the scroll buttons multiple times, or accessing the plant overview through swipe-up gestures, in order to load a partner into the content area. As an example, on the representation of the "08HT02" element, the "08RF02" can be recognised as the bottom partner. By pressing its label, this element is automatically loaded into the content area positioned on the right.

According to the type of element and the properties set for it on the PLC program, the "hold action", "slow movement" and "layer" are accessible or visible when the element is in manual mode.

As this screen accommodates two element inputs, it does not make use of the default footer for the "elements" category. To retain essential features, such as the

operational state of the single element and the fault indicator/acknowledge button, they were relocated to the header of their respective content areas. Additionally, the swipe-up gesture to access the elements list has been individualised for each element, making their selection independent.

This screen offers two additional noteworthy features for the operator's convenience (Figure 6.54). The first feature, known as "setup", proves valuable for commissioning and maintenance tasks. When activated, it places the selected elements' groups into construction mode, resulting in all movement instructions executing at a slower pace. Moreover, in the "setup" mode, if an element is experiencing a fault, the execution of movements is still authorised.

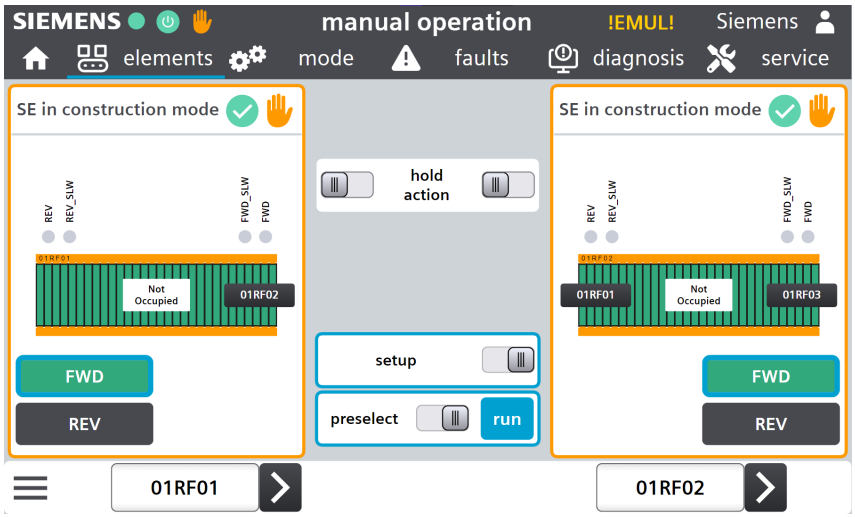


Figure 6.54: "Manual Operation" screen, "setup" mode and "preselect" functionality.

The second feature, "preselect", allows the operator to choose a movement from each element. Once selected, a blue border highlights the chosen movements, and by pressing the "run" button, both instructions are executed simultaneously.

6.2.5 Operation Mode

In the following Sub-subsections, improvements and new developments to the screens used to manipulate the operational state of the groups of elements are presented.

Overview

The "Operation Mode" screen (Figure 6.55) provides status information for each group that can be controlled by the HMI based on the connected CP. Each group is represented by a content area displaying its name and number, allowing evaluation of its operational status and enabling control instructions. The border surrounding

each content area is programmed to display a colour corresponding to the group's operational mode.

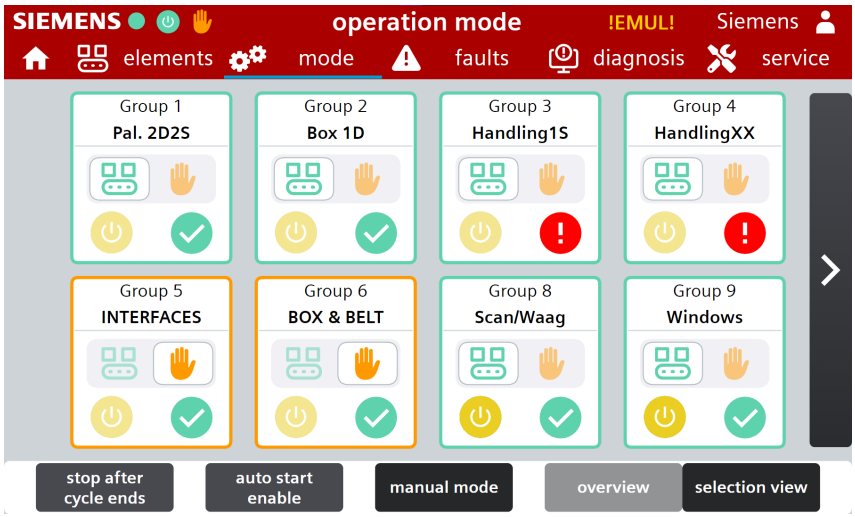


Figure 6.55: "Operation mode overview" screen.

For each group, the screen shows the automatic or manual operation status which can be controlled using segmented buttons, and a yellow pause button in order to request that the group to stop. Additionally, a graphic view indicates if the group has a single element with an active fault. Pressing it allows acknowledgement of the fault, if possible.

Compared to the equivalent screen on the previous HMI solution, the new solution has undergone a redesign to enhance its visual appeal. Additionally, it has been modified to accommodate a larger number of groups simultaneously, enabling faster interaction with the process. The new screen now provides an overview of eight groups, compared to the previous visualization, which only displayed four.

With the new scrolling feature, the groups displayed on the screen are dynamically updated based on the remaining groups to be shown. For example, if there are four additional groups and the scroll button is pressed, they will appear in the second row of content areas. However, if there are eight additional groups available for control and the button is pressed, all eight will be displayed simultaneously. In contrast, the old solution would only update the last content area when the scroll button was pressed, resulting in a slow process of navigating through the groups.

These improvements were implemented by making changes to the PLC program, specifically by introducing a new PLC constant called "C_OPERATING_TYPE_-GROUP_UNIFIED" with a value of 8. Additionally, the condition associated with the scroll button press was adjusted according to the modifications outlined in Listing B.1, available in Appendix B.

In terms of the footer functionalities, the first button allows the operator to request the PLC to stop the conveying process for all groups at the end of the

current cycle. When this option is active, the button's background changes to the same yellow colour as the pause buttons of the groups. The "auto start enable" button serves as a two-step safety measure to transition a group into automatic mode. When a group is in manual mode and the operator intends to switch it to automatic, they need to press the corresponding group's switch. Simultaneously, they will notice that the border of the group's content area and the "auto start enable" button flash together. This visual feedback helps the operator understand that to transition the group to automatic mode, they need to press the enable button, preventing any accidental instructions. The other two buttons in the footer have distinct background colours and are meant to navigate to different screens.

Selection View

The "Operation Mode Selection" screen (Figure 6.56) is a new feature added to the new visualization system. This screen's goal is to provide users with a "selection view" where they can apply several actions to the groups more quickly when compared to the overview screen. In the selection view, the user will observe that the groups controllable from the CP are listed on individual content areas, similar to the overview screen. Due to screen space constraints, just the names of the groups are presented in this form. When the user presses on a group, the selection status, indicated by a check mark, is toggled. By performing a customised selection of the groups, either manually or using the "select all" or "deselect all" buttons, the operator will be able to perform operations to multiple groups at once, without having to do it one by one. These operations can be found on the bar available on the right side of the screen, and are similar to the ones provided on the overview screen.

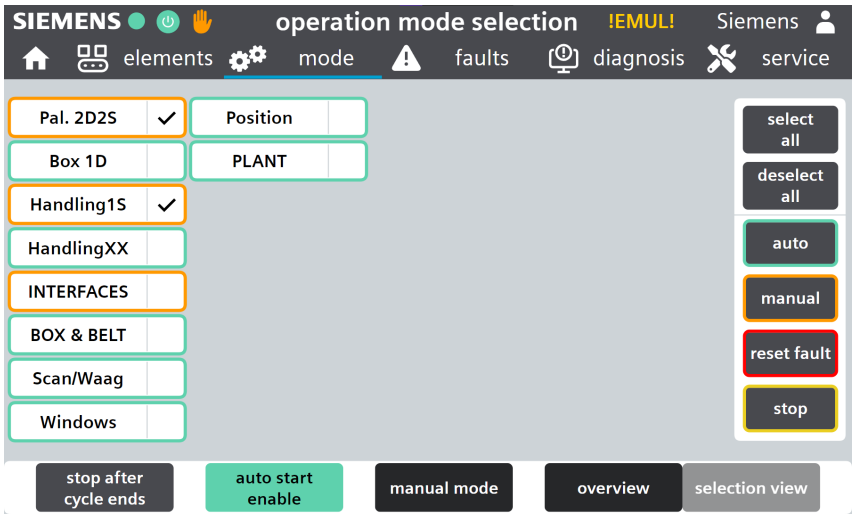


Figure 6.56: "Operation mode selection view" screen.

Appendix B contains a comprehensive explanation of the operation of the PLC code, developed to support the functionalities provided by the aforementioned screen.

6.2.6 Faults

The "Faults" screen (Figure 6.57) incorporates an "Alarm Control" component which grants users access to the program alarm feature of the PLC.

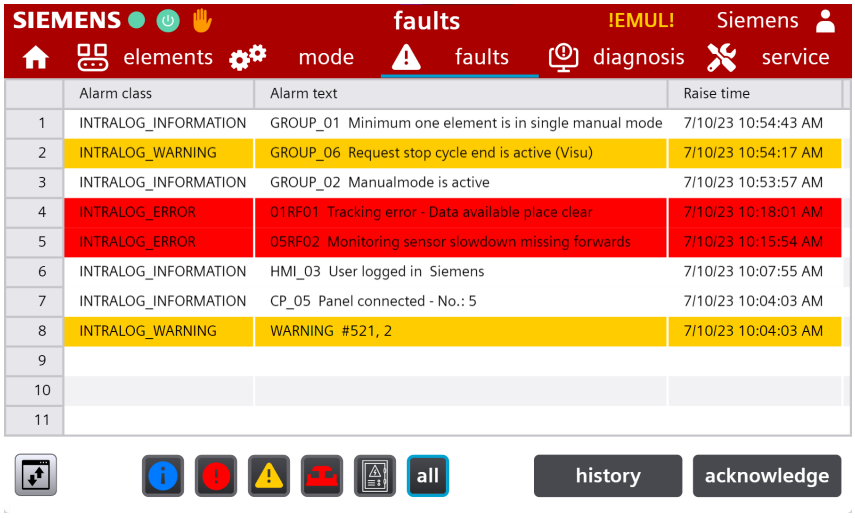


Figure 6.57: "Faults" screen.

Through this screen, users can evaluate pending alarms specifically from the "INTRALOG_" alarm classes. The properties of the alarm control have been customised to display only the most pertinent information for common users. These include the alarm type, determined by its alarm class; the alarm text, which provides a message describing the content of the alarm; and the raise time, which records the moment the alarm became active. To facilitate evaluation of the alarm status, each row in the control is assigned a suitable background colour corresponding to its alarm class. Furthermore, aside from system errors, this screen also allows extraction of more generic information about the state of groups and the system. For example, it can indicate when a group has an element operating in manual mode or when an HMI panel is connected to the system.

As mentioned, the alarms originate from the S7 PLC program alarm feature. In Figure 6.58, the definition of these alarm messages can be observed within the "PLC supervisions & alarms" section of TIA Portal. These messages are parameterised based on the FB that originate the single elements, as indicated in the "Location" column. The "Alarm text" is then constructed using data from these FB. For instance, the "Addition text 2" field will contain the name of the specific element when a fault is associated with it. Additionally, an "Alarm class" is configured for each type of alarm.

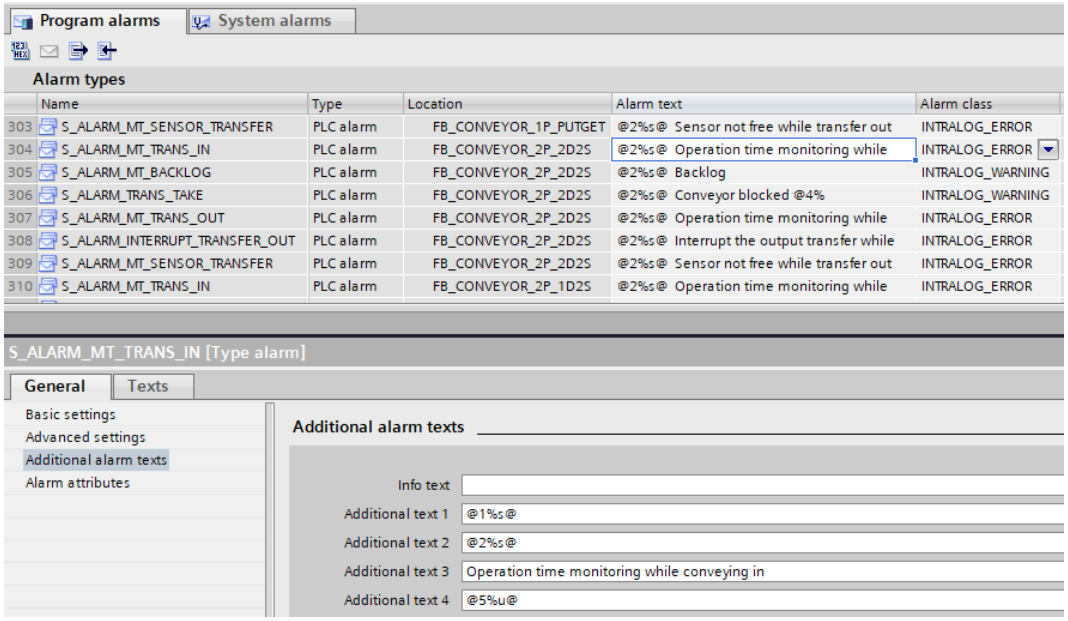


Figure 6.58: Program alarm text source.

