COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

MÁSTER EN INGENIERÍA INDUSTRIAL + MÁSTER EN INDUSTRIA CONECTADA / SMART INDUSTRY

TRABAJO FIN DE MÁSTER

ICAI Factory Digital Twin

Autor: Gonzalo Carrasco Velilla

Director: José Antonio Rodríguez Mondejar

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

ICAI Factory Digital Twin

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2022/23 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Gonzalo Carrasco Velilla Fecha: 24/07/2023

Gonzalo Carrasco

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: José Antonio Rodríguez Mondejar Fecha: 24/07/2023

Firmado por RODRIGUEZ MONDEJAR JOSE ANTONIO - ***5731** el día 24/07/2023 con un certificado emitido por AC FNMT Usuarios

MÁSTER EN INGENIERÍA INDUSTRIAL + MÁSTER EN INDUSTRIA CONECTADA / SMART INDUSTRY

TRABAJO FIN DE MÁSTER

ICAI Factory Digital Twin

Autor: Gonzalo Carrasco Velilla

Director: José Antonio Rodríguez Mondejar

Madrid

Agradecimientos

En primer lugar, quiero agradecer a José Antonio su trabajo como tutor en este proyecto. Durante el último año, he adquirido conocimientos que han superado con creces todas mis expectativas, y me siento sumamente agradecido por el tiempo y esfuerzo que ha dedicado. Además, su pasión por la ingeniería ha sido verdaderamente contagiosa, inspirándome a seguir formándome en el sector de la automatización industrial.

También quiero dedicar este trabajo a mis padres, que se han sacrificado siempre para apoyarme en mis estudios y que han movido tierra y aire para ayudarme a llegar al sitio en el que me encuentro. Quiero daros las gracias de corazón por haber estado siempre que os he necesitado y por haber sido los mejores padres que nunca habría podido imaginar.

Finalmente, quiero agradecer a Nuria, la persona más importante de mi vida, por su cariño y apoyo incondicional. Con su alegría y amor me ha hecho sentir la persona más afortunada del mundo, ayudándome a superar momentos difíciles y descubriéndome lo que es ser verdaderamente feliz. Quiero compartir mi vida contigo y estoy emocionado por empezar esta nueva etapa en nuestra propia casita. Te amo.

Por todos los buenos momentos que me habéis regalado,
este trabajo es para vosotros.

ICAI FACTORY DIGITAL TWIN

Autor: Carrasco Velilla, Gonzalo.
Director: Rodríguez Mondejar, Antonio.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

Este proyecto tiene como objetivo implementar la arquitectura NOA en la fábrica de ICAI para integrarla en la nueva era conectada, al tiempo que preserva la seguridad e integridad del proceso industrial. La solución propuesta por NOA se basa en la extracción segura de información de los equipos y sensores no intrusivos de la fábrica, consolidando todos los datos en un servidor concentrador OPC UA. Para extraer los datos de manera segura, se diseña un diodo NOA que incluye: clientes OPC UA, sensores de monitoreo y optimización, y un optoacoplador que funciona como diodo de información. Finalmente, el proyecto concluye con el desarrollo de una interfaz SCADA impulsada por NOA, junto con un estudio de ciberseguridad del nuevo sistema. Para la puesta en marcha del proyecto, se elige la pasarela IOT2050 de Siemens.

Palabras clave: NOA, OPC UA, SCADA, pasarela IoT, Diodo Información, Sistema M+O

1. Introducción

La cuarta revolución industrial se encuentra en pleno auge gracias al desarrollo de nuevos avances tecnológicos como los sistemas ciberfísicos, los gemelos digitales o la optimización impulsada por inteligencia artificial. No obstante, la pirámide de automatización, que ha constituido la base de los procesos industriales durante siglos, ha resultado ser inflexible, cerrada e incapaz de adaptarse a las tecnologías emergentes.

Por ello, para abordar estos desafíos, se están desarrollando nuevas arquitecturas que permitan integrar tecnologías propias de la Industria 4.0 en las fábricas. Estas incluyen NOA (Namur Open Architecture) y O-PAS (Open-Process Automation Standard), construidas sobre el estándar de comunicación OPC UA [1]. Por un lado, NOA se centra en modernizar las plantas industriales mediante la implementación de un sistema en paralelo a la pirámide de automatización, en el que integrar las nuevas tecnologías sin interferir con las operaciones de producción [2]. Por el contrario, arquitecturas modulares como O-PAS tienen como objetivo establecer un modelo "abierto" y “modular” que interconecte todos los componentes dentro de las fábricas.

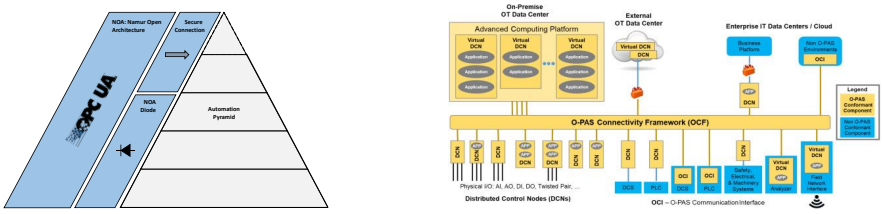


Figura 1. NOA vs O-PAS.

Estas arquitecturas solo son concebibles gracias al estándar OPC UA [3], que desempeña un papel crucial en proyectos de digitalización y es probable que se convierta en el estándar más empleado para el intercambio de información a nivel industrial.

2. Definición del Proyecto

El objetivo de este proyecto es desarrollar una nueva arquitectura para la fábrica de ICAI que supere las limitaciones asociadas a su estructura monolítica. Para minimizar las modificaciones en la planta, se considera que la solución más apropiada es la adopción de la arquitectura NOA, ya que preserva la pirámide de automatización existente, permitiendo una implantación fluida y sin perturbar sus actividades diarias.

En cuanto al alcance, la solución propuesta se basa en la extracción segura de información de los equipos y sensores no intrusivos de la fábrica, consolidando todos los datos en un servidor concentrador OPC UA. Para ello, se diseña un diodo NOA con los componentes necesarios para extraer los datos de manera segura, sin comprometer las operaciones de la fábrica. Estos elementos incluyen clientes OPC UA, sensores de monitoreo y optimización, y un optoacoplador. Finalmente, el proyecto concluye con el desarrollo de una interfaz SCADA impulsada por NOA, junto con un estudio de ciberseguridad de la solución propuesta.

3. Descripción del sistema

El sistema NOA consta de varios componentes esenciales que facilitan la extracción segura de los datos de la fábrica garantizando la fiabilidad y robustez de la pirámide de automatización. En el núcleo del sistema se encuentra el servidor concentrador (OPC UA Aggregate Server), desarrollado en .NET con el SDK de la Fundación OPC, que sirve como repositorio para almacenar, en tiempo real, los datos extraídos de la pirámide.

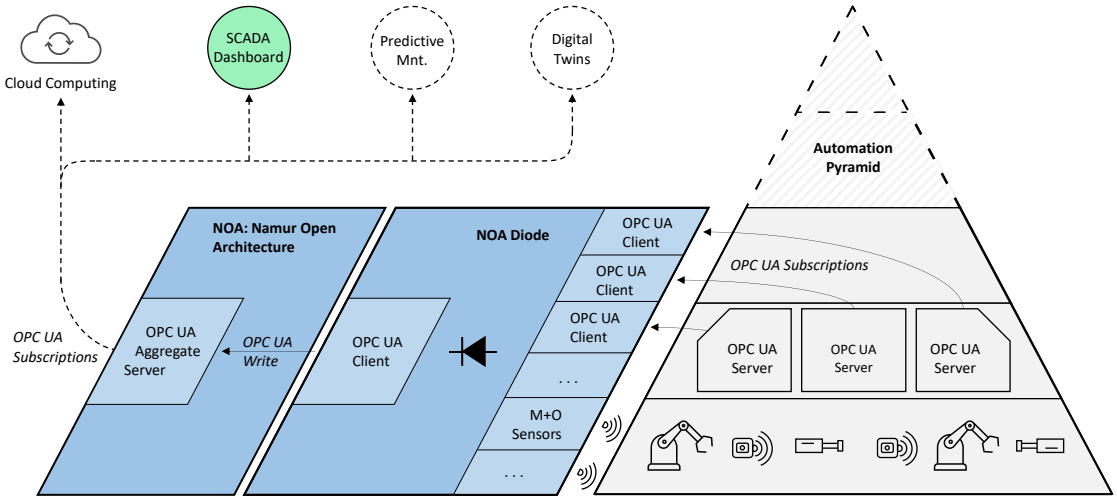


Figura 2. Sistema NOA implementado en la fábrica de ICAI.