The configuration of "Alarm classes" is expanded to the Unified project as depicted in Figure 6.59. This configuration allows for the management of colour schemes associated with each alarm class. Moreover, it provides the option to generate an "Alarm log", referred to as "Alarm log_1", which serves to store alarms and offer a history function to the users.

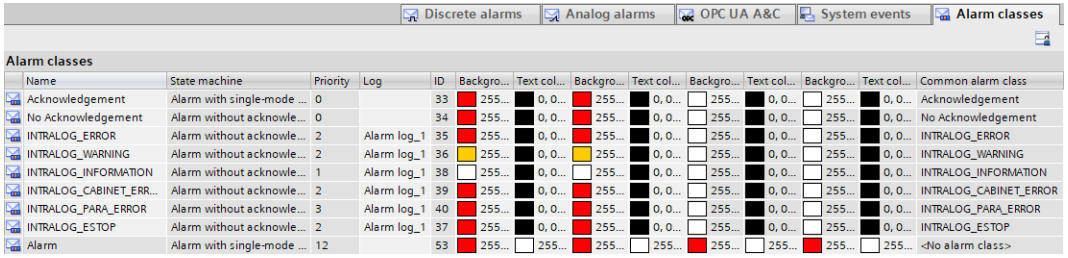


Figure 6.59: Alarm classes.

Upon pressing the "history" button, the "Alarm source" property of the alarm control is toggled from "Pending alarms" to "Logged alarms updated." This change enables the retrieval of alarms from the "Alarm log_1" file.

Alongside the "history" button, the lower section of the screen features six buttons that enable the user to filter alarms based on their alarm class. When one of these buttons is pressed, a script (presented in Listing A.22, available in Appendix A) is triggered, modifying the alarm filter accordingly. Moreover, the button's border is altered to provide visual emphasis, highlighting the currently active filter. In the left corner, there is a button that opens a pop-up window offering various sorting

options for the alarms. This feature enables users to sort the alarms based on criteria such as the raise time. Lastly, the "acknowledge" button serves the purpose of requesting the PLC to acknowledge the alarms once the errors have been identified and resolved.

The "Alarm control" was configured to allow users to scroll freely through the list of alarms. This feature, combined with the utilisation of the "Selection changed" event, allows for the execution of a script when a user selects a specific row from the alarms list. This functionality opens up possibilities such as retrieving information about the single element associated with the selected fault and automatically navigating to the corresponding process screen where the faulty element is already preloaded. As illustrated in Listing A.23, available in Appendix A, when the "Selection changed" event is detected, a script is triggered. The function associated with this event receives the "SelectedRowData" structure as a parameter, enabling access to the event text. By leveraging this text and recognising that the initial portion of the message indicates the faulty element, its label can be extracted. Subsequently, a request can be made to the PLC to switch to that specific element, while simultaneously updating the process screen to the last accessed elements screen.

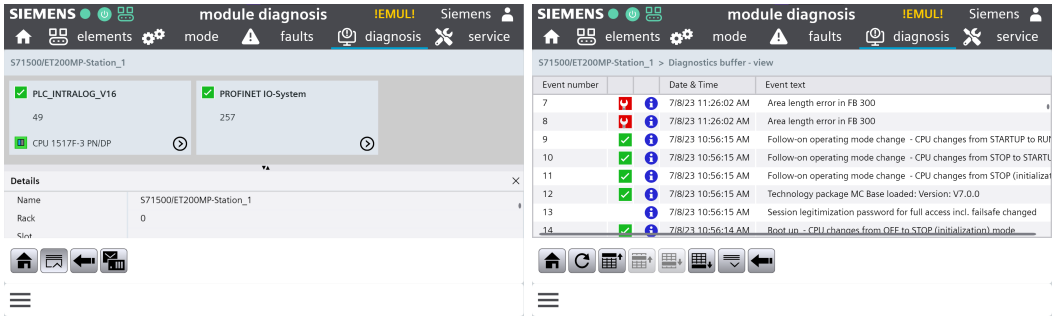
6.2.7 Diagnosis

The following Sub-subsections of this document describe the development of the screens under the "diagnosis" category.

Module Diagnosis

The "Module Diagnosis" screen incorporates a "System diagnostics control" screen component from the Unified toolbox. It is utilised with its default settings, requiring no modifications other than adjusting its size properties to occupy most of the process screen. This tool enables quick diagnostics of the components within the PLC network using a matrix-style navigation system. It not only detects and evaluates faults but also allows the retrieval of hardware installation parameters. Figure 6.60a illustrates that by clicking on the second button on the built-in toolbar, a "Details" window opens, providing information such as the component's name, IP address, error descriptions, and more.

Additionally, the fourth button on the toolbar allows the user to access the diagnostics buffer view (Figure 6.60b) corresponding to the selected component in the matrix view. This view presents a list of various events, including errors in the processing of the PLC code or general notes related to the operation mode of the system. By selecting a specific line and clicking the "Details" button, additional information is provided. In the case of an error-related event, a concise description outlining the steps to resolve the issue is provided.



(a) (b)

Figure 6.60: Module diagnosis: (a) Matrix view (b) Buffer event list.

Reporting

Data exchange between the PLC and the WMS occurs through the transmission and reception of telegrams. These telegrams contain an encapsulated message that follows a predefined format and includes a timestamp. Typically, these telegrams are exchanged to track the movement of goods within the system. For instance, when a pallet changes its location, a "Location Left" telegram is sent to the WMS. Additionally, the PLC can utilise telegrams received from the WMS to determine the appropriate destination for a particular pallet, enabling it to control the conveyor technology accordingly to achieve the desired target.

To facilitate the monitoring of data exchange between the systems, specific process screens were developed as part of the visualization. The "Reporting Overview" screen (Figure 6.61) provides a dedicated content area to access reports. Within this area, all available reports are listed, and selecting a report will transition the interface to a dedicated screen displaying the detailed content of that particular report point.

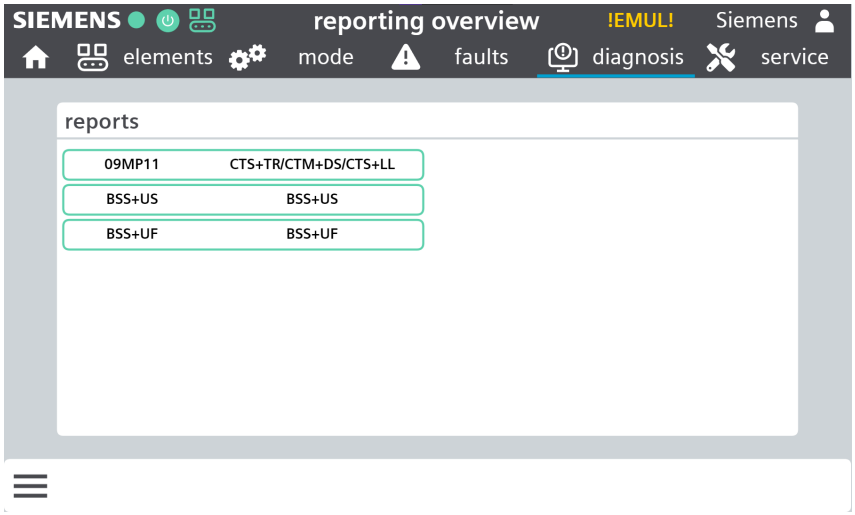


Figure 6.61: WMS "Reporting Overview" screen.

The “Reporting” screen (Figure 6.62) consists of four main content areas, scroll buttons, and a footer with additional functionalities. In the first content area, the number corresponds to the position of the report on the reporting overview list, while the label and function help identify the telegram and its purpose. The middle content areas display the sent or received telegram along with its respective timestamp. The bottom content area allows the operator to determine the element that generated the report point. Additionally, a circular indicator notifies the operator when a response from the WMS is awaited, in reference to the telegram sent by the PLC.

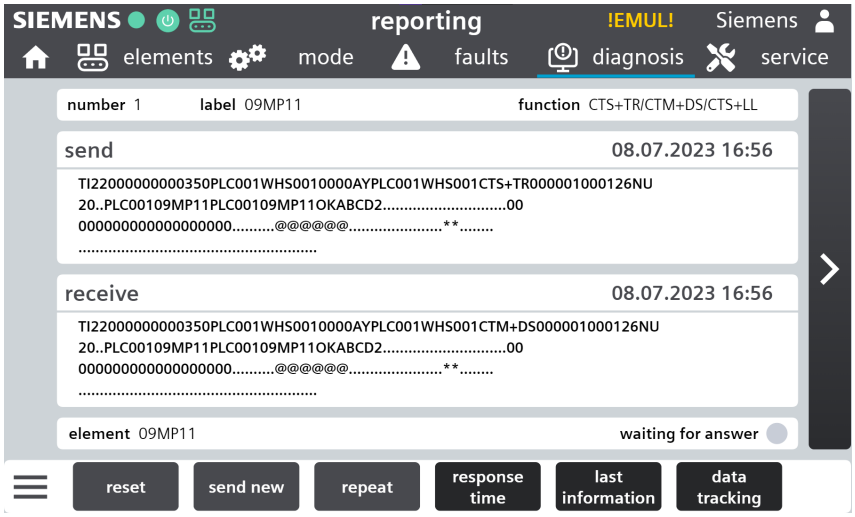


Figure 6.62: WMS report example.

Finally, the footer presents a series of buttons available to the operator. The first three buttons serve as tools to manage the telegrams sent from the PLC. The first button is self-explanatory, as it allows for resetting the telegram. Choosing the “send new” option enables the PLC to send the same telegram with a different timestamp, while selecting “repeat” resends the telegram with the original timestamp.

The last three buttons are dedicated to navigation functionalities, distinguishable by their darker background tone. The first two buttons transition the process screen to complementary tools to monitor the communication process and its performance. Lastly, the “data tracking” button switches the process screen to the “Data Tracking” (Subsection 6.2.4), where the single element under analysis in the report is loaded. Given the close relationship between reporting and data tracking, this button is essential for troubleshooting and effectively monitoring the conveying process.

6.2.8 Service

This Subsection aims to explore the development made for the screens related to the “service” category.

Group Operation Control

The "Group Operation Control" screen (Figure 6.63) serves the purpose of managing access and control over groups from each HMI instance, particularly when control over groups of elements is shared among multiple CP.

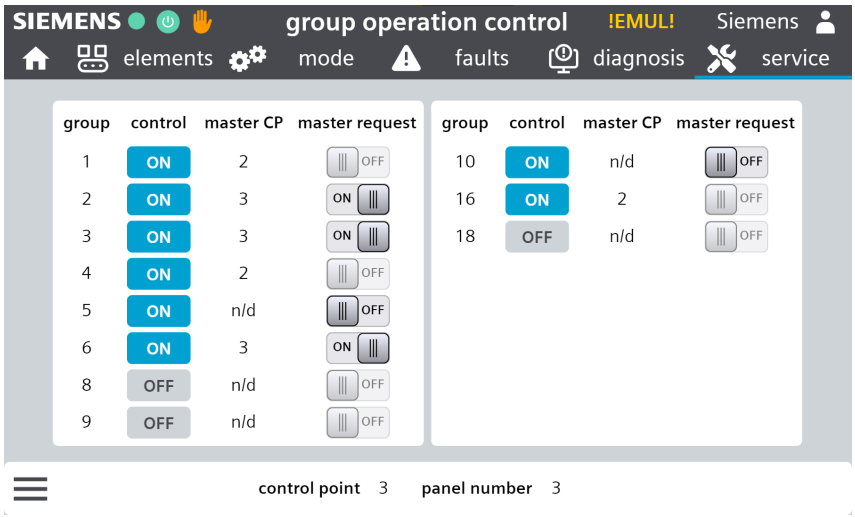


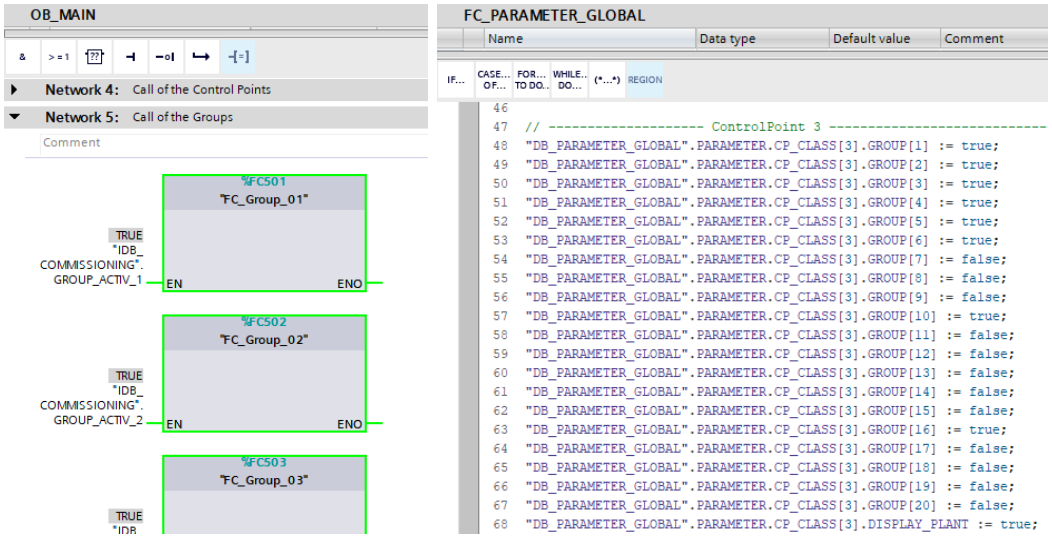
Figure 6.63: "Group operation control" screen.

The "master CP" column indicates which CP is currently responsible for managing each group. If a CP wishes to control a group but the master CP is assigned to another, a request to control the group is required. This is a crucial safety measure. For instance, it can be observed in Figure 6.63 that Group 1 is being controlled by CP 2. Without access management to the groups, a user on CP 3 could potentially switch Group 1 to automatic mode, posing a risk to anyone involved in manual operations on that group. Therefore, if the HMI on CP 3 wants to control Group 1, a pop-up notification is sent to the HMI on the master CP (2) to grant or deny privileges and authorise the operation.