Para interactuar con los servidores (OPC UA Server) integrados en los PLCs y los controladores de la fábrica, se crean clientes (OPC UA Client) utilizando Node-RED. Estos clientes utilizan el protocolo MQTT para suscribirse a las variables de interés de los servidores.

Además, el proyecto implementa sensores de Monitorización y Optimización (M+O Sensors), basados en el protocolo Modbus RS-485, para extraer datos adicionales de los

equipos industriales [4]. Estos sensores aportan información valiosa sobre su estado de salud, facilitando el desarrollo de futuros estudios de mantenimiento predictivo.

Desde el punto de vista del hardware, el sistema NOA utiliza una pasarela IOT2050 [5], que aloja todos los clientes OPC UA responsables de recoger los datos de los PLCs, así como el maestro Modbus para los sensores M+O. Además, el proyecto incorpora un diodo de información, basado en comunicación RS-232 y un optoacoplador, para garantizar una comunicación segura y unidireccional.

Finalmente, empleando Node-RED, se desarrolla un cuadro de mandos (SCADA dashboard) para mostrar el potencial de una aplicación real impulsada por NOA. Esta aplicación consiste un cliente OPC UA que se subscribe a los datos recogidos en el servidor concentrador.

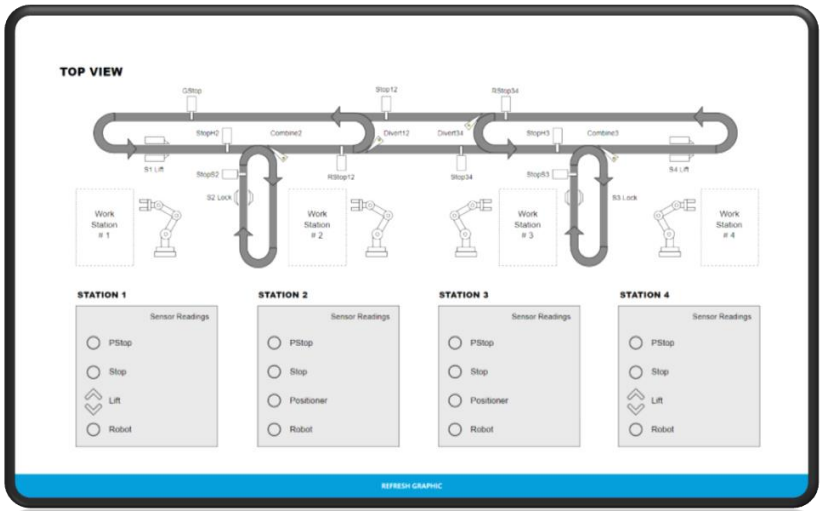


Figura 3. Cuadro de mandos de la fábrica.

4. Resultados

Los resultados del proyecto se materializan una vez se realiza la puesta en marcha del sistema. Por ello, tras varios días de implementación y de exhaustivas pruebas, se pudo confirmar el correcto funcionamiento de los nuevos componentes en la fábrica.

Estos resultados reflejan el éxito del proyecto, demostrando la viabilidad de aplicar la arquitectura NOA en ICAI para la extracción segura de datos. Además, confirman que la solución propuesta por NOA no se limita a un mero concepto teórico, sino que se trata de una arquitectura perfectamente aplicable a entornos industriales.

5. Conclusiones

En conclusión, la implementación de la arquitectura NOA en la fábrica de ICAI ha permitido dar el primer paso hacia su digitalización, sentando las bases para el futuro desarrollo de gemelos digitales, aplicaciones de mantenimiento predictivo, análisis de Big Data y computación en la nube, entre muchos otros.

6. Referencias

[1] Mondejar, J. A. (2023). Automatización Avanzada. ICAI Comillas.

[2] Berge, J. (2019a, november 20). Implementing the NAMUR Open Architecture (NOA). Retrieved from LinkedIn: https://www.linkedin.com/pulse/implementing-namur-open-architecture-noa-jonas-berge

[3] Industry40tv. (2020a, september 2). What is OPC UA and How it Works? Retrieved from YouTube: https://www.youtube.com/watch?v=vRk42W_4R0o&list=PLIrJJXPVFRvjHsA9tta8yULOB8nPUO_G7

[4] Berge, J. (2022c, september 20). M+O Sensors: Monitoring & Optimization. Retrieved from LinkedIn: https://www.linkedin.com/pulse/mo-sensors-monitoring-optimization-jonas-berge

[5] SIEMENS. (2021, october). SIMATIC IOT2050. Retrieved from RS Componentes : https://my.rs-online.com/web/p/iot-gateways/2017732

ICAI FACTORY DIGITAL TWIN

Author: Carrasco Velilla, Gonzalo.
Supervisor: Rodríguez Mondejar, Antonio.
Collaborating Entity: ICAI – Comillas Pontifical University

ABSTRACT

This project covers the implementation of Namur's Open Architecture (NOA) in ICAI's factory to enable its digitalisation and integration into the connected era, preserving its original security and integrity. The solution involves the secure extraction of information from the factory equipment's and non-intrusive sensors, storing and consolidating all data in an OPC UA aggregate server. For its successful implementation, a NOA diode is designed with the necessary components to extract the data securely without compromising the factory's operations. These elements include OPC UA clients, Monitoring and Optimisation sensors, and an optocoupler-based data diode. The project incorporates an IoT gateway as the hardware platform to execute the solution, accompanied by the development of a SCADA dashboard to showcase a real-life application powered by NOA.

Keywords: NOA, OPC UA, SCADA, IoT Gateway, Data Diode, M+O System

1. Introduction

The increasing demand for optimisation and the progress of digital transformation have created a pressing need to modernise the industrial sector. The conventional automation pyramid, backbone of industrial processes for centuries, has failed to adapt to new technological advancements like cyber-physical systems, digital twins, and AI-driven optimisation due to its inflexible and closed structure.

To tackle these challenges, new architectures are being developed to integrate Industry 4.0 technologies into industrial processes. Examples of these architectures include NOA (Namur Open Architecture) and O-PAS (Open-Process Automation Standard), both built upon the OPC UA communication standard [1]. NOA focuses on modernising industrial plants through an information acquisition system, that operates in parallel to production operations, without causing any interference [2]. On the other hand, O-PAS aims to establish an "open" model that interconnects all components within a factory.

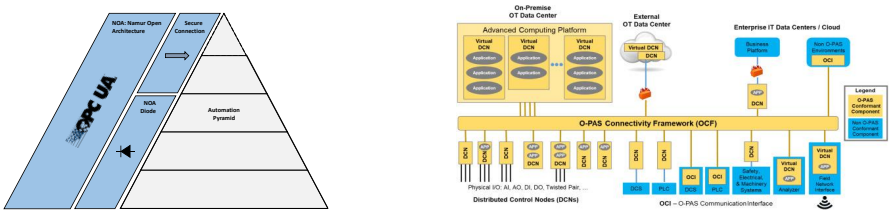


Figure 1. NOA vs O-PAS.

These innovative architectures are made possible by the OPC UA standard [3], which has been essential in digitalisation projects and is expected to become the de-facto standard for exchanging information across the industrial sector, bridging the frontier between OT-IT, and paving the way for interconnected and flexible industries.

2. Project definition

The objective of this project is to develop a new architecture for ICAI's manufacturing plant that overcomes the limitations of its current monolithic approach. To minimise disruption to the plant, a non-intrusive approach is essential. Thus, the adoption of NOA is the most effective solution as it preserves the existing automation pyramid, allowing for seamless implementation without disturbing the plant's day-to-day activities.

The project scope involves the secure extraction of information from the factory equipment's and non-intrusive sensors, storing and consolidating all data in an OPC UA aggregate server. For its successful implementation, a NOA diode is designed with the necessary components to extract the data securely without compromising the factory's operations. These elements include OPC UA clients, Monitoring and Optimisation sensors, and an optocoupler-based data diode. The project incorporates an IoT gateway as the hardware platform to execute the solution, accompanied by the development of a SCADA dashboard to showcase a real-life application powered by NOA.

3. System's description

The NOA system consists of several essential components that facilitate secure data extraction from the factory while ensuring the reliability and robustness of the automation pyramid. At the core of the system is the aggregate server, developed using .NET and the OPC Foundation SDK. This server acts as a centralised hub for collecting and consolidating data from various sources within the manufacturing plant.

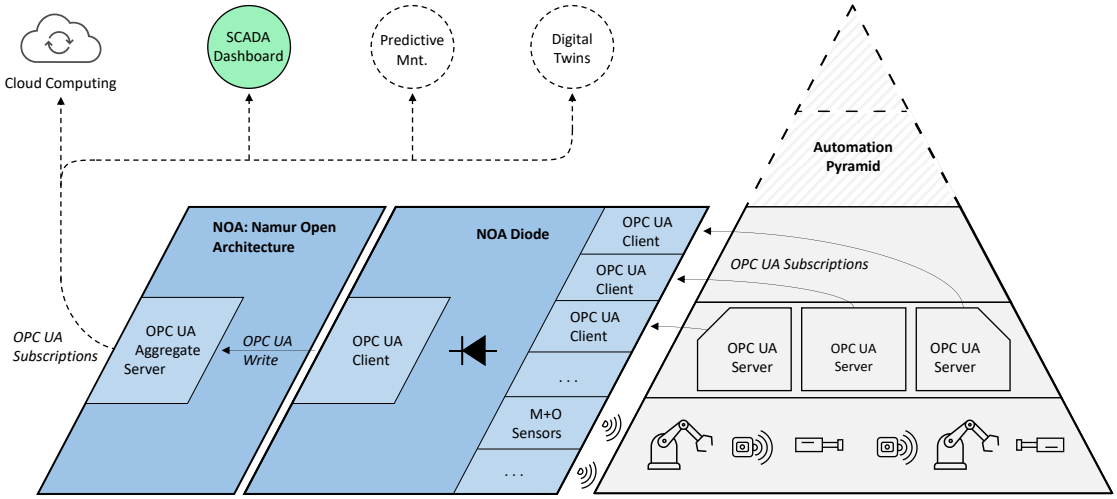


Figure 2. NOA implementation in ICAI's factory.

To interact with the embedded OPC UA servers in the factory's PLCs and controllers, specialised clients are created using Node-RED. These clients utilise the MQTT protocol to subscribe to and retrieve data from the servers, facilitating efficient data exchange between system components.

Additionally, the project implements Monitoring and Optimisation (M+O) sensors to gain access to non-process data [4]. This system provides valuable insights into the condition of equipment and facilitates predictive maintenance studies. In this project, the M+O sensors are installed using an RS-485 communication bus.

From a hardware perspective, NOA's system utilises an IOT2050 gateway [5], which serves as a centralised funnel for directing all data through a single connection point. This device hosts all the clients responsible for retrieving data from the PLCs, as well as the Modbus master for the M+O sensors. Furthermore, the project incorporates a data diode, which employs RS-232 communications and an optocoupler, to ensure a secure one-directional communication.

Additionally, the project covers the development of a SCADA dashboard to showcase a real-life application powered by NOA.

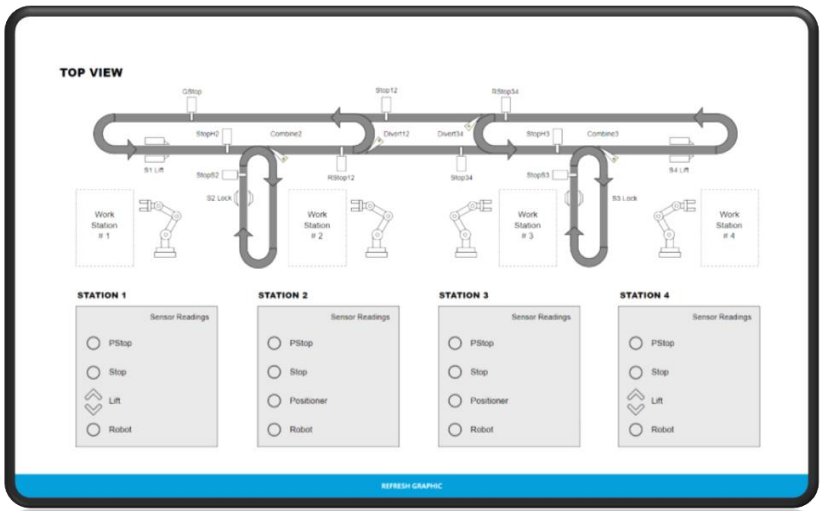


Figure 3. Factory dashboard.

4. Results

Due to the "research-nature" of the project, its results can only be assed after commissioning the entire system in the factory. Thus, over a three-day period, rigorous testing and fine-tuning were conducted to achieve optimal performance of the system, confirming that all components were functioning correctly.

These results demonstrate the success of the project, showcasing the feasibility of implementing NOA in ICAI's factory. Moreover, they confirm that this approach is not only a theoretical concept but a practical solution, perfectly valid for modern industrial facilities.

5. Conclusion

In conclusion, the successful deployment of Namur's Open Architecture in ICAI's factory has enabled its digitalisation and improved the operational capabilities of the plant. This achievement sets the stage for the future development of digital twins, predictive maintenance applications, big data analysis, and cloud computing among many others, driving innovation and competitiveness into the factory's operations.

6. References

[1] Mondejar, J. A. (2023). Automatización Avanzada. ICAI Comillas.

[2] Berge, J. (2019a, november 20). Implementing the NAMUR Open Architecture (NOA). Retrieved from LinkedIn: https://www.linkedin.com/pulse/implementing-namur-open-architecture-noa-jonas-berge

[3] Industry40tv. (2020a, september 2). What is OPC UA and How it Works? Retrieved from YouTube: https://www.youtube.com/watch?v=vRk42W_4R0o&list=PLIrJJXPVFRvjHs A9tta8yULOB8nPUO_G7

[4] Berge, J. (2022c, september 20). M+O Sensors: Monitoring & Optimization. Retrieved from LinkedIn: https://www.linkedin.com/pulse/mo-sensors-monitoring-optimization-jonas-berge

[5] SIEMENS. (2021, october). SIMATIC IOT2050. Retrieved from RS Componentes : https://my.rs-online.com/web/p/iot-gateways/2017732

Index

Figure Index

Table Index

Chapter 1. INTRODUCTION

The rise of digital transformation and the need to create added value from data has become a significant driver for industrial activities world-wide. The traditional approach of industrial automation has prioritised reliability and resilience for factory operations through monolithic and inflexible architectures. However, as companies move further into the era of Industry 4.0, there is a growing need to reimagine industrial automation and move towards more open and scalable structures that can integrate emerging technologies into its operations to enhance efficiency and productivity, while preserving the system's reliability and availability.

In this new era, the automation pyramid, which has been the backbone of industrial processes for years, has failed to accommodate the rapid pace of technological innovation and the implementation of new Industry 4.0 solutions, such as the use of cyber-physical systems, digital twins, optimisation through artificial intelligence, or cloud computing, among many others.

As Information Technology (IT) continues to rapidly advance into the new era of digital transformation, characterised by high-speed communications and massive information transfer, Operation Technologies (OT) are struggling and falling behind, remaining mired in traditional systems. Consequently, the viability of maintaining the current automation pyramid has been called into question in recent years, as its closed and rigid nature delays the development of new OT technologies and fails to capitalise on the huge potential offered by IT breakthroughs.