If a group doesn't have a designated master CP and is represented as "n/d," it means that the first CP to make any changes to the group state will automatically become the master. For example, if Group 5 has no master CP assigned and CP 3 switches it to manual mode, CP 3 will be automatically assigned as the master CP.

Additionally, from this screen, it is possible to initiate a "master request" via a switch for groups that have an undefined master CP state. Furthermore, if the user decides to relinquish their master CP privileges at any point, the master request switch can be toggled to the OFF position.

In this screen, all active groups in the "OB_MAIN" (Figure 6.64a) will be listed. However, only the groups with an "ON" state under the control column can be controlled from the HMI instance that is connected to the CP, based on the configuration set in the "FC_PARAMETER_GLOBAL" function (Figure 6.64b).



(a) (b)

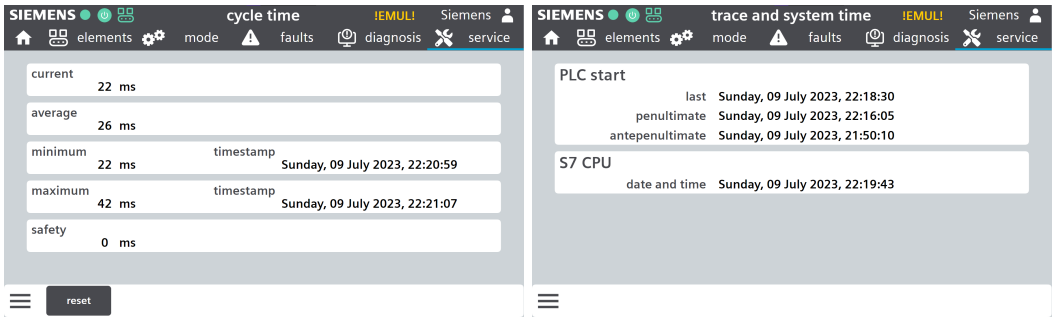
Figure 6.64: Groups call and parameterisation: (a) "OB_MAIN" (b) "FC_PARAMETER_GLOBAL".

System Performance and Information

As additional information, the service category includes several screens designed to provide users with insights into the performance of the system and the product they are utilising.

The "Cycle Time" screen (Figure 6.65a) displays various metrics related to the performance of the PLC. Alongside the current cycle time, the system calculates and records the average, as well as the maximum and minimum cycle time values with their corresponding timestamps. Furthermore, the safety system cycle time is also displayed.

In reference to the "Trace and System Time" screen (Figure 6.65b), it allows users to access the record of the three most recent PLC starts, in addition to displaying the configured date and time.



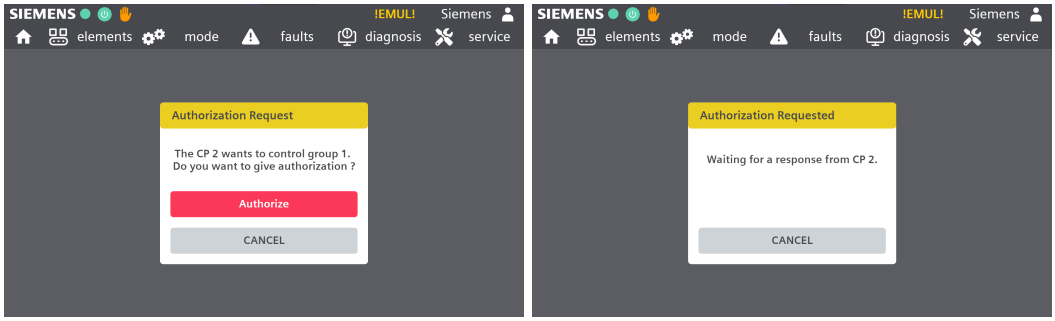
(a) (b)

Figure 6.65: PLC performance and information: (a) "Cycle Time" (b) "Trace and System Time".

6.2.9 Administration of Group Control Access

As was described previously, when a group is controllable from multiple CP, a mechanism is needed to prevent that two users perform control operations over the same group. Furthermore, it is important to have a way to request or give authorisation to another operator to work on a certain group when needed and suitable. In order to achieve this, there are two screens, which essentially display a message in the format of a pop-up, that allow this process to be handled.

Figure 6.66a displays the message that will appear on the HMI that holds the title of being the master CP of a group, when another CP requests the control authority for it. Additionally, Figure 6.66b displays the message that the user who requested access will see while awaiting a response regarding their control request.



(a) (b)

Figure 6.66: Administration of group control access: (a) View (b) Filter.

Due to the significance of these messages, the process screen is blocked until a response is received or the request is cancelled. After a predefined period specified in the PLC code, the user is returned to the screen they were working on. If the time limit is exceeded without obtaining a response from the master CP, control privileges are transferred to the HMI that initiated the control request.

The construction of the notification messages is achieved through a small script, as exemplified in Listing A.24, available in Appendix A, for the “Authorisation Request” pop-up.

6.3 Testing Process and Validations

To ensure the functionality of the operating programs, it is necessary to conduct appropriate testing. As previously explained, the sample project used for this purpose is INTRALOG_V16_0_13_MUSTER. This project encompasses the configuration of control and visualization for multiple groups of conveyor systems with varying structures and components. It serves as a foundation for customers projects, while

also playing a crucial role in testing, debugging, and training new employees who will be working directly with the standard.

The testing process and validations can be divided into two parts: control technology and visualization specific aspects.

Control technology focuses on the mechanisms for testing and validating the developed operating programs within the development environment. Through these mechanisms, the correct functioning of process-related functionalities, such as code developed for the PLC, Unified scripts and screen parameters interactions, can be tested using the INTRALOG standard built-in emulation mode.

On the other hand, the application-specific part of the testing process involves evaluating the operational interface flow, particularly the utilisation of the HMI by an operator. In this regard, the emphasis is placed less on testing the control functionality of the visualization solution and more on assessing user-friendliness, intuitiveness, plausibility of operation, and overall user experience.

To attain optimal performance and leverage the latest features, both the TIA Portal and WinCC Unified were utilised in their most recent version, specifically V18, with all the latest updates and service releases installed. The same approach was also adopted for the panel image employed on the UCP.

6.3.1 Control

The sample project includes an integrated emulation mode that allows running the control software without the need for physical hardware connections. In this mode, components such as sensors and drives can be emulated, enabling testing and development of the software. After activating the emulation functionality, from the corresponding DB all relevant data can be used and manipulated.

Additionally, the TIA Portal provides tools that allow simulating the controller and the HMI panel without physical hardware. During development, the UCP simulation was accessed through a Google Chrome browser on a PC running Microsoft Windows 10. Furthermore, the monitoring function in STEP 7 can be utilised during runtime to display current values of the used data areas. This allows verification of the correct functioning of the program structures and facilitates identifying and rectifying errors.

In addition to simulating the operator panel, the development and testing process involved utilising physical hardware available in the Siemens Experience Centre, such as a S7-1500 CPU, a MTP700 UCP, a Connection Box Advanced and an IWP10F Mobile WebClient. The use of real hardware provided valuable insights during the development phase. By testing the interface and overall operation on the physical panels, the performance of the solution could be evaluated and validated. Since there are potential differences in functionality between PC systems and UCP, testing on real hardware is a must [89].

To validate the group control access of the HMI based on the connected CP, in the PLC project two CP for Mobile WebClients were instantiated and designated with IDs 16 and 240. Due to the availability of only one Connection Box for testing, initially, the rotary switches were set to 0x10, and the IWP10F was connected, resulting in the establishment of a connection to the CP with ID 16. Subsequently, the IWP10F was disconnected, and the rotary switches of the Connection Box were adjusted to 0xF0. When the Mobile WebClient was reconnected, it was evident that the HMI CP now had the ID 240, enabling control over a distinct set of groups. This process ensured the proper functioning of CP detection, guaranteeing that the operator would only have access to the elements physically visible to them. In doing so, this crucial safety feature was successfully addressed, thereby overcoming one of the main challenges during the transition to Unified.

Using Mobile WebClients to access the visualization interface offers the advantage of establishing connections at multiple distinct points within an industrial facility. However, given that these facilities often encompass vast areas with complex conveyor systems, multiple PLC are required to control the entire system. Consequently, several PLC and Unified Servers are set up with different IP address ranges to which the IWP10F must connect. To avoid the cumbersome task of manually configuring the IWP10F IP address to match the PLC range every time the user switches connections, the capabilities of the PROFINET protocol were harnessed. By utilising the name of a device existing on a PROFINET network, the system can automatically assign an appropriate IP address to the device. To validate this feature, two PLC projects were created, each with distinct IP address ranges. In each project, an IWP10F panel was integrated into the network with the PROFINET name “iwp10f”. Subsequently, on the IWP10F panel network settings, the PROFINET option for IP address assignment was selected, and the panel name was also set to “iwp10f”. After completing the parametrization, the PLC projects were individually downloaded, and the panel was connected to each one. As a result, it was observed that the IWP10F IP address was automatically assigned based on the range of each PLC, thereby successfully validating this feature.

6.3.2 Visualization Interface

For debugging the visualization project, different options are available depending on the test case. A simple method involves using I/O fields in the visualization to quickly display and modify specific variables. However, the most powerful and comprehensive debugging method in WinCC Unified is the RTILtraceViewer, being particularly useful for testing scripts, which are often prone to errors during development.

The RTILtraceViewer, an external program bundled with WinCC Unified, allows viewing JavaScript trace logs, which record all messages during runtime, including

script execution errors. To generate and display custom trace messages in WinCC Unified, the script function `HMIRuntime.Trace("User Defined Trace Message")` can be used at any point in the operating programs. The desired message is passed to the function as a character string and is handy for outputting and monitoring variable values or adding execution notes during debugging. This tool provides a way to track the overall execution of the script or function sequence.

Throughout the development process, the project underwent testing by various individuals who provided feedback on aspects such as screen object sizes, navigation flow, and readability. This feedback was carefully incorporated into the implementation to improve the overall design and enhance usability. Additionally, regular meetings were held with the standard development team and commissioning personnel who have extensive expertise in the standard. These meetings served the purpose of verifying the feasibility of the design and implemented functionalities, while also gathering feedback to incorporate new features and enhancements.

Chapter 7

Conclusions

One of the main objectives of this thesis was to accomplish the transition of the visualization interface of the INTRALOG CS standard from WinCC Advanced to Unified, performing not only a complete visual redesign but also its integration with new hardware from the Unified catalogue. Since this new visualization solution follows a server-client architecture and is based on web technologies, the goal of providing a completely wireless system for the interface is a milestone for the INTRALOG standard. This led to another objective of this project: the study of technologies that allow a complete wireless implementation of the visualization system.

The evaluation of the viability of transitioning to wireless technology revealed that, despite the potential benefits, there are significant challenges to be overcome, such as increased project complexity and higher hardware and licensing costs. In this context, opting to maintain the wired installation is considered a more reliable and economically advantageous choice for the customers, as well as being easier to maintain compared to implementing wireless systems. Furthermore, the transition to Unified is already a huge step for the INTRALOG standard. In order to provide a solid system, it's important to have time to test intensively and collect feedback from future customers about Unified before taking the next big step to transition to wireless technologies. Moreover, keeping the wired installation will allow a smoother transition for commissioners and development personnel.

Regarding the new visualization interface concept, it showed notable improvements, such as a modern and clean design and a reduced number of presses needed

to get to the desired process screen. This was possible by making the menu bar continuously accessible, storing the last accessed screen for each menu category, and introducing a side navigation that replaced the old overview screens and "return" button. Another major improvement was the introduction of the "Detail" faceplate on almost all screens in the "elements" category, making them more comprehensive and reducing the need for frequent screen changes due to the access to condensed information. The introduction of new screens, especially for data tracking, should be appreciated by commissioners and even future operators since the old interface is hard to read and has a prone-to-error design.

In addition to the overall design improvements and valuable new features, it's important to consider the system's performance. Even though, at this point, the use of PC as clients is not something that should be offered to customers, the system was tested on this platform to evaluate future development and test the visualization system hardware abstraction, since it was one of the objectives of this thesis. It is concluded that the performance on PC systems is excellent, as instructions like the loading of screens occur in around 1 second. About the UCP, it presents a performance that is inferior to the previous. Though the loading events occur at a pace that makes the user interaction very fluid, it was observed that some screens take around 2 seconds to load. In contrast, the IWP10F Mobile WebClient showcased mediocre performance, though very usable. The loading events of screens like "manual operation", which makes use of multiple SVG and nested faceplates, presented loading times of approximately 3 seconds, which are slightly higher when compared to the WinCC Advanced Mobile Panels. It is important to note that, even though the previous system appeared to be faster, it was also much more basic in its functionality. The performance of the Unified visualization interface is intimately dependent on the hardware on which it is running, and it is expected that future updates of Unified and panel images will improve this issue, considering that the system is still in its early versions.

Overall, WinCC Unified demonstrated satisfactory performance and the potential to provide powerful visualization interfaces, especially due to the capabilities provided by the use of JavaScript and dynamic SVG. Since this project has accompanied the release of several versions of the Unified system, it is clear that it is continuously evolving and that the degree of maturity is improving.

As final considerations about the project, it met all proposed objectives and was well-received by the entire Siemens team, suggesting that the next version of the INTRALOG CS standard, expected to be launched around the beginning of next year, will very likely make use of the research, software, and hardware development brought by this project.

7.1 Future Work

After completing the work associated with this thesis, several ideas emerge for future endeavours. Notably, achieving a fully wireless concept for the visualization interface of the INTRALOG CS stands as a significant milestone. Consequently, in-depth research pertaining to this aspect is imperative, with hardware tests being essential to establish in-house results concerning the feasibility and, particularly, the reliability of this approach.