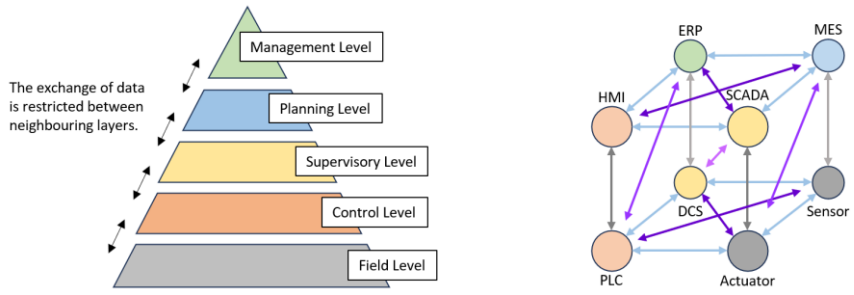


Figure 4. Traditional vs New approach to automation systems. (Represa, et al., 2023)

As a result, ongoing efforts are underway to develop new architectures aimed at maximising the value derived from industrial plant data. These architectures include NOA (Namur Open Architecture), O-PAS (Open-Process Automation Standard), and OPM (Open Manufacturing Platform), along with other innovations in the sector such as the development of MTPs (Modular Type Package) (ABB, 2022).

These architectures are built on top of OPC UA, a secure communication standard developed by the OPC Foundation, which has proven to be crucial in the advancement of industrial digitalisation projects and is likely to become the de-facto standard for IIoT information exchange.

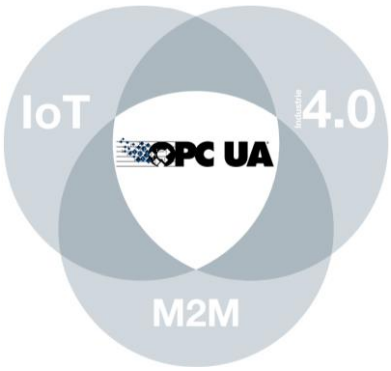


Figure 5. OPC UA role in industry digitalisation.

To grasp the benefits and characteristics of these advanced architectures, the key concepts of each approach are outlined below.

- **NOA (Namur Open Architecture):** NOA focuses on modernising and enhancing industrial plants by implementing an information acquisition system that operates in parallel to production operations without causing any interference. This approach is applicable to both new and existing industrial facilities, emphasising the preservation of essential OT systems while integrating emerging technologies and new market innovations.

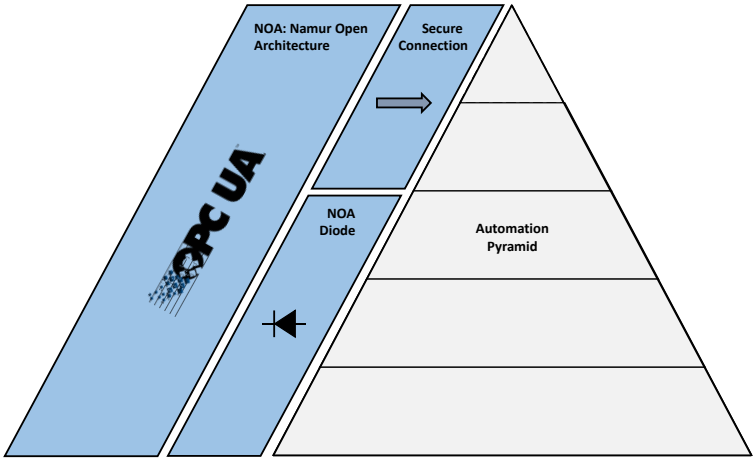


Figure 6. Namur Open Architecture.

- **O-PAS (Open-Process Automation Standard):** O-PAS aims to establish a novel concept of industrial automation departing from the traditional automation pyramid and embracing an "open" model where all components within the factory are interconnected, using the information model provided by OPC UA. This framework fosters a more open, modular, and scalable structure, specifically designed to accommodate the development of new factories.

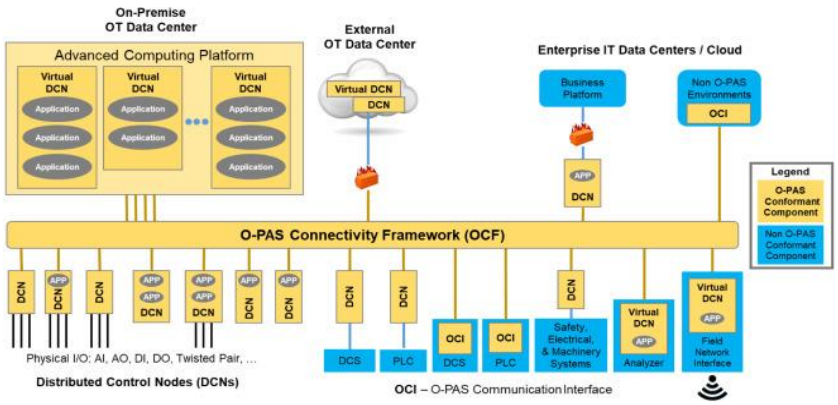


Figure 7. O-PAS Standard.

1.1 PROJECT MOTIVATION

The objective of the present project is to apply these conceptual frameworks to the manufacturing plant of ICAI, developing a new architecture that will alleviate the limitations of its current monolithic approach and serve as a foundational framework for integrating new functionalities associated with digitalisation, such as digital twins and predictive maintenance.

Considering that the facility is already constructed and operational, it is imperative to adopt a non-intrusive approach that guarantees minimal disruption to the plant's regular activities. Consequently, a solution such as NOA, which preserves the existing automation pyramid, is considered the most effective. By adhering to this approach, the original structure remains intact, facilitating a smooth implementation of the project without causing any disturbances to the plant's day-to-day operations.

1.2 PROJECT SCOPE

The project's central objective is to implement the NOA architecture using OPC UA. This entails establishing a robust infrastructure tailored to OPC UA requirements, with the pivotal component being the establishment of an OPC UA aggregate server. This server acts as a unifying platform for storing and consolidating data extracted from the manufacturing plant, enabling seamless accessibility for a multitude of industrial applications aimed at augmenting the factory’s efficiency. Simultaneously, specialised clients must be developed to gather information by subscribing to the OPC UA servers embedded within the factory's PLCs and controllers.

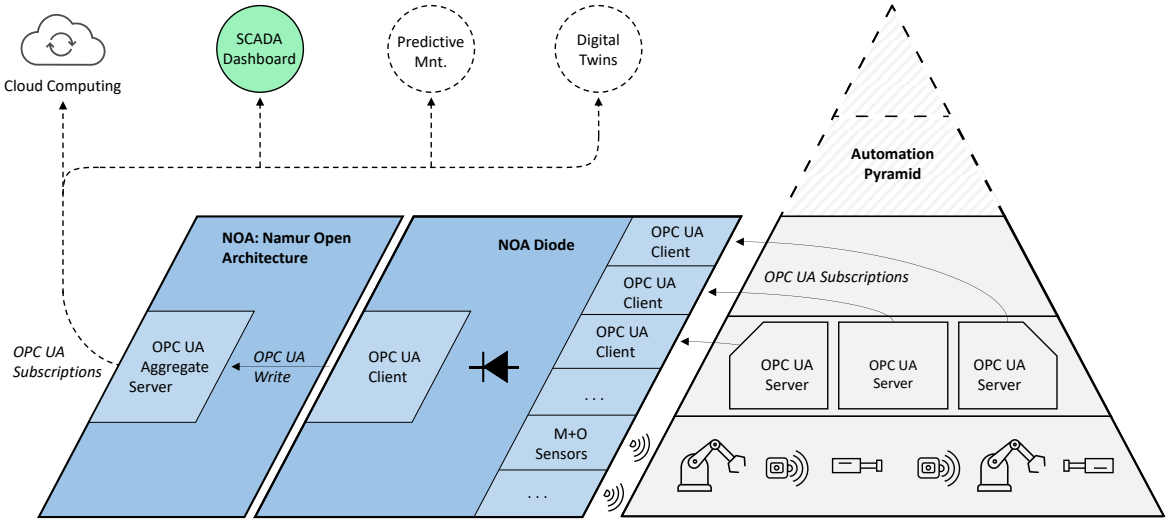


Figure 8. NOA architecture diagram (Original work).