Furthermore, it is crucial to stay abreast of upcoming updates and service releases of WinCC Unified. This is not only to ensure the interface remains up-to-date with the latest functionalities but also to enhance screen loading performance, especially on lower-end hardware like the IWP10F WebClient.

Anticipating the integration of this project into a new version of the INTRALOG CS standard, and its subsequent deployment in customer projects, it is foreseeable that minor adjustments may be necessary. While the project has already undergone extensive testing, the tests were conducted on a smaller scale and in a controlled environment. Hence, active participation in this integration process becomes essential to elevate the project to a more robust and mature state.

References

[1] H. Winkler and L. Zinsmeister, "Trends in digitalization of intralogistics and the critical success factors of its implementation," *Brazilian Journal of Operations & Production Management*, vol. 16, no. 3, pp. 537–549, 2019. [Cited on pages 1 and 2]

[2] G. Kartnig, B. Grösel, and N. Zrnic, "Past, state-of-the-art and future of intralogistics in relation to megatrends," *FME Transactions*, vol. 40, pp. 193–200, 1 2012. [Cited on pages 1 and 2]

[3] R. Miller, "How WinCC Unified Impacts the Future of HMIs." Available at `http://www.siemens.com/wincc-unified`, Mar. 2020. (Last accessed on 19/11/2022). [Cited on page 9]

[4] Siemens AG, "Efficiency needs flexibility." Available at `https://assets.new.siemens.com/siemens/assets/api/uuid:e6b0f585-b00f-424f-a98f-1cf0dbe6aecf/titelstory-wincc-unified-ind-autom-04-2021-en.pdf`, 2021. (Last accessed on 08/09/2023). [Cited on page 9]

[5] F. Sauer, M. Niedermaier, S. Kießling, and D. Merli, "Licster – a low-cost ics security testbed for education and research," *6th International Symposium for ICS & SCADA Cyber Security Research 2019 (ICS-CSR)*, p. 2, 2019. [Cited on pages ix and 10]

[6] K. S. Schulz, C. O'Brien, A. Rozenson, and M. P. Smith, "Human-machine interface system and method for remotely monitoring and controlling a machine." Available at `https://patentimages.storage.googleapis.com/3f/2e/b6/9adcca9749d745/US20050155043A1.pdf`, Jul 2005. (Last accessed on 19/02/2022). [Cited on page 10]

[7] P. Papcun, E. Kajáti, and J. Koziorek, "Human machine interface in concept of industry 4.0," *2018 World Symposium on Digital Intelligence for Systems and Machines (DISA)*, pp. 289–296, 2018. [Cited on pages 11 and 12]

[8] A. Ardanzaa, A. Morenoa, A. Seguraa, M. de la Cruzb, and D. Aguinagab, "Sustainable and flexible industrial human machine interfaces to support adaptable applications in the industry 4.0 paradigm," *International Journal of Production Research*, vol. 57, no. 12, pp. 4045–4059, 2019. [Cited on pages 11 and 12]

[9] Bill R. Hollifield, Dana Oliver, Ian Nimmo and Eddie Habibi, ed., *The High Performance HMI Handbook*. Houston, TX: 360 Digital Books, 2008. [Cited on pages ix, 13, 14, 15, and 16]

[10] International Society of Automation, "New ISA101 HMI technical report focuses on usability and performance." Available at `https://www.isa.org/intech-home/2019/july-august/departments/new-isa101-hmi-technical-report-focuses-on-usabili`, 2019. (Last accessed on 14/12/2022). [Cited on page 13]

[11] A. Ujvarosi, "Evolution of SCADA systems," *Bulletin of the Transilvania University of Braşov*, vol. 9, no. 1, pp. 63–68, 2016. [Cited on page 13]

[12] A. Sajid, H. Abbas, and K. Saleem, "Cloud-Assisted IoT-Based SCADA Systems Security: A Review of the State of the Art and Future Challenges," *IEEE Access*, vol. 4, pp. 1375–1384, 2016. [Cited on page 13]

[13] ABB, "How a high performance HMI can lead to greater awareness, faster response and better decisions." Available at `https://library.e.abb.com/public/d62da22416d94dbf9b60a4dd3efc1471/8VZZ000576T0000_High_Performance_HMI_WP_v3.pdf`, 2018. (Last accessed on 08/09/2023). [Cited on pages xiii, 16, and 17]

[14] A. Baji, U. Patil, and V. R. Nike, "Study of different types of conveyor system and their use accordingto the various needs of different industries," *National Conference on Excellence in Design, Manufacturing & Automation*, pp. 225–228, 2018. [Cited on page 17]

[15] project S&P GmbH, "Conveyor technology - PROCON." Available at `https://project-sp.de/en/our-products/conveyor-technology/`. (Last accessed on 25/02/2023). [Cited on pages ix, 18, 19, and 20]

[16] J. Prajapati, T. Soni, K. Shah, and A. Dwivedi, "Design of a simplified vertical conveyor system," *International Research Journal of Engineering and Technology*, vol. 3, no. 10, pp. 954–958, 2016. [Cited on page 19]

[17] Regal Rexnord Automation Solutions, "Vertical Case Conveyor." Available at `https://www.automation-solutions.com/product/vertical-case-conveyor/`. (Last accessed on 25/02/2023). [Cited on pages ix and 19]

[18] LEWCO, Inc., "Turntable." Available at `https://conveyors.lewcoinc.com/products/turntable/`. (Last accessed on 25/02/2023). [Cited on page 19]

[19] ATS Group, "Transfer Car: Dividing and Merging Flows Gets Simple." Available at `https://www.ats-group.com/EN/product-solutions/products/transfer-car.html`. (Last accessed on 25/02/2023). [Cited on pages ix and 20]

[20] Siemens AG, "STEP 7 and WinCC Engineering V17." Available at `https://cache.industry.siemens.com/dl/files/671/109798671/att_1071920/v1/STEP_7_WinCC_V17_enUS_en-US.pdf`, 2021. [Cited on pages ix and 24]

[21] Siemens AG, "Human Machine Interface Systems/PC-based Automation." Available at `https://cache.industry.siemens.com/dl/files/146/109744146/att_1110182/v1/simatic-st80-stpc-complete-english-2022.pdf`, 2022. [Cited on pages ix, 24, 25, 26, 27, 58, and 59]

[22] Siemens AG, "SIMATIC HMI Comfort Panels." Available at `https://new.siemens.com/global/en/products/automation/simatic-hmi/panels/comfort-panels.html`. (Last accessed on 30/11/2022). [Cited on page 25]

[23] Siemens AG, "SIMATIC HMI Mobile Panels." Available at `https://new.siemens.com/global/en/products/automation/simatic-hmi/panels/mobile-panels.html`. (Last accessed on 30/11/2022). [Cited on pages 25 and 26]

[24] Siemens AG, "SIMATIC WinCC RT Professional." Available at `https://new.siemens.com/global/en/products/automation/industry-software/automation-software/scada/simatic-wincc-professional-rt.html`. (Last accessed on 01/12/2022). [Cited on page 27]

[25] Siemens AG, "Visualization with WinCC Unified, now even more flexible!." Available at `http://www.siemens.com/wincc-unified`. (Last accessed on 19/11/2022). [Cited on pages ix and 28]

[26] Siemens AG, "WinCC Engineering V17 – WinCC Unified." Available at `https://cache.industry.siemens.com/dl/files/204/109794204/att_1069891/v1/WinCCVisualizingProcessesUnifiedenUS_en-US.pdf`, 2022. [Cited on page 28]

[27] Siemens AG, "Web visualization with WinCC Unified View of Things." Available at `https://new.siemens.com/global/en/products/automation/simatic-hmi/wincc-unified/view-of-things-web-hmi.html`. (Last accessed on 24/11/2022). [Cited on pages xiii and 29]

[28] Siemens AG, "WinCC Unified View of Things (VoT) - Getting started." Available at `https://cache.industry.siemens.com/dl/files/395/109803395/att_1089146/v2/109803395_ViewOfThings_FirstStep_DOC_en.pdf`, 2021. [Cited on page 29]

[29] Siemens AG, "Highlights of the new SIMATIC HMI Unified Comfort Panels." Available at `https://new.siemens.com/global/en/products/automation/simatic-hmi/wincc-unified/hardware.html`. (Last accessed on 21/11/2022). [Cited on pages ix and 30]

[30] Siemens AG, "Software for the visualization of the future." Available at https://new.siemens.com/global/en/products/automation/simatic-hmi/wincc-unified/software.html. (Last accessed on 03/12/2022). [Cited on page 30]

[31] Siemens AG, "SIMATIC WinCC Unified V17 Architectures." Available at https://assets.new.siemens.com/siemens/assets/api/uuid:680f6b71-4beb-439b-b631-b11867bd3bc8/simatic-wincc-unified-v17-architectures--techslides-2021-07-07-e.pdf, 2021. [Cited on pages ix and 31]

[32] Siemens AG, "SIMATIC HMI IWP10F Mobile WebClient." Available at https://mall.industry.siemens.com/mall/en/WW/Catalog/Product/6AV2145-6KM00-0AA0. (Last accessed on 08/09/2023). [Cited on pages ix, 31, and 32]

[33] Schneider Electric, "Basic HMI panels." Available at https://www.se.com/ww/en/product-subcategory/2140-basic-hmi-panels/. (Last accessed on 12/12/2022). [Cited on pages 32 and 33]

[34] Schneider Electric, "Advanced HMI panels." Available at https://www.se.com/ww/en/product-subcategory/88816-advanced-hmi-panels/. (Last accessed on 12/12/2022). [Cited on pages 32 and 33]

[35] Schneider Electric, "HMI Software." Available at https://www.se.com/ww/en/product-subcategory/82307-hmi-software/?filter=business-1-industrial-automation-and-control. (Last accessed on 12/12/2022). [Cited on page 33]

[36] Rockwell Automation, "PanelView Plus 7 Performance Terminals." Available at https://literature.rockwellautomation.com/idc/groups/literature/documents/td/2711p-td009_-en-p.pdf, 2022. [Cited on page 33]

[37] Rockwell Automation, "MobileView Tethered." Available at https://literature.rockwellautomation.com/idc/groups/literature/documents/pp/2711t-pp001_-en-p.pdf, 2017. [Cited on pages 33 and 34]

[38] Rockwell Automation, "PanelView Plus 7." Available at https://www.rockwellautomation.com/en-us/products/hardware/allen-bradley/human-machine-interface/graphic-terminals/2711p-panelview-plus-7.html. (Last accessed on 14/12/2022). [Cited on page 34]

[39] B&R Industrial Automation GmbH, "HMI." Available at https://www.br-automation.com/en/products/hmi/. (Last accessed on 05/01/2023). [Cited on page 34]

[40] B&R Industrial Automation GmbH, "Power Panel T80 User's manual." Available at https://download.br-automation.com/BRP4440000000000007164

71/MAPPT80-en_V2.03.1.pdf?px-hash=4d015d7e65c72d1d12932751addae332&px-time=1680169027, 2022. (Last accessed on 08/09/2023). [Cited on page 34]

[41] B&R Industrial Automation GmbH, "Mobile Panel 7100 series." Available at https://www.br-automation.com/en/products/hmi/mobile-panel-7100-series/. (Last accessed on 05/01/2023). [Cited on page 34]

[42] PI North America, "Countless Possibilities: PROFINET and Industrial Wireless." Available at https://us.profinet.com/white-papers/countless-possibilities-profinet-industrial-wireless, 2019. (Last accessed on 08/09/2023). [Cited on pages ix, x, xiii, 38, 39, 40, 42, 45, and 46]

[43] T. Nguyen, M. Tsurita, M. Toyofuku, Y. Ito, Y. Nagao, M. Kurosaki, and H. Ochi, "Development of Factory Automation WLAN System Compatible with Asynchronous Industrial Ethernet," *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 24, pp. 1338–1341, 2019. [Cited on page 38]

[44] V. Maxim, J. Seminsky, and V. Slanina, "Trends in industrial wireless communication and applications," *American Journal of Mechanical Engineering*, vol. 3, pp. 235–239, 2015. [Cited on pages 38 and 41]

[45] P. Zeng, Z. Wang, Z. Jia, L. Kong, D. Li, and X. Jin, "Time-slotted software-defined industrial ethernet for real-time quality of service in industry 4.0," *Future Generation Computer Systems*, vol. 99, pp. 1–10, 2019. [Cited on page 39]

[46] Siemens AG, "Basic information on configuring an Industrial Wireless LAN." Available at https://cache.industry.siemens.com/dl/files/063/90880063/att_950081/v1/90880063_Aufbau_IWLAN_DOKU_V50_en.pdf, 2018. (Last accessed on 08/09/2023). [Cited on pages ix, x, 39, 41, 42, 43, 44, 46, 47, 48, and 49]

[47] X. Wu and L. Xie, "On the wireless extension of profinet networks," *IEEE VTS Asia Pacific Wireless Communications Symposium (APWCS)*, pp. 1–5, 2019. [Cited on pages 39 and 40]

[48] Phoenix Contact, "The Industrial Ethernet network portfolio from Phoenix Contact." Available at https://dam-mdc.phoenixcontact.com/asset/156443151564/6c2b1eccd71bac4db79749194e3642b3/52003209_EN_Industrial_Ethernet_LoRes.pdf, 2021. (Last accessed on 08/09/2023). [Cited on page 39]

[49] R. Palacios, F. Granelli, D. Gajic, C. Liß, and D. Kliazovich, "An Energy-efficient Point Coordination Function Using Bidirectional Transmissions of Fixed Duration for Infrastructure IEEE 802.11 WLANs," *IEEE International Conference on Communications (ICC)*, pp. 2443–2448, 2013. [Cited on page 41]

[50] Siemens AG, "Understanding and Using iPRP." Available at https://cache.industry.siemens.com/dl/files/106/109761106/att_1040776/v4/109761106_iPRP_DOKU_V2_0_en.pdf, 2020. (Last accessed on 08/09/2023). [Cited on page 42]