In addition to the OPC UA aggregate server, the project aims to enhance the factory's efficiency and long-term operability through the implementation of Monitoring and Optimisation (M+O) sensors. These sensors monitor non-process data, providing valuable insights into the factory's equipment status and enabling predictive maintenance studies. It is essential to note that the integration of these new sensors must follow the guiding principles of NOA, ensuring their non-intrusiveness to preserve the integrity of the automation pyramid.

To establish the connection between the factory and the external environment, the project incorporates Siemens' IOT2050 as an IoT gateway. This hardware component acts as a bridge, allowing to connect the internal processes operating within the automation pyramid and the OPC UA aggregate server. Consequently, the project's scope includes the configuration of the IOT 2050 device, its deployment, and its setup within the factory.

In terms of security, the IoT gateway works like a funnel, ensuring the secure retrieval of data through a single connection point. This approach facilitates the implementation of robust security measures, including the deployment of a data diode—an essential hardware component that enforces strict one-directional communication. By establishing this controlled connection between the factory and the outside world, the project maximises the benefits of connectivity while effectively preventing unauthorised external access to the industrial system.

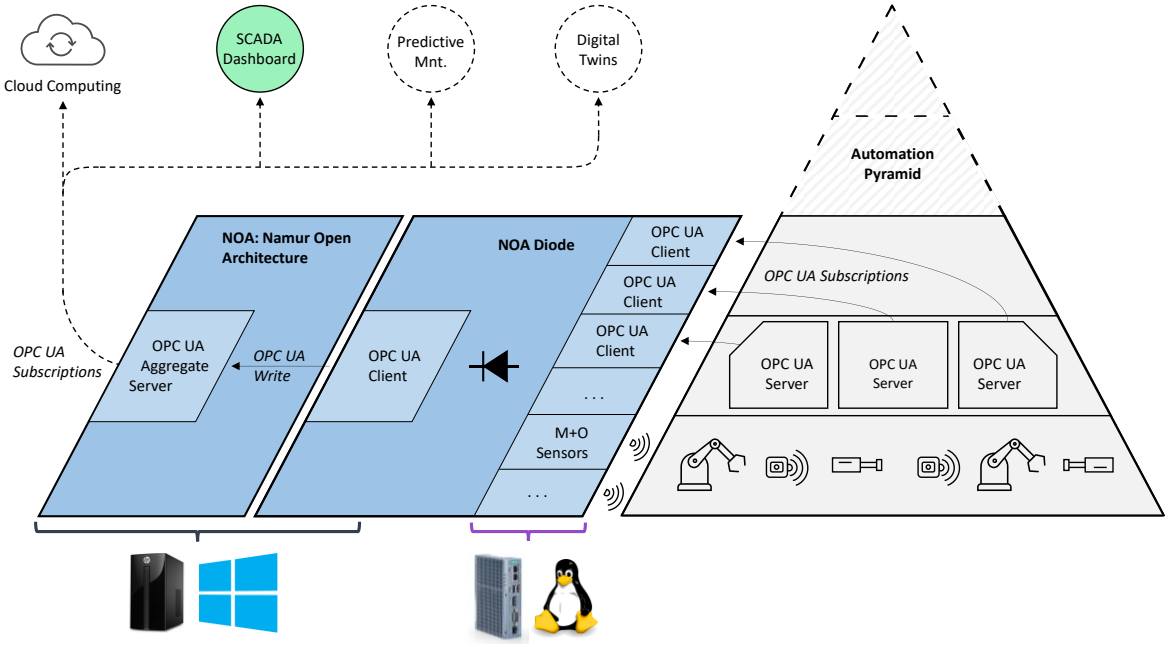


Figure 9. System deployment diagram (Original work).

Lastly, the work concludes with the design of a comprehensive dashboard that empowers operators with monitoring and supervisory capabilities. The dashboard, driven by data from the OPC UA aggregate server, provides operators with a centralised and comprehensive overview of the factory's status, resembling a simplified SCADA system. Hence, it offers a holistic view of operational parameters, enabling informed decision-making and ensuring effective management of the manufacturing plant.

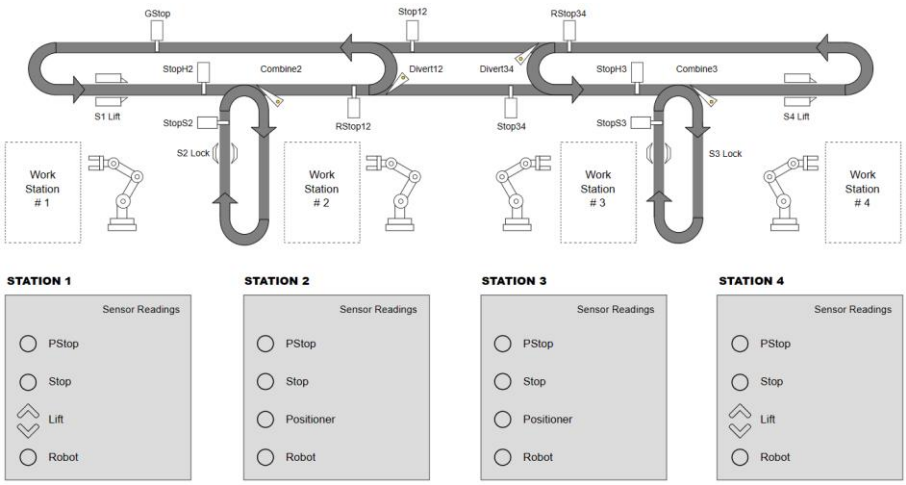


Figure 10. Dashboard with SCADA-like Interface (Original work).

Chapter 2. STATE OF THE ART

This second chapter provides a comprehensive analysis of the current state of the art regarding the application of NOA in industrial environments, with particular emphasis on the utilisation of OPC UA. Furthermore, it thoroughly explores the various applications and uses of predictive maintenance and digital twins within the industry, shedding light on their respective roles and significance. Lastly, the chapter assesses the paramount importance of cybersecurity in modern industrial plants, addressing the challenges and priorities associated with safeguarding these infrastructures.

In recent years, NOA has emerged as a popular approach to enhance data interoperability across different layers in automation. It offers parallel integration, making it possible to develop digitalisation projects in brownfield plants that follow conventional architectures. However, as highlighted in a recent study by (Pontarolli, Bigheti, Sá, & Godoy, 2023), other solutions based on Microservice-Oriented Architectures (MOA), such as O-PAS, are emerging as viable and appealing options for green plants that can be designed from the ground up[1].

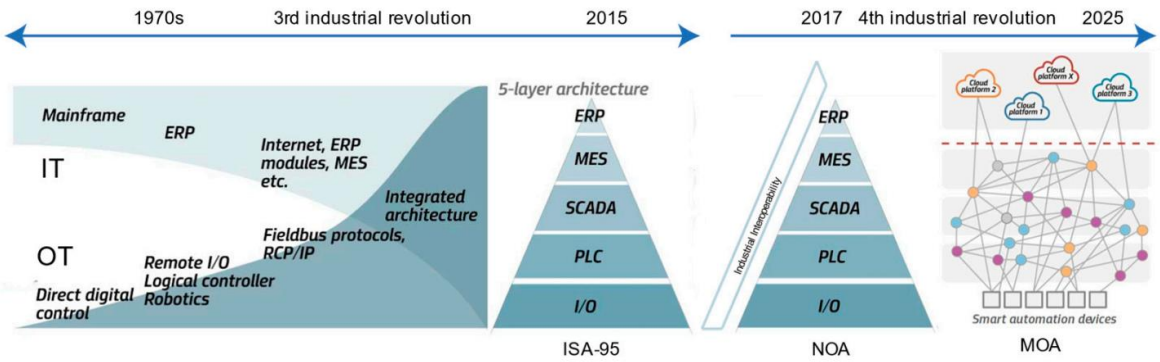


Figure 11. Timeline for IT-OT convergence (Pontarolli, Bigheti, Sá, & Godoy, 2023).

[1] The term "greenfield" typically refers to the development of new projects or facilities, while "brownfield" projects involve the renovation, modification, or expansion of existing industrial plants.

As emphasised earlier, the technological feasibility of these architectures relies on the adoption of an open and standardised communication interface, such as the OPC UA standard. The OPC UA standard is developed and continuously improved by the OPC Foundation, an organisation consisting of numerous influential members. Among these members are industry leaders like ABB, KUKA, Schneider Electric, and Siemens, that advocate for an interconnected industry built on the principles of cooperation and compatibility.

Numerous projects have emerged focusing on the development and design of OPC UA servers and clients, utilising various frameworks and programming languages. However, the OPC Foundation recommends the utilisation of Microsoft's .NET framework, offering their own SDK equipped with essential tools, libraries, and documentation. Although python libraries are also available and are popular for their ease of use (González & BorgesRivero, 2022), their performance limitations often restrict their application to prototyping scenarios.

In addition to NOA and OPC UA, predictive maintenance has become increasingly valuable in proactively identifying, through sensor data collected in Monitoring and Optimisation Systems, the status and health of equipment (Zonta, et al., 2020). Employing cutting-edge technologies like machine learning, artificial intelligence, and data analytics, predictive maintenance enables accurate predictions of asset life and optimisation of maintenance processes, leading to significant cost reductions, minimised downtime, improved productivity, and enhanced product quality (Matyas, Nemeth, Kovacs, & Glawar, 2017).

To understand the scope and technologies employed for asset monitoring in Industry 4.0, the following diagram is provided.

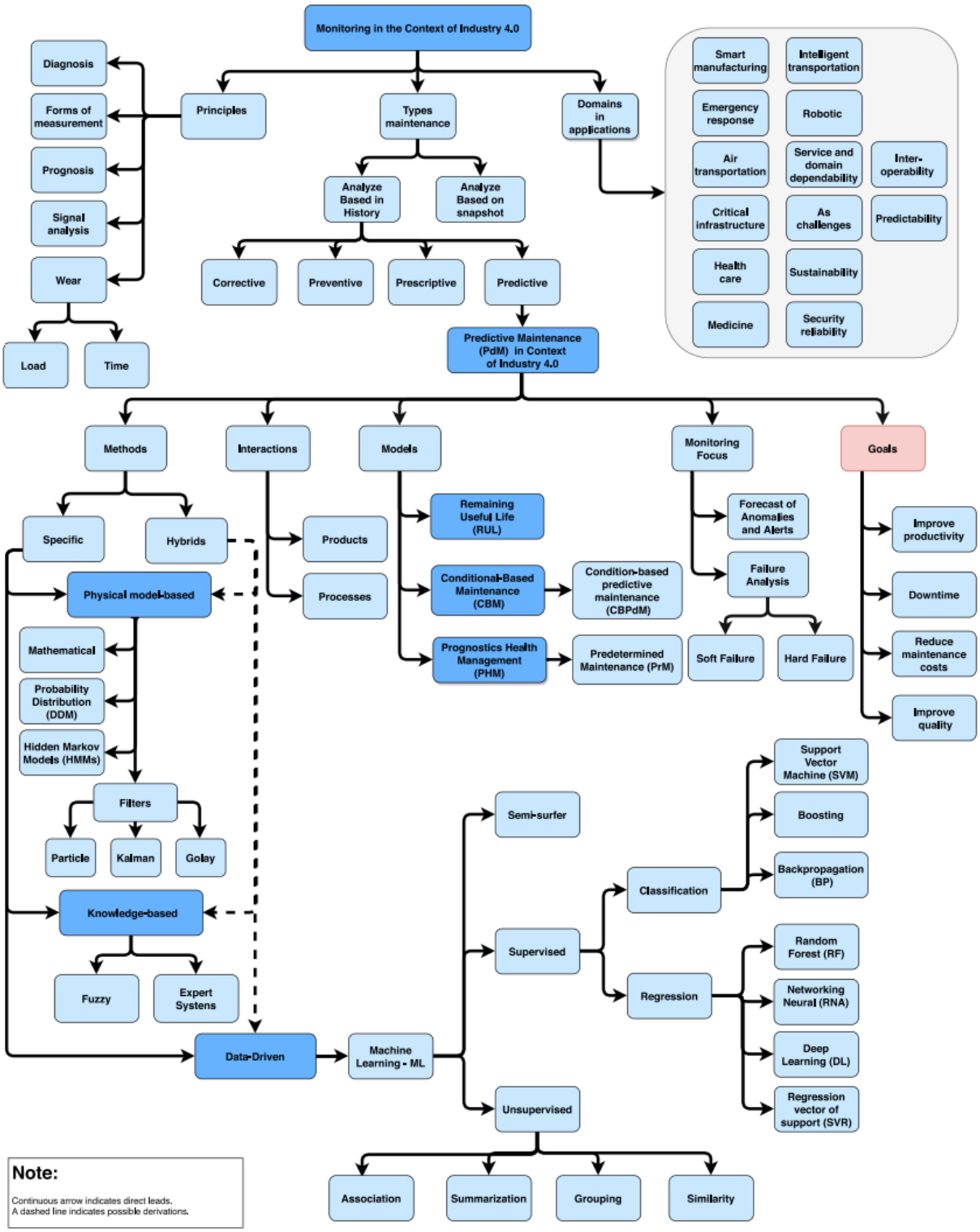


Figure 12. Monitoring in Industry 4.0 (Zonta, et al., 2020).

The use of digital twins in the industry has also been growing rapidly in recent years, with many companies acknowledging the benefits of having virtual replicas of their physical assets (Piromalis & Kantaros, 2022).

In industrial environments, digital twins can play different roles depending on their purpose:

- Product twins: are used to create prototypes of objects to study their functionality and design flaws before the production process starts.
- Process twins: simulate processes to analyse a wide range of scenarios, allowing operators to optimise the factory's parameters to better adapt to the current situation.
- System twins: are virtual replicas of real systems and can be used to gather information, optimise operations, and improve the overall system's performance.

In addition, the application of digital twins encompasses various process functions:

- Design: visualisation processes help verify the 3D design assembly and ensure proper matching and fitting of components.
- Diagnostics: simulations and sensor data analysis reveal non-accessible information, such as forces and stresses applied to different parts of a product or machine.
- Prediction: engineering and deep learning algorithms enable accurate forecasting for equipment longevity and support real-time operation for maintenance planning.
- Maintenance: analyse performance and non-process data to implement predictive maintenance actions.

Finally, the topic of “cybersecurity in the new connected industry” is becoming ever more important. In the past, industrial systems were designed to operate in isolated environments, ensuring that production processes could run securely. However, with the integration of advanced technologies, such as IoT and Big Data analytics, the merging of industrial systems with enterprise networks has become more frequent, resulting in new security threats (Asghar, Hu, & Zeadally, 2019). Thus, to successfully mitigate these risks and safeguard critical infrastructure, the implementation of frameworks, such as NIST, has become crucial to proactively address vulnerabilities and ensure the integrity of entire industrial systems.

Chapter 3. WORK DEFINITION

3.1 PROJECT JUSTIFICATION

The purpose of this chapter is to present a thorough justification for the proposed project, supporting the benefits it brings to its diverse stakeholders. As previously stated, the focal point of this work revolves around enhancing ICAI's manufacturing plant, strategically addressing inherent limitations, and unlocking new opportunities within the organisation.

However, it is important to note that the justification for embarking on this project extends beyond the scope of factory efficiency and advancements. It also encompasses a diverse set of factors that contribute substantial value to ICAI as a University. Therefore, the chapter is divided in two sections that justify the project's ability to enhance and improve both the operations of ICAI's manufacturing plant and the academic landscape of the University.

3.1.1 INDUSTRIAL JUSTIFICATION

Firstly, the project's justification to optimise operations within ICAI's manufacturing plant is presented.

Figure 13. ICAI's manufacturing plant.