[51] X. Li, D. Li, J. Wan, A. V. Vasilakos, C.-F. Lai, and S. Wang, "A review of industrial wireless networks in the context of industry 4.0," *Wireless Netw*, vol. 23, pp. 23–41, 2017. [Cited on page 46]

[52] Siemens AG, "Industrial Wireless LAN (IWLAN) – WLAN for industry." Available at https://cache.industry.siemens.com/dl/files/732/109807732/att_1131233/v1/IWLAN_EN.pdf, 2022. (Last accessed on 08/09/2023). [Cited on page 49]

[53] Siemens AG, "IWLAN – the Wireless LAN for demanding industrial applications." Available at https://new.siemens.com/global/en/products/automation/industrial-communication/industrial-wireless-lan.html. (Last accessed on 20/02/2023). [Cited on page 49]

[54] Siemens AG, "6GK5761-1FC00-0AA0." Available at https://mall.industry.siemens.com/mall/en/ww/Catalog/Product/6GK5761-1FC00-0AA0, 2023. (Last accessed on 30/03/2023). [Cited on page 50]

[55] Siemens AG, "6GK5774-1FX00-0AA0." Available at https://mall.industry.siemens.com/mall/en/ww/Catalog/Product/6GK5774-1FX00-0AA0, 2023. (Last accessed on 30/03/2023). [Cited on page 50]

[56] Siemens AG, "6GK5778-1GY00-0AA0." Available at https://mall.industry.siemens.com/mall/en/ww/Catalog/Product/6GK5778-1GY00-0AA0, 2023. (Last accessed on 30/03/2023). [Cited on page 50]

[57] Siemens AG, "6GK5795-6DC00-0AA0." Available at https://mall.industry.siemens.com/mall/en/ww/Catalog/Product/6GK5795-6DC00-0AA0, 2023. (Last accessed on 30/03/2023). [Cited on page 51]

[58] Siemens AG, "6GK5795-6MP00-0AA0." Available at https://mall.industry.siemens.com/mall/en/ww/Catalog/Product/6GK5795-6MP00-0AA0, 2023. (Last accessed on 30/03/2023). [Cited on page 51]

[59] Siemens AG, "6GK5795-6MN10-0AA6." Available at https://mall.industry.siemens.com/mall/en/ww/Catalog/Product/6GK5795-6MN10-0AA6, 2023. (Last accessed on 30/03/2023). [Cited on page 51]

[60] ISO/IEC, *ISO/IEC 24730-1:2014 - Information technology — Real-time locating systems (RTLS)*, 2014. [Cited on page 51]

[61] A. Löcklin, T. Ruppert, L. Jakab, R. Libert, N. Jazdi, and M. Weyrich, "Trajectory prediction of humans in factories and warehouses with real-time locating systems," *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, pp. 1317–1320, 2020. [Cited on page 52]

[62] F. M. F. Cordeiro, "Real-time location systems and internet of things sensors," Master's thesis, Faculdade De Engenharia Da Universidade do Porto, Porto, Portugal, 2019. [Cited on page 52]

[63] Sewio Networks, "SEWIO UWB Real-Time Location System (RTLS)." Available at `https://www.sewio.net/real-time-location-system-rtls-on-uwb/`. (Last accessed on 15/02/2023). [Cited on page 52]

[64] KINEXON, "Trust the GPS for Industry: Real-Time Localization via UWB." Available at `https://kinexon.com/technology/real-time-locating-system-rtls/`. (Last accessed on 15/02/2023). [Cited on page 52]

[65] Inpixon, "Real-Time Location Systems." Available at `https://www.inpixon.com/technology/rtls`. (Last accessed on 15/02/2023). [Cited on page 52]

[66] Siemens AG, "RTLS system for real-time locating." Available at `https://new.siemens.com/global/en/products/automation/industrial-identification/simatic-rtls.html`. (Last accessed on 15/02/2023). [Cited on page 52]

[67] Siemens AG, "Digital transformation - A guide for manufacturers." Available at `https://www.siemens.com/global/en/products/automation/identification-and-locating/simatic-rtls/rtls-whitepaper.html`, 2021. (Last accessed on 08/09/2023). [Cited on pages x, 53, 54, and 55]

[68] Siemens AG, "Transponders." Available at `https://mall.industry.siemens.com/mall/en/WW/Catalog/Products/10380211?tree=CatalogTree`. (Last accessed on 22/12/2022). [Cited on page 54]

[69] Siemens AG, "Gateways." Available at `https://mall.industry.siemens.com/mall/en/WW/Catalog/Products/10380209?tree=CatalogTree`. (Last accessed on 22/12/2022). [Cited on pages xiii, 55, 56, 57, and 60]

[70] Intel, "Ultra-Wideband (UWB) Technology - Enabling high-speed wireless personal area networks." Available at `https://www.3g4g.co.uk/Other/Uwb/Wp/Ultra-Wideband.pdf`, 2004. (Last accessed on 08/09/2023). [Cited on page 55]

[71] S. Faruque, *Phase Shift Keying (PSK)*. Grand Forks, ND, USA: Springer International Publishing, 2017. [Cited on page 55]

[72] Siemens AG, "Software." Available at https://mall.industry.siemens.com/mall/en/WW/Catalog/Products/10380224?tree=CatalogTree. (Last accessed on 22/12/2022). [Cited on page 56]

[73] A. Singh, A. Pal, D. Garg, and D. Yadav, "Location-based services using geofencing," *International Journal of Advance Research and Development*, vol. 3, pp. 85–88, 2018. [Cited on page 56]

[74] Siemens AG, "Data exchange between S7 controller and RTLS Locating Manager." Available at https://support.industry.siemens.com/cs/document/109782483/data-exchange-between-s7-controller-and-rtls-locating-manager?dti=0&lc=en-BH, 2023. (Last accessed on 08/09/2023). [Cited on pages x, 56, and 57]

[75] Honeywell International Inc., "RT10W Rugged Tablet Data Sheet." Available at https://prod-edam.honeywell.com/content/dam/honeywell-edam/sps/ppr/fr-fr/public/products/mobile-computers/tablets/rt10w/documents/sps-ppr-rt10w-rugged-tablet-data-sheet-en.pdf, 2021. (Last accessed on 08/09/2023). [Cited on page 60]

[76] Siemens AG, "Industrial Tablet PC - SIMATIC ITP1000." Available at https://cache.industry.siemens.com/dl/files/234/109745234/att_918320/v1/ipc_tablet_operating_instructions_en_en-US.pdf, 2020. (Last accessed on 08/09/2023). [Cited on page 60]

[77] Advantech, "AIM-68." Available at https://advdownload.advantech.com/productfile/PIS/AIM-68/file/AIM-68_B%E7%89%88_DS(082021)20210823130558.pdf, 2021. (Last accessed on 08/09/2023). [Cited on page 60]

[78] Getac USA, "UX10." Available at https://www.getac.com/wp-content/uploads/2020/09/Getac_UX10_US_20200902.pdf, 2020. (Last accessed on 08/09/2023). [Cited on page 60]

[79] Advantech Europe B.V., "AIM-68CT-C2101000." Available at https://buy.advantech.eu/Tablet-PCs/Rugged-Tablet-PCs/model-AIM-68CT-C2101000.htm. (Last accessed on 16/02/2023). [Cited on page 60]

[80] LOGISCENTER LLC, "Honeywell RT10W, 2D, SR, USB, BLUETOOTH, Wi-Fi, 10 IoT Enterprise." Available at https://www.logiscenter.pt/terminal-honeywell-rt10w-l00-17c12s0e. (Last accessed on 16/02/2023). [Cited on page 60]

[81] LOGISCENTER LLC, "Getac UX10 G2-EX, USB, BLUETOOTH, Wi-Fi, Win. 10 Pro, ATEX." Available at https://www.logiscenter.pt/terminal-getac-um21z4vixdxe. (Last accessed on 16/02/2023). [Cited on page 60]

[82] Siemens AG, "6AV7880-0AA22-1AA2." Available at `https://mall.industry.siemens.com/mall/en/PT/Catalog/Product/?mlfb=6AV7880-0AA22-1AA2&SiepCountryCode=PT`. (Last accessed on 17/02/2023). [Cited on page 60]

[83] Siemens AG, "6AV2128-3GB06-0AX1." Available at `https://mall.industry.siemens.com/mall/en/PT/Catalog/Product/?mlfb=6AV2128-3GB06-0AX1`. (Last accessed on 17/02/2023). [Cited on pages x and 61]

[84] Siemens AG, "SIMATIC Box PCs." Available at `https://mall.industry.siemens.com/mall/en/pt/Catalog/Products/5101001?tree=CatalogTree`. (Last accessed on 17/02/2023). [Cited on page 61]

[85] Siemens AG, "6AG4021-0AB11-1CA0." Available at `https://mall.industry.siemens.com/mall/en/PT/Catalog/Product/?mlfb=6AG4021-0AB11-1CA0`. (Last accessed on 17/02/2023). [Cited on page 62]

[86] Siemens AG, "6ES7647-8CF33-2AB3." Available at `https://mall.industry.siemens.com/mall/en/PT/Catalog/Product/?mlfb=6ES7647-8CF33-2AB3`. (Last accessed on 17/02/2023). [Cited on page 62]

[87] Siemens AG, "Guide for switching from WinCC Advanced to WinCC Unified." Available at `https://cache.industry.siemens.com/dl/files/002/109768002/att_1103718/v2/109768002_RTAdvanced_to_UnifiedPCRT_V20_DOC_en.pdf`, 2022. (Last accessed on 08/09/2023). [Cited on pages x and 66]

[88] "HMI Template Suite (WinCC Unified)." Available at `https://support.industry.siemens.com/cs/attachments/91174767/Final_91174767_HMITemplateSuiteUnified_V4.1_DOC_en.pdf`, 2023. (Last accessed on 08/09/2023). [Cited on pages xiii, 70, and 71]

[89] Siemens AG, "SIMATIC HMI WinCC (TIA Portal)." Available at `https://cache.industry.siemens.com/dl/files/308/109813308/att_1122197/v1/WinCC_VisualizingProcessesUnified_enUS_en-US.pdf`, 2022. (Last accessed on 08/09/2023). [Cited on pages 74 and 142]

[90] Siemens AG, "WinCC (TIA Portal) WinCC Engineering V18 – Communication." Available at `https://cache.industry.siemens.com/dl/files/307/109813307/att_1122146/v2/WinCC_Communicat_enUS_en-US.pdf`, 2022. (Last accessed on 08/09/2023). [Cited on page 89]

[91] Siemens AG, "How do I configure area pointers for WinCC Unified HMI operator panels?." Available at `https://support.industry.siemens.com/cs/document/109794243/how-do-i-configure-area-pointers-for-wincc-unified-hmi-operator-panels`, 2023. (Last accessed on 08/09/2023). [Cited on page 89]

Appendix A

Unified Scripting

A.1 Global Module Functions

The "ChangeMainContentScreen" function (Listing A.1) utilises a native Unified instruction to dynamically update the main content window with the desired process screen, which is passed as an argument when the function is called.

```
1 export function ChangeMainContentScreen(screen) {
2   HMIRuntime.UI.SysFct.ChangeScreen(screen, "~/SW_MainContent");
3 }
```

Listing A.1: Function to change the screen on the main content screen window.

On the "ChangeScreenRequest" function (Listing A.2), initially, the requested screen number is retrieved, and a switch case statement is employed to match the number with available screen numbers in the project. This enables the evaluation of the requested screen's name and facilitates the screen change. The first case represents a unique scenario where the value 1000 is tested, but it does not correspond to any specific screen number. Instead, the number 1000 is used to indicate a request to load the previously active elements screen. This request is made by the "Plant Overview" screen, which lists all the available elements in the plant. When a user selects one of these elements, the value 1000 is written to the "internal_ScreenRequestedFromFaceplate" tag and subsequently assessed by this function. If this case

is triggered, the "internal_ElementsPreviousScreen" tag is read, and its corresponding screen name is utilised to load the new screen. During the initial boot of the interface, as expected, the "internal_ElementsPreviousScreen" tag does not contain any value. In such cases, the "Multicontrol" screen is loaded by default. For the sake of brevity, most of the cases in the switch statement have been omitted here, as this is purely for demonstration purposes. Finally, before exiting the function, the "internal_ScreenRequestedFromFaceplate" tag is set to 0, allowing the evaluation of new screen change requests originating from the faceplates.

```
1  export function ChangeScreenRequest() {
2
3    let screen = Tags("internal_ScreenRequestedFromFaceplate");
4    let requestedScreen = screen.Read();
5
6    switch (requestedScreen) {
7      case 1000:
8        let elementsScreen = Tags("internal_ElementsPreviousScreen")
             ;
9        let previousScreen = elementsScreen.Read();
10       if (previousScreen == ""){
11         ChangeMainContentScreen("39_MULTICONTROL");
12       } else {
13         ChangeMainContentScreen(previousScreen);
14       }
15       break;
16     case 1:
17       ChangeMainContentScreen("01_PLANT_STATE");
18       break;
19 // ...
20   }
21
22   screen.Write(0);
23 }
```

Listing A.2: Function to change the main content screen from a faceplate request.

Regarding the functionality of the "ElementChanged" global module function (Listing A.3), it initially identifies the instance of Unified that triggered its execution to utilise the correct tag table. Next, it reads the number of the currently displayed process screen, and if the value falls between 10 and 30, it confirms that a screen related to the elements is open. In this case, the corresponding bit of the "Usability" word, stored in the variable "usabilityBit", is determined based on the active screen. For the interface screens, since they will be under a particular evaluation, the usability bits are defined via two constants. After extracting the required bit for

validations, the "IDB_HMI_S_SE_OPERATION.OVERVIEW.USABILITY" tag is read and stored.