3.1.1.1 Operational Excellence

The current monolithic approach in ICAI's manufacturing plant hampers operational efficiency and agility. The project's objective is to develop a new architecture that addresses these limitations and fosters operational excellence. Following NOA's approach and making use of the OPC UA standard, the plant will optimise resource allocation, streamline workflows, and enhance long-term sustainability.

3.1.1.2 Future-Proofing

The fourth industrial revolution is characterised by rapid technological advancements and evolving market demands. To remain adaptable and resilient, ICAI's factory must future-proof its operations. The proposed project provides the factory with the flexibility needed to seamlessly integrate emerging technologies, ensuring that the plant can readily adapt to ground-breaking innovations.

3.1.1.3 Data-Driven Decision-Making

In today's digital era, data-driven decision-making is crucial. The project emphasises the centralisation and consolidation of data through an OPC UA aggregate server, enabling real-time insights into manufacturing processes. By leveraging this data-driven approach, factory workers can make informed decisions, optimise performance, and identify areas for continuous improvement.

3.1.1.4 Security and Compliance

Protecting sensitive data and ensuring compliance with industry regulations are critical in modern manufacturing. The project incorporates robust security measures, such as the deployment of a data diode, to safeguard against unauthorised access and protect industrial systems.

3.1.2 ACADEMIC JUSTIFICATION

This second section highlights the justification for the project's significance in fostering academic excellence and enhancing the academic landscape at ICAI.

3.1.2.1 Cutting-Edge Education

As a leading educational institution, ICAI is committed to provide students with an excellent learning experience. Through this project, students gain hands-on exposure to advanced technologies and industry-relevant practices, equipping them with the necessary skills and knowledge to introduce themselves into the rapidly evolving field of industrial automation. By working with industry-standard tools and protocols, such as OPC UA servers, OPC UA clients, and IoT gateways, students develop practical expertise utilising advanced automation technologies, therefore enhancing their employability and readiness once completed the tuition.

3.1.2.2 Industry-Academia Collaboration

The project creates collaboration opportunities between ICAI and industrial partners. By integrating emerging technologies, ICAI can establish partnerships with industry leaders, enabling students to work on real-world projects and gain practical insights. This collaboration enhances the educational experience, fosters innovation, and strengthens ties with industry leaders.

3.1.2.3 Research and Development

The project provides a platform for research and development activities within ICAI. Faculty members and researchers can explore new methodologies, techniques, and applications related to the fourth industrial revolution. This research not only contributes to the advancement of knowledge but also enhances ICAI's reputation as a hub for cutting-edge research in the field of industrial automation.

In conclusion, the proposed project aligns with the factory's strategic objectives by promoting technological advancement, operational excellence, future-proofing, data-driven decision-making, and security. Furthermore, the project enhances ICAI's reputation as an academic leader, attracting top talent, and establishing the University as a preferred destination for students, researchers, and industry partners in the field of industrial automation.

3.2 OBJECTIVES

After addressing the strategic and market-related aspects of the project, the focus shifts towards the objectives of modernising and adapting ICAI's factory to the new connected era. These objectives are divided into six different phases.

Phase 1: Previous Studies

- Study the available technologies for implementation and select the most suitable ones for the factory.

Phase 2: Development of the NOA Architecture and Information Model

- Redesign the factory using the concepts provided by the NOA architecture, with OPC UA as the main building block.
- Develop a comprehensive information model that accurately represents the factory's elements and hierarchies, providing a clear understanding of its overall operations.

Phase 3: Development of the OPC UA Server and Clients

- Develop the necessary OPC UA servers and clients to communicate and interconnect the system.

Phase 4: Implementation of the Monitoring and Optimisation sensors

- Integrate the Monitoring and Optimisation sensors to extract valuable data for future studies focused on predictive maintenance of industrial hardware.
- Configure the Siemens IOT2050 Gateway to implement the M+O sensors and data diode.

Phase 5: SCADA Dashboard Design

- Design a SCADA dashboard that can graphically represent all data from the factory’s operations through the information model.

Phase 6: Cybersecurity Study and Network Architecture Design

- Conduct an in-depth cybersecurity study of the proposed solution.

3.3 PLANNING AND COST ESTIMATION

The temporal planning is presented, providing a comprehensive overview of all tasks performed and their chronology. Additionally, the budget of the project is analysed considering the costs and resources, both material and human, employed to carry out the work.

3.3.1 PLANNING

Firstly, the Work Breakdown Structure is examined, which encompasses the essential tasks required for the completion of the project. This process involves dividing the work into hierarchical blocks, providing a clear understanding of the project's scope.

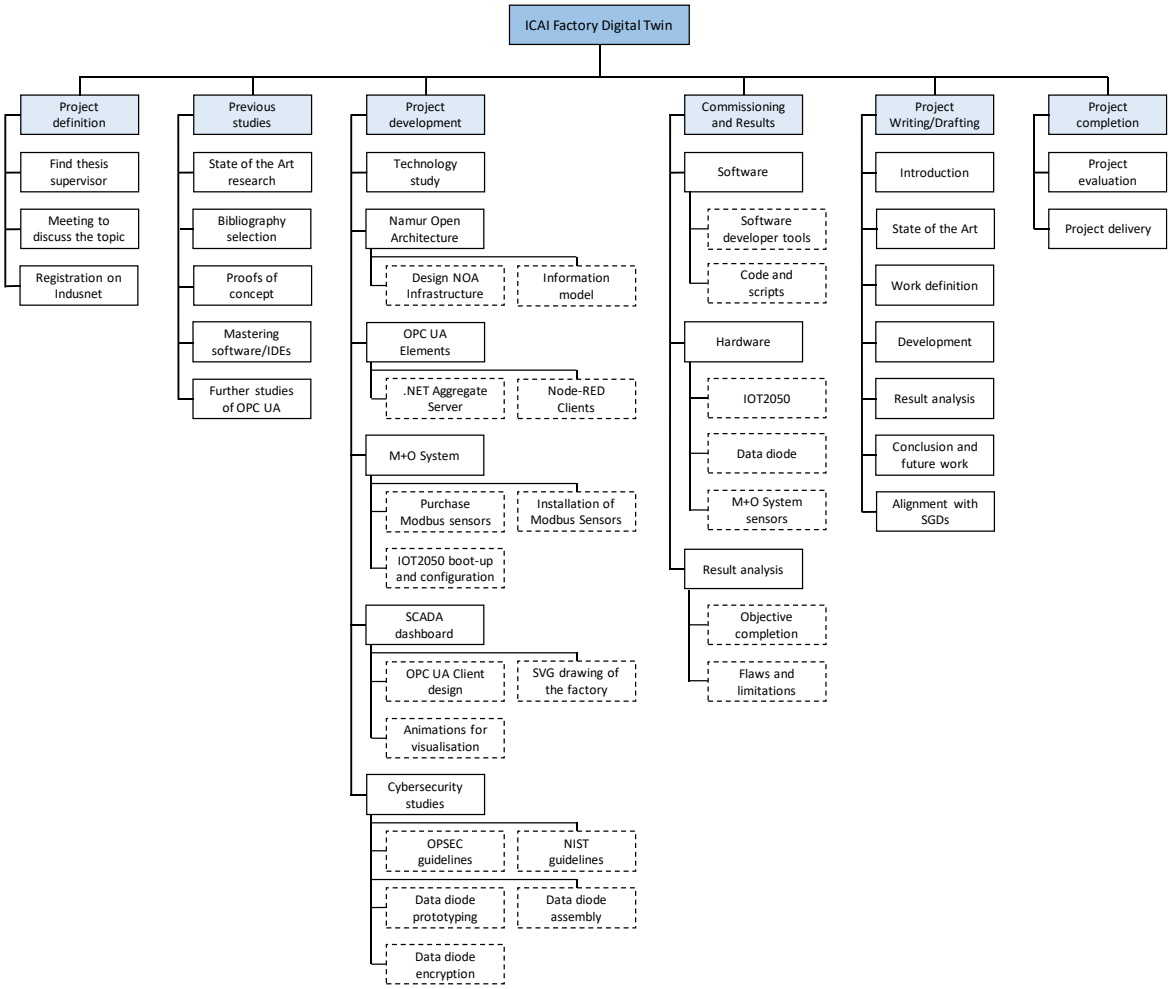


Figure 14. Work Breakdown Structure of the project.