The values of "Usability" and the usability bits are utilised to create four variables, enabling the content within them to be used for validations. The first variable, "noUsability", indicates whether the currently active screen is not an interface and if the newly selected element does not support it. The "interfaceNoUsability" variable is used to determine if the process screen is an interface and if the element supports any of them. The remaining two variables, "interfaceTopDownNoUsability" and "interfaceFrontRearNoUsability", indicate if the currently active interface screen is not supported while the other one is. These four variables allow for the execution of usability checks for all screens using conditional statements (if-else).

In the first scenario, when the current process screen is not supported by the newly selected element, the screen is changed to the default "Multicontrol" since it does not rely on any specific element. Additionally, the side navigation is displayed to inform the operator about the available screen options. The last two conditions apply to interface screens; if one interface is not available but the other is, the screen is changed to the appropriate one.

```
1  export function ElementChanged() {
2
3    let idbName = Tags("internal_IDBName").Read();
4    let currentScreen = Tags(idbName+"_HMI_S_RANGE_POINTER_IMAGE").
         Read();
5
6    if(currentScreen > 10 && currentScreen < 30) {
7
8      let usabilityBit = 1000;
9
10     switch (currentScreen) {
11       case 11:  // Control
12       usabilityBit = 0;
13       break;
14 //...
15      }
16
17     const usabilityBitInterfaceFR = 4;
18     const usabilityBitInterfaceTD = 5;
19
20     let usability = Tags(idbName+"_HMI_S_SE_OPERATION.OVERVIEW.
           USABILITY").Read();
21
22     let noUsability = currentScreen != 15 && currentScreen != 20
           && (((usability & (1 << usabilityBit)) === 0 ? 0 : 1) == 0)
           ;
23
```

```
24      let interfaceNoUsability = (currentScreen == 15 ||
            currentScreen == 20) && (((usability & (1 <<
            usabilityBitInterfaceFR)) === 0 ? 0 : 1) == 0) && (((
            usability & (1 << usabilityBitInterfaceTD)) === 0 ? 0 : 1)
            == 0);
25
26      let interfaceTopDownNoUsability = currentScreen == 20 && (((
            usability & (1 << usabilityBitInterfaceFR)) === 0 ? 0 : 1)
            == 1) && (((usability & (1 << usabilityBitInterfaceTD)) ===
             0 ? 0 : 1) == 0);
27
28      let interfaceFrontRearNoUsability = currentScreen == 15 && (((
            usability & (1 << usabilityBitInterfaceFR)) === 0 ? 0 : 1)
            == 0) && (((usability & (1 << usabilityBitInterfaceTD)) ===
             0 ? 0 : 1) == 1);
29
30      if (noUsability || interfaceNoUsability) {
31        ChangeMainContentScreen("39_MULTICONTROL");
32        MainNavigationVisible();
33      } else if (interfaceTopDownNoUsability) {
34        ChangeMainContentScreen("15_SE_INTERFACE_CONV");
35      } else if (interfaceFrontRearNoUsability) {
36        ChangeMainContentScreen("20_SE_INTERFACE_HANDL");
37      }
38    }
39    Tags("internal_SignalElementChanged").Write(0);
40  }
```

Listing A.3: Function to change the main content screen when the element is changed.

By invoking the "MainNavigationHide" function (Listing A.4), it becomes possible to hide all the screen windows that contain navigation screens, including the button intended to hide the side navigation. The `FindItem(screenItemPath)` method is utilised to accomplish this task, requiring only the name of the item to be manipulated as a parameter. When the item to be manipulated exists on the screen calling the instruction, using `Screen.FindItem` allows passing just the item name as a parameter. However, because the method is being called from a global module function in this case, the absolute path must be utilised to locate the item that needs to be altered. For instance, the "SW_Navigation_Elements" item is located on the start screen, hence the "`~/`" forms the initial part of the path. Lastly, to modify the visibility of any item, the "Visible" property can be employed.

```
1  export function MainNavigationHide() {
2
3    Screen.FindItem("~/SW_NavigationElements").Visible = false;
```

```
4    Screen.FindItem("~/SW_NavigationDiagnosis").Visible = false;
5    Screen.FindItem("~/SW_NavigationService").Visible = false;
6    Screen.FindItem("~/BTN_HideMainNavigation").Visible = false;
7  }
```

Listing A.4: Function to hide the side navigation.

To display the side navigation, the function "MainNavigationVisible" (refer to Listing A.5) is utilised. This function retrieves the HMI instance name and the currently active screen number. Conditions are then evaluated to determine the category to which the active process screen belongs. Once a condition is satisfied, the corresponding screen window from the layout screen is made visible. Additionally, the invisible button, which enables the side navigation to be hidden, is also made visible.

```
1  export function MainNavigationVisible() {
2
3    let idbName = Tags("internal_IDBName").Read();
4
5    let screen = Tags(idbName+"_HMI_S_RANGE_POINTER_IMAGE");
6    let currentScreen = screen.Read();
7
8    if (currentScreen > 10 && currentScreen < 40) {
9      Screen.FindItem("~/SW_NavigationElements").Visible = true;
10
11   } else if (currentScreen > 60 && currentScreen < 70) {
12     Screen.FindItem("~/SW_NavigationDiagnosis").Visible = true;
13
14   } else if (currentScreen > 70 && currentScreen < 90) {
15     Screen.FindItem("~/SW_NavigationService").Visible = true;
16   }
17
18   Screen.FindItem("~/BTN_HideMainNavigation").Visible = true;
19 }
```

Listing A.5: Function to display the side navigation.

The "ChangeLayout" function (Listing A.6) is responsible for adjusting the screen layout according to the operator's preferences, which are determined through choices made on the settings screen. The function takes input parameters that specify the desired positions for the information and menu bars, as well as the navigation elements. Within the local tags region, several variables are created to enhance the script's portability. These variables primarily store the necessary values related to the screen window dimensions for constructing the layout. Next, a switch case statement is executed based on the desired position of the menu bar. There are four

possible positions for the menu bar, while the information bar and navigation can only assume two positions. When a case is entered, the screen windows are positioned or resized based on the content of the "infoPosition" and "navigationPosition" variables. This is achieved using the functions "ChangePos" and "ChangeSizePos", respectively. The parameters for these functions are derived from the variables defined at the beginning of the function. The screen used on the menu bar screen window is also changed in order to accommodate horizontal or vertical positions.

```
1  export function ChangeLayout(infoPosition, menuPosition,
       navigationPosition) {
2
3    /* Local tags ---------------- */
4    const panelWidth = 800;
5    const panelHeight = 480;
6    /* Information bar size */
7    const infoHeight = 32;
8    /* Navigation handle size */
9    const handleSize = 50;
10   /* Content size when menu is in horizontal position */
11   const menuHeight = 38;
12   const contentHeight = 410;
13   const naviWidth = 240;
14   const naviHeight = 410;
15   /* Content size when menu is in vertical position */
16   const menuVerticalWidth = 36;
17   const menuVerticalHeight = panelHeight - infoHeight;
18   const contentWidthVertical = panelWidth - menuVerticalWidth;
19   const contentHeightVertical = panelHeight - infoHeight;
20   const naviHeightVertical = 420;
21   /* Main navigation and content position */
22   let mainLeftPosition;
23   let mainTopPosition;
24   /* Screens names */
25   let headerMenu = "1_CS_Header_Menu";
26   let headerMenuVertical = "2_CS_Header_Menu_Vertical";
27   /* -------------------------------- */
28
29   switch (menuPosition) {
30     /* Menu horizontal in top position */
31     case 0:
32       HMIRuntime.UI.SysFct.ChangeScreen(headerMenu, "~/SW_Header")
           ;
33       /* Information bar in top position */
34       if (infoPosition == 0) {
35         mainLeftPosition = 0;
36         mainTopPosition = infoHeight + menuHeight;
```

```
37          ChangeSizePos("SW_Header", 0, infoHeight, menuHeight,
                panelWidth);
38          ChangeSizePos("SW_MainContent", mainLeftPosition,
                mainTopPosition, contentHeight, panelWidth);
39          ChangePos("SW_Info", 0, 0);
40        /* Information bar in down position */
41        } else if (infoPosition == 1) {
42          mainLeftPosition = 0;
43          mainTopPosition = menuHeight;
44          ChangeSizePos("SW_Header", 0, 0, menuHeight, panelWidth);
45          ChangeSizePos("SW_MainContent", mainLeftPosition,
                mainTopPosition, contentHeight, panelWidth);
46          ChangePos("SW_Info", 0, (menuHeight + contentHeight));
47        }
48        /* Navigation in left position */
49        if (navigationPosition == 0) {
50          ChangePos("SW_NavigationHandle", mainLeftPosition, (
                mainTopPosition+contentHeight-handleSize));
51          ChangeSizePos("SW_NavigationElements", mainLeftPosition,
                mainTopPosition, contentHeight, naviWidth);
52          ChangeSizePos("SW_NavigationDiagnosis", mainLeftPosition,
                mainTopPosition, contentHeight, naviWidth);
53          ChangeSizePos("SW_NavigationService", mainLeftPosition,
                mainTopPosition, contentHeight, naviWidth);
54        /* Navigation in right position */
55        } else if (navigationPosition == 1) {
56          ChangePos("SW_NavigationHandle", panelWidth-handleSize, (
                mainTopPosition+contentHeight-handleSize));
57          ChangeSizePos("SW_NavigationElements", panelWidth-
                naviWidth, mainTopPosition, contentHeight, naviWidth);
58          ChangeSizePos("SW_NavigationDiagnosis", panelWidth-
                naviWidth, mainTopPosition, contentHeight, naviWidth);
59          ChangeSizePos("SW_NavigationService", panelWidth-naviWidth
                , mainTopPosition, contentHeight, naviWidth);
60        }
61        break;
62
63  // ...
64
65    }
66  }
```

Listing A.6: Function to change the screen layout.

The “ChangePos” function (Listing A.7) enables the repositioning of the screen window provided as a parameter. By “utilising” the “X” and “Y” parameters, the top and left properties of the screen window can be defined.

```
1 export function ChangePos(screenWindow, X, Y) {
2
3   let windowPath = "~/" + screenWindow;
4
5   Screen.FindItem(windowPath).Top = Y;
6   Screen.FindItem(windowPath).Left = X;
7 }
```

Listing A.7: Function to change the position of a screen window.

The "ChangePosSize" function (Listing A.8) shares similarities with the previous function, but it offers an additional capability. In addition to repositioning the screen window provided as a parameter, it also allows for resizing of the window.

```
1 export function ChangeSizePos(screenWindow, X, Y, height, width) {
2
3   let windowPath = "~/" + screenWindow;
4
5   Screen.FindItem(windowPath).Top = Y;
6   Screen.FindItem(windowPath).Left = X;
7   Screen.FindItem(windowPath).Height = height;
8   Screen.FindItem(windowPath).Width = width;
9 }
```

Listing A.8: Function to change the size and position of a screen window.

A.2 Local Scripts

As outlined in Chapter 6, multiple local scripts were utilised on screens and faceplates to execute a range of operations that do not necessitate inclusion within the Global Module Functions' domain. Listings A.9 through A.24 contain examples of these scripts.

```
1 export function __CS_ScreenLayout_OnLoaded(item) {
2
3   Tags("internal_IDBName").Write(RetrieveIDB());
4   let idbName = Tags("internal_IDBName").Read();
5
6   Screen.Items("FC_AreaPointers").Properties.LifeBit.Tag = idbName
        +"_HMI_S_RANGE_POINTER_COORDINATION";
7   Screen.Items("FC_AreaPointers").Properties.ControlOrder.Tag =
        idbName+"_HMI_S_RANGE_POINTER_CONTROL_ORDER";
8
```

```
9    UI.RootWindow.InteractiveZooming = false;
10 }
```

Listing A.9: Script executed on the loading event of the start screen.

```
1  export function BTN_UserLogin_OnDown(item, x, y, modifiers,
       trigger) {
2
3    let userName = Tags("@UserName").Read();
4
5    if (userName == "" || userName == "DefaultUser") {
6      HMIRuntime.UI.UserManagement.SysFct.ShowLoginDialog();
7    } else {
8      HMIRuntime.UI.SysFct.LogOff();
9    }
10 }
```

Listing A.10: Script executed when the login button is pressed.

```
1  export function BTN_UserLogin_Text_OnPropertyChanged(item, value)
       {
2
3    let idbName = Tags("internal_IDBName").Read();
4    let userName = Tags("@UserName").Read();
5
6    Tags(idbName+"_HMI_S_USER_ADMIN_HMI_SIGNED_USERNAME").Write(
         userName);
7
8    if (userName == "" || userName == "DefaultUser") {
9      Tags(idbName+"_HMI_S_USER_LOG_OFF").Write(1);
10     MainNavigationHide();
11   }
12 }
```

Listing A.11: Script executed when the logged in user changes.

```
1  export function TA_ShowSettings_OnGestureDetected(item, gesture) {
2
3    if (gesture == UI.Enums.HmiGesture.SwipeDown || gesture == UI.
         Enums.HmiGesture.SwipeUp) { Screen.FindItem("~/SW_Settings").
         Visible = true; }
4  }
```

Listing A.12: Script executed when a gesture is detected on the information bar touch area.

```
export function TB_CurrentScreen_Text_Trigger(item) {
  var value;
  let txtListIndex = Tags("RangePointerImage").Read();
  let txtList = Faceplate.Properties.Current_Screen_Name;
  value = HMIRuntime.Resources.TextLists(txtList).Item(
      txtListIndex).Item(HMIRuntime.Language);
  return value;
}
```

Listing A.13: Script to retrieve the currently active process screen name.

```
export function BTN_Elements_OnDown(item, x, y, modifiers, trigger
    ) {

  MainNavigationHide();

  let screen = Tags("internal_ElementsPreviousScreen");
  let previousScreen = screen.Read();

  if (previousScreen == "") {
      ChangeMainContentScreen("39_MULTICONTROL");
  } else {
      ChangeMainContentScreen(previousScreen);
  }
}
```

Listing A.14: Script executed when the elements category button is pressed on the menu bar.

```
export function TB_Script_BackColor_Trigger(item) {
var value;

  let screenHeight = 68; // First dynamic entry starts at 68
      pixels from top
  const entryHeight = 60;

  let usability = Tags("Operation.OVERVIEW.USABILITY");
  let overviewUsability = usability.Read();

  let elementChanged = Tags("ElementChanged");
  elementChanged.Write(1);  // To trigger the evaluation of the
      usability of the current screen

  /* Check Word bits from 0 to 11 */
  for (let i = 0; i < 12; i++) {
```

```
15
16      /* Entry can be operated */
17      if ( ((overviewUsability & (1 << i)) === 0 ? 0 : 1) == 1) {
18
19        // Datatracking
20        if (i == 7) {
21            Faceplate.Items("TB_Entry_" + i).Visible = true;
22            Faceplate.Items("TB_Entry_" + i).Top = screenHeight;
23            screenHeight = screenHeight + entryHeight;
24            Faceplate.Items("TB_Entry_" + i + "_1").Visible = true;
25            Faceplate.Items("TB_Entry_" + i + "_1").Top =
                  screenHeight;
26            screenHeight = screenHeight + entryHeight;
27            Faceplate.Items("TB_Entry_" + i + "_2").Visible = true;
28            Faceplate.Items("TB_Entry_" + i + "_2").Top =
                  screenHeight;
29            screenHeight = screenHeight + entryHeight;
30
31          /* Special images */
32        } else if (i == 9) {
33          /* Verify the entries of the special display */
34          for (let j = 0; j < 4; j++) {
35             let displayFirst4 = Tags("Operation.SPECIAL_DISPLAY.
                   OVERVIEW_TEXT["+j+"].VISIBLE");
36             let displayFirst4Visible = displayFirst4.Read();
37             let displayLast4 = Tags("Operation.SPECIAL_DISPLAY.
                   OVERVIEW_PIC["+j+"].VISIBLE");
38             let displayLast4Visible = displayLast4.Read();
39
40             if (displayFirst4Visible == 1) {
41               Faceplate.Items("TB_Entry_" + i + "_1_" + j).Visible
                     = true;
42               Faceplate.Items("TB_Entry_" + i + "_1_" + j).Top =
                     screenHeight;
43               screenHeight = screenHeight + entryHeight;
44             }
45             if (displayLast4Visible == 1) {
46               Faceplate.Items("TB_Entry_" + i + "_2_" + j).Visible
                     = true;
47               Faceplate.Items("TB_Entry_" + i + "_2_" + j).Top =
                     screenHeight;
48               screenHeight = screenHeight + entryHeight;
49             }
50          }
51        } else {
52            Faceplate.Items("TB_Entry_" + i).Visible = true;
53            Faceplate.Items("TB_Entry_" + i).Top = screenHeight;
54            screenHeight = screenHeight + entryHeight;
55        }
```

```
56        Faceplate.Height = screenHeight; // Adjust faceplate height
              based on the number of visible entries
57
58      /* Entry can not be operated */
59      } else {
60        /* Data tracking */
61        if (i == 7) {
62            Faceplate.Items("TB_Entry_" + i).Visible = false;
63            Faceplate.Items("TB_Entry_" + i + "_1").Visible = false;
64            Faceplate.Items("TB_Entry_" + i + "_2").Visible = false;
65        /* Special images */
66        } else if (i == 9) {
67            for (let j = 0; j < 4; j++) {
68              Faceplate.Items("TB_Entry_" + i + "_1_" + j).Visible =
                    false;
69              Faceplate.Items("TB_Entry_" + i + "_2_" + j).Visible =
                    false;
70            }
71        } else {
72            Faceplate.Items("TB_Entry_" + i).Visible = false;
73        }
74      }
75    }
76
77    let navHeight = Tags("ElementsNavigationHeight");
78    navHeight.Write(screenHeight);
79
80  return value;
81  }
```

Listing A.15: Script to dynamically adjust the entries on the elements side navigation.

```
1  export function BTN_CleaningScreen_OnDown(item, x, y, modifiers,
       trigger) {
2    Screen.Items("TB_CleaningScreen").Visible = true;
3    Screen.Items("GV_CleaningScreen_1").Visible = true;
4    Screen.Items("TA_HideSettings").Visible = false;
5
6    HMIRuntime.Timers.SetTimeout(callbackFromTimer, 5000);
7    function callbackFromTimer() {
8      Screen.Items("TB_CleaningScreen").Visible = false;
9      Screen.Items("GV_CleaningScreen_1").Visible = false;
10     Screen.Items("TA_HideSettings").Visible = true;
11   }
12 }
```

Listing A.16: Script to display the cleaning screen content.

```
export function SVG_RollerConveyor_OnTapped(item, x, y, modifiers,
    trigger) {

  while (Tags("Operation.ADMIN.SE_INPUT[0].DISPLAY.LABEL").Read()
      != "01RF05") {
    HMIRuntime.Tags.SysFct.SetTagValue("Operation.ADMIN.SE_INPUT
        [0].DISPLAY.LABEL", "01RF05");
  }

  let groupFault = Tags("Operation.ADMIN.SE_INPUT[0].DISPLAY.ERROR
      ").Read();

  if (groupFault == 1) {
    let po = Faceplate.OpenFaceplateInPopup("
        ElementContent_V_0_0_35", "01RF05", false, false);
    po.Left = 0;
    po.Top = 0;
    po.Visible = true;

  } else {
    HMIRuntime.Tags.SysFct.SetTagValue("RequestedScreen", 1000);
  }
}
```

Listing A.17: Script to load element process screen or open pop-up with fault on the graphic plant overview.

```
export function Faceplate_type_OnLoaded(item) {

  let currentElement = Tags("Operation.ADMIN.SE_INPUT[0].DISPLAY.
      LABEL").Read();

  if (Tags("localElement").Read() == "" && currentElement != "") {

    Tags("localElement").Write(currentElement);

    Faceplate.Items("AC_Alarm").Filter = "AlarmText2 = "+"'"+Tags(
        "localElement").Read()+"'"+" AND AlarmClassName = '
        INTRALOG_ERROR'";
  }
}
```

Listing A.18: Script executed on loading of the element content pop-up by pressing a conveyor on the graphic plant overview.

```
export function TB_Script_BackColor_Trigger(item) {

  let currentElement = Tags("Operation.ADMIN.SE_INPUT[0].DISPLAY.
      LABEL").Read();

  Faceplate.Items("AC_Alarm").Filter = "AlarmText2 = "+"'"+
      currentElement+"'"+" AND AlarmClassName = 'INTRALOG_ERROR'";

  let usability = Tags("Operation.OVERVIEW.USABILITY").Read();
  let rollerConveyor = Tags("Operation.DETAILS_PIC[0].VIEW.RC").
      Read();
  let lift = Tags("Operation.DETAILS_PIC[0].VIEW.LIFT").Read();
  let belt = Tags("Operation.DETAILS_PIC[0].VIEW.BELT").Read();
  let arm = Tags("Operation.DETAILS_PIC[0].VIEW.ARM").Read();
  let turntable = Tags("Operation.DETAILS_PIC[0].VIEW.TURNTABLE").
      Read();
  let shuttle = Tags("Operation.DETAILS_PIC[0].VIEW.SHUTTLE").Read
      ();

  if (((usability & (1 << 10)) === 0 ? 0 : 1) == 0) {
    Faceplate.Layers("Layer_1").Visible = false;
    Faceplate.Layers("Layer_2").Visible = false;
    Faceplate.Layers("Layer_3").Visible = false;
    Faceplate.Layers("Layer_4").Visible = false;
    Faceplate.Layers("Layer_5").Visible = false;
    Faceplate.Layers("Layer_6").Visible = true;
    Faceplate.Layers("Layer_7").Visible = false;

  } else if (rollerConveyor == 1 && lift == 0 && turntable == 0 &&
       arm == 0 && belt == 0 && shuttle == 0) {
    Faceplate.Layers("Layer_1").Visible = true;
    Faceplate.Layers("Layer_2").Visible = false;
    Faceplate.Layers("Layer_3").Visible = false;
    Faceplate.Layers("Layer_4").Visible = false;
    Faceplate.Layers("Layer_5").Visible = false;
    Faceplate.Layers("Layer_6").Visible = false;
    Faceplate.Layers("Layer_7").Visible = false;

// ...
}
```

Listing A.19: Script to manage the content of “Detail” faceplate.

```
export function TA_ShowPlantOverview_OnGestureDetected(item,
    gesture) {

  if (gesture == UI.Enums.HmiGesture.SwipeUp) {
```

```
4
5       let idbName = Tags("internal_IDBName").Read();
6
7       Tags(idbName+"_HMI_S_OVERVIEW.SE_INPUT").Write(0);
8
9       ChangeMainContentScreen("09_PLANT_OVERVIEW");
10    }
11  }
```

Listing A.20: Script to detect gesture and access the "Plant Overview" screen from the elements footer.

```
1  export function BTN_ShowHideBuffer_OnTapped(item, x, y, modifiers,
       trigger) {
2
3    let ui = UI;
4    let count = ui.PopupScreenWindows.Count;
5
6    if(count == 0) {
7
8      HMIRuntime.Tags.SysFct.SetBitInTag("OperationTracking.BUFFER.
         VISIBLE", 0);
9
10     let po = Faceplate.OpenFaceplateInPopup("
         DataTrackingPartnersBufferPopUp_V_0_0_4", "Buffer", false,
         false);
11     po.Left = 0;
12     po.Top = 0;
13     po.WindowFlags = ui.Enums.HmiWindowFlag.CanMove | ui.Enums.
         HmiWindowFlag.CanSize | ui.Enums.HmiWindowFlag.ShowBorder |
          ui.Enums.HmiWindowFlag.ShowCaption;
14     po.Visible = true;
15     Faceplate.Items("BTN_ShowHideBuffer").Text = "hide buffer";
16
17   } else {
18     HMIRuntime.Tags.SysFct.ResetBitInTag("OperationTracking.BUFFER
         .VISIBLE", 0);
19     ui.PopupScreenWindows(0).Close();
20     Faceplate.Items("BTN_ShowHideBuffer").Text = "show buffer";
21   }
22 }
```

Listing A.21: Script to manage the buffer pop-up visibility.

```
export function BTN_ShowWarnings_OnDown(item, x, y, modifiers,
    trigger) {

  Screen.Items("AC_Alarm").Filter = "AlarmClassName = '
      INTRALOG_WARNING'";

  Screen.Items("BTN_ShowInfo").BorderColor = HMIRuntime.Math.RGB
      (72, 73, 78);
  Screen.Items("BTN_ShowErrors").BorderColor = HMIRuntime.Math.RGB
      (72, 73, 78);
  Screen.Items("BTN_ShowWarnings").BorderColor = HMIRuntime.Math.
      RGB(0, 161, 209);
  Screen.Items("BTN_ShowEstop").BorderColor = HMIRuntime.Math.RGB
      (72, 73, 78);
  Screen.Items("BTN_ShowCabinet").BorderColor = HMIRuntime.Math.
      RGB(72, 73, 78);
  Screen.Items("BTN_ShowAll").BorderColor = HMIRuntime.Math.RGB
      (72, 73, 78);

}
```

Listing A.22: Script to change the filter of the alarm control.

```
export function AC_Alarm_OnOnSelectionChanged(item,
    SelectedRowData) {

  let selectedRowText = SelectedRowData.EventText;
  let splitText = selectedRowText.split(" ");
  let elementFromdRow = splitText[0];

  if (elementFromdRow != "") {

    let idbName = Tags("internal_IDBName").Read();
    Tags(idbName+"_HMI_S_SE_OPERATION.ADMIN.SE_INPUT[0].DISPLAY.
        LABEL").Write(elementFromdRow);

    let previousScreen = Tags("internal_ElementsPreviousScreen").
        Read();

    if (previousScreen == "") {
        ChangeMainContentScreen("39_MULTICONTROL");
    } else {
        ChangeMainContentScreen(previousScreen);
    }
  }
}
```

Listing A.23: Script executed when an elements alarm is selected.

```
1  export function txtDescription_Text_Trigger(item) {
2    var value;
3
4    let cpNumber = Tags("Access.ASK_NO.CP_NO").Read();
5    let groupNumber = Tags("Access.ASK_NO.GROUP_NO").Read();
6
7    value = "The CP "+cpNumber+" wants to control group "+
         groupNumber+". Do you want to give authorization ?";
8
9    return value;
10 }
```

Listing A.24: Script to construct the message of the authorisation request pop-up.

Appendix B

PLC Programs

B.1 Adjustments to the Existing PLC Programs

As previously mentioned in Chapter 6, an adjustment was made to the existing PLC code regarding the functionality of the "Operation Mode" screen in order to achieve a fast scroll through the groups. The new code used to achieve this new feature is presented in Listing B.1.

```
1  IF #S_SCROLL.PB_FWD THEN
2      #S_SCROLL.PB_FWD := false;
3      #t_Int := (#S_OPERATION_TYPE.GROUP_COUNT - "
           C_OPERATING_TYPE_GROUP_UNIFIED" + 1) - #S_MEM.
           OPERATION_TYPE.GROUP_COU;
4      IF #t_Int > "C_OPERATING_TYPE_GROUP_UNIFIED" THEN
5          #S_MEM.OPERATION_TYPE.GROUP_COU := #S_MEM.OPERATION_TYPE.
               GROUP_LAST + 1;
6      ELSIF #t_Int < 8 THEN
7          #S_MEM.OPERATION_TYPE.GROUP_COU += #t_Int;
8      ELSE
9          #S_MEM.OPERATION_TYPE.GROUP_COU += 8;
10     END_IF;
11 END_IF;
```

Listing B.1: PLC code adjustments to provide a faster scroll over the groups on the "Operation Mode" screen.

B.2 Operation Mode Selection View Screen

The working principle of the PLC code behind the "Operation Mode Selection View" screen (Figure B.1) lays primarily on the evaluation of the active bits on a Dword data type variable named "S_OPERATION_TYPE.GROUP_SELECT".

At the beginning of the function, the data stored on this variable is copied to an auxiliary one in order to manipulate and therefore evaluate the word content without disrupting the user selection.

Then a for cycle is initiated to go through all the groups and validate if they are available to be controlled by the CP in analysis. Furthermore, the retrieval of the relevant data to display on the screen, like the group name and its operational state, is performed.

After that, the first bit of the auxiliary group selection word is tested, and if it is in a true state, the state of the operations buttons are tested as well, in order to check if the user requested any operation. If an operation is requested, it will be handled accordingly on the "DB_OPERATION".

After this procedure is completed, the bits of the auxiliary word will be shifted to the right, making it possible to check if the user performed the selection of the the next group.

Before exiting the for loop it is checked if there are any more groups available.

At last, before exiting the function, the state of the operation buttons are reset in order to allow the detection of a new press. This procedure is only made before exiting the function in order to make sure that the operation is applied to all the selected groups and not only to the first, as would occur if the button was reset inside its pressed condition.

Currently, the selection of groups is limited to 32 due to the maximum capacity of a DWord variable in terms of bits. However, this limitation should not be a concern as it is highly unlikely to have more than 32 controllable groups for each CP. Nevertheless, if there is a need to exceed this limit, one solution is to introduce an additional DWord variable and incorporate scroll buttons in the visualization. By doing so, it would be possible to navigate through all the groups, thereby extending the limit to 64.

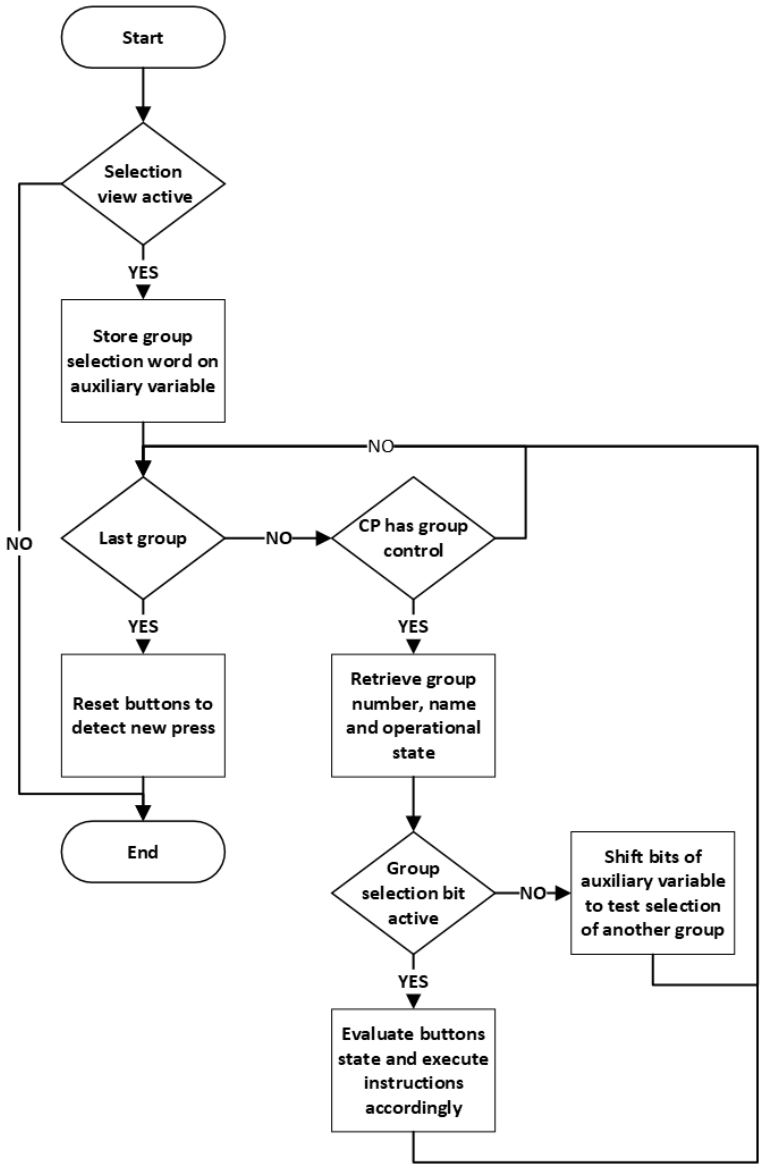


Figure B.1: PLC code behind the operation mode selection view.

B.3 New Screens for Data Tracking

In order to provide data from the PLC to the HMI regarding data tracking, in addition to the creation of several data types, two FB must be edited as they exchange data between them in order to accomplish the desired functionalities previously described for the data tracking screens.

These FB are the "FB_HMI_PANEL_UNIFIED" and "FB_HMI_PANEL_-TRACKING_UNIFIED". The first is in charge of passing to the second the conveyor data tracking place (referred in the code as "count") that the user is trying to access from the visualization interface. Then, from this number, the tracking FB can

retrieve the appropriate data and also allow it to be transferred between places and to or from the buffer. Moreover, the first FB will also inform about the process screen that is currently being used in the visualization, to ensure that only the necessary code is processed by the PLC. These instructions are performed on one of the last networks of the "FB_HMI_PANEL_UNIFIED", refer to Listing B.2.

```
1 #S_SE_OPERATION_TRACKING_DATA.ACTIVE_PARTNERS := #t_activ_Pic.
      SE_TRACKING_PARTNERS;
2 #S_SE_OPERATION_TRACKING_DATA.TRACKING_PARTNERS[1].COUNT := #
      S_SE_OPERATION.TRACKING_PARTNERS[1].COUNT;
3 #S_SE_OPERATION_TRACKING_DATA.TRACKING_PARTNERS[2].COUNT := #
      S_SE_OPERATION.TRACKING_PARTNERS[2].COUNT;
4 #S_SE_OPERATION_TRACKING_DATA.TRACKING_PARTNERS[3].COUNT := #
      S_SE_OPERATION.TRACKING_PARTNERS[3].COUNT;
5
6 #S_SE_OPERATION_TRACKING_DATA.ACTIVE_PLACES := #t_activ_Pic.
      SE_TRACKING_PLACES;
7 FOR #t_loop_places := 1 TO 6 DO
8     #S_SE_OPERATION_TRACKING_DATA.TRACKING_PLACES[#t_loop_places].
          COUNT := #S_SE_OPERATION.TRACKING_PLACES[#t_loop_places].
          COUNT;
9 END_FOR;
```

Listing B.2: PLC code for transferring data from the "FB_HMI_-PANEL_UNIFIED" to the "FB_HMI_PANEL_TRACKING_-UNIFIED".

B.3.1 Partners Data Tracking

Regarding the network dedicated to "Partners Data Tracking", on the "FB_HMI_-Panel_UNIFIED", initially the index of the front and rear partners of the currently selected element is retrieved from the DB. Then a loop through the three elements is performed, and the appropriate index is defined. For each conveyor, by checking the usage of the places scroll button, the places count is adjusted, as are the validations to hide the scroll buttons if there are no more places available. After that, some more relevant data, like the occupied state of the conveyors and the partners names, is retrieved from the appropriate memory areas in order to provide that information through the visualization. Lastly, whenever a partner button is pressed from the graphical representation of the conveyors, that element is loaded into the main content area. Out of the loop, the last instructions performed inside this network are executed to allow an interface with the tracking FB, passing to it the index of each conveyor. The code's working principle can be evaluated according to the flowchart presented in Figure B.2.

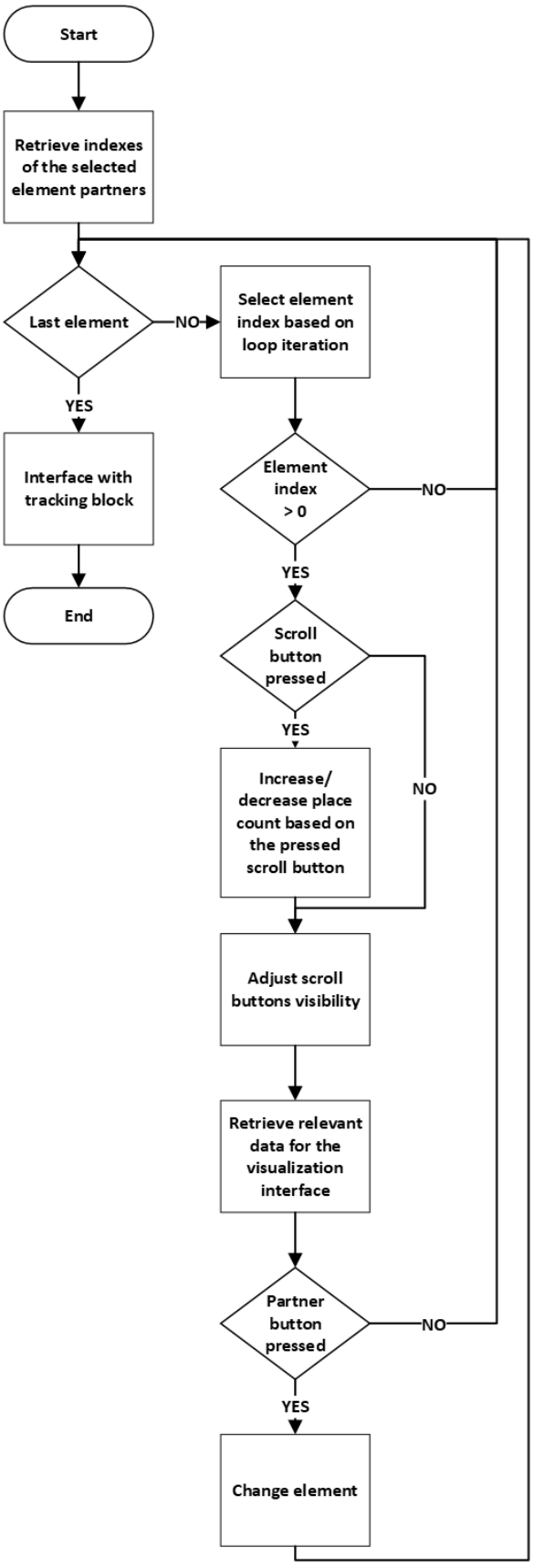


Figure B.2: PLC code for "Partners Data Tracking" on the "FB_-HMI_Panel_UNIFIED" .

In reference to the "FB_HMI_Panel_Tracking_UNIFIED" (Figure B.3), the region dedicated to processing the data for the "Partners Data Tracking" screen starts by initiating a loop instruction to cycle through the functionalities available for the conveyors. Then multiple validations are conducted to ensure that the HMI is allowed to manipulate the data, and the values of the tags used to toggle the visibility of the screen components are also set. After this initial procedure, the function will essentially check if any of the buttons were pressed and execute the desired actions, like copying data between places or exchanging parameters with the buffer based on the selected conveyor.

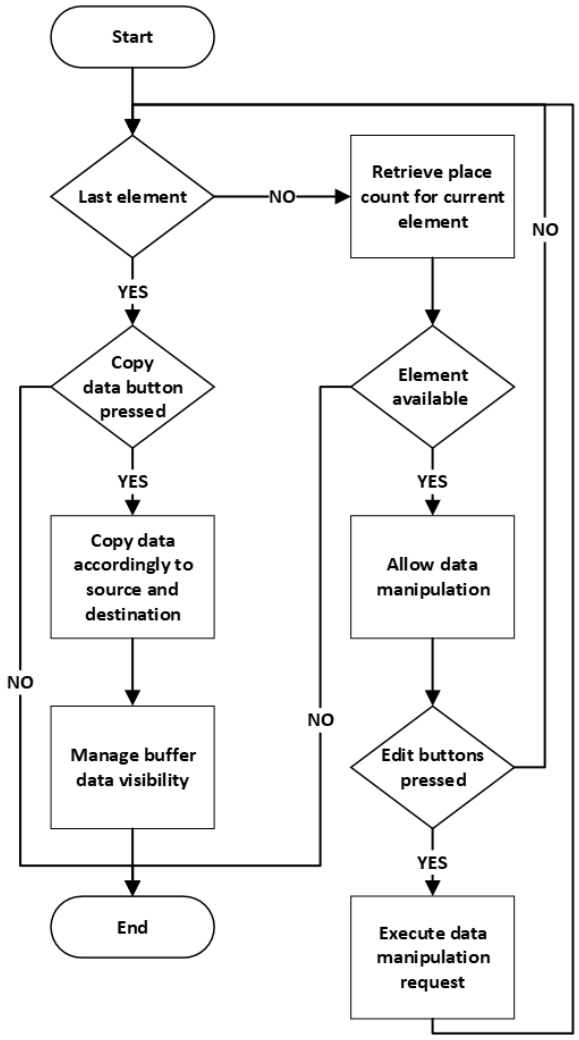


Figure B.3: PLC code for "Partners Data Tracking" on the "FB_-HMI_Panel_TRACKING_UNIFIED" .

B.3.2 Places Data Tracking

Since the "Places Data Tracking" screen is less complex when compared to the previous, the amount of code needed to accomplish its functionalities is considerably

reduced. For this reason, on the "FB_HMI_Panel_UNIFIED", in addition to the initial verification regarding the active state of this screen, a for loop instruction is used to iterate through the places that the selected element supports. For each place, the data regarding its number and position in the tracking database is retrieved. Lastly, the interface with the FB for tracking is executed.

As for the tracking block, the working principle is very similar to the partners tracking, but in this case, the loop cycle is executed for the six places that a conveyor can support. Moreover, since the number of buttons that the screen provides is reduced, the number of validations is also minor.