Below is a summary of all tasks performed during the project's execution and their timeframes.

Table 1. Project tasks and timeframes.

Project definition	**15/09/2022**	**16/09/2022**	**15 days**
Find thesis supervisor	01/09/2022	16/09/2022	15 days
Meeting to discuss the topic	13/09/2022	14/09/2022	1 days
Registration on Indusnet	15/09/2022	16/09/2022	1 days
Previous studies	**01/11/2022**	**15/02/2023**	**149 days**
State of the Art research	05/10/2022	31/10/2002	-7279 days
Bibliography selection	19/09/2022	31/10/2022	42 days
Proofs of concept	01/11/2022	15/12/2022	44 days
Mastering software/IDEs	19/09/2022	15/12/2022	87 days
Further studies of OPC UA	19/09/2022	15/02/2023	149 days
Project development	**03/04/2023**	**29/05/2023**	**123 days**
Technology study	01/11/2022	13/01/2023	73 days
Namur Open Architecture	16/01/2023	24/02/2023	39 days
- Design NOA infrastructure	16/01/2023	25/01/2023	9 days
- Information model	26/01/2023	24/02/2023	29 days
OPC UA Elements	26/01/2023	29/05/2023	123 days
- .NET Aggregate Server	26/01/2023	29/05/2023	123 days
- Node-RED Clients	26/01/2023	14/04/2023	78 days
M+O System	01/12/2022	06/03/2023	95 days
- Purchase Modbus sensors	01/12/2022	14/12/2022	13 days
- Installation of Modbus Sensors	25/01/2023	24/02/2023	30 days
- IOT2050 boot-up and configuration	27/02/2023	06/03/2023	7 days
SCADA dashboard	26/01/2023	31/03/2023	64 days
- OPC UA Client design	26/01/2023	03/02/2023	8 days
- SVG drawing of the factory	06/02/2023	15/03/2023	37 days
- Animations for visualisation	16/03/2023	31/03/2023	15 days
Cybersecurity studies	16/02/2023	10/04/2023	53 days
- OPSEC guidelines	16/02/2023	23/02/2023	7 days
- NIST guidelines	16/02/2023	02/03/2023	14 days
- Data diode prototyping	03/03/2023	21/03/2023	18 days
- Data diode assembly	22/03/2023	31/03/2023	9 days
- Data diode encryption	03/04/2023	10/04/2023	7 days
Commissioning and Results	**16/06/2023**	**30/06/2023**	**25 days**
Software	30/05/2023	05/06/2023	6 days
- Software developer tools	30/05/2023	01/06/2023	2 days
- Code and scripts	02/06/2023	05/06/2023	3 days
Hardware	30/05/2023	02/06/2023	3 days
- IOT2050	30/05/2023	02/06/2023	3 days
- Data diode	30/05/2023	02/06/2023	3 days
- M+O System sensors	30/05/2023	02/06/2023	3 days
Result analysis	05/06/2023	30/06/2023	25 days
- Objective completion	05/06/2023	15/06/2023	10 days
- Flaws and limitations	16/06/2023	30/06/2023	14 days
Project Writing/Drafting	**26/06/2023**	**05/07/2023**	**139 days**
Introduction	19/09/2022	14/10/2022	25 days
State of the Art	01/11/2022	11/11/2022	10 days
Work definition	27/02/2023	09/03/2023	10 days
Development	16/02/2023	05/07/2023	139 days
Result analysis	16/06/2023	23/06/2023	7 days
Conclusion and future work	26/06/2023	30/06/2023	4 days
Alignment with SGDs	15/02/2023	15/03/2023	28 days
Project completion	**10/07/2023**	**14/07/2023**	**4 days**
Project evaluation	03/07/2023	07/07/2023	4 days
Project delivery	10/07/2023	14/07/2023	4 days

With the data provided in Table 1, a Gantt chart is constructed to offer graphical and immediate visualisation of the work's planning.

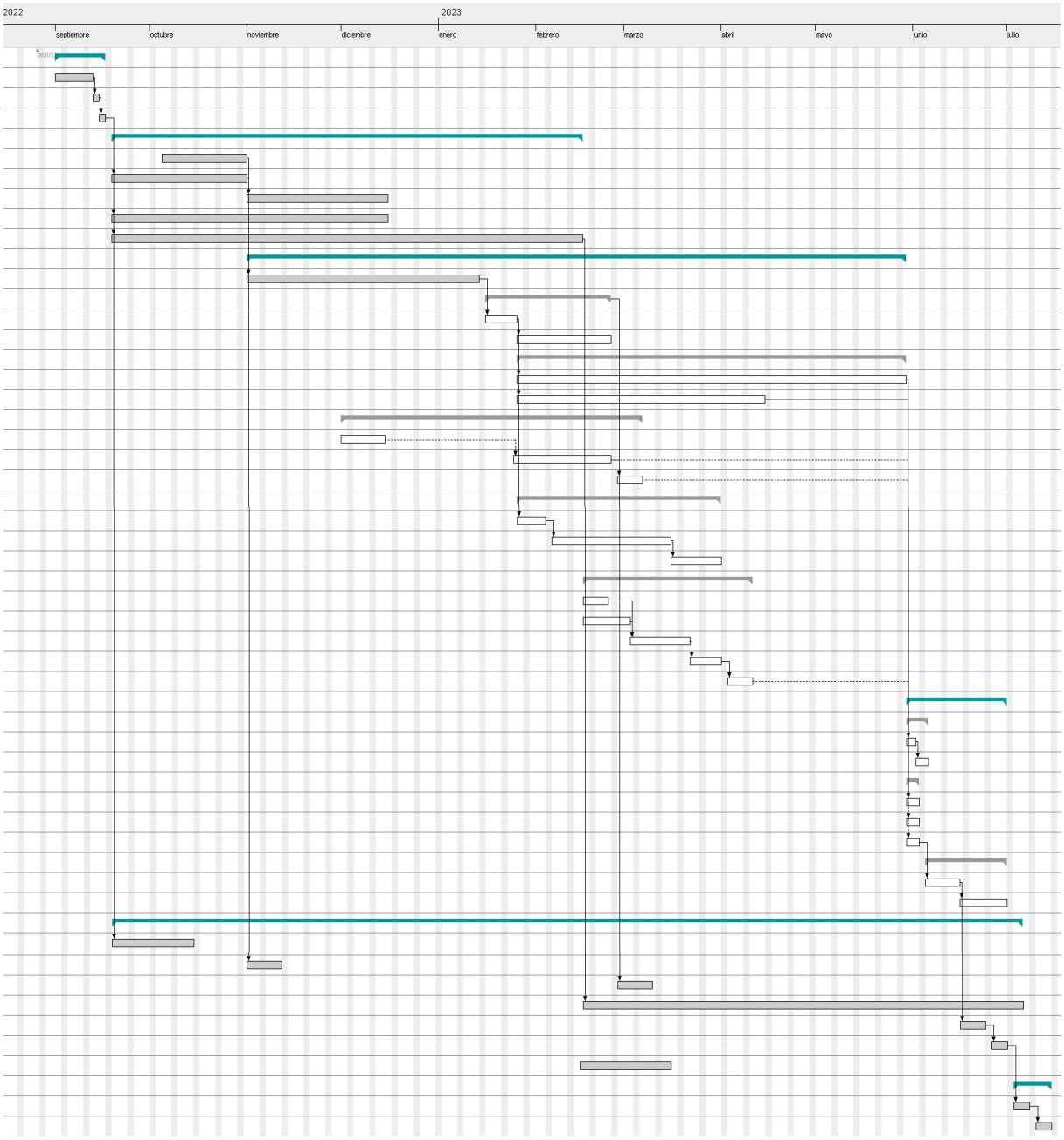


Figure 15. Gantt diagram for the project.

3.3.2 COST ESTIMATION

Finally, the required resources for the thesis implementation are presented, which are categorised into material and human resources:

- The material resources are associated with the hardware required to implement Namur's Open Architecture in ICAI's factory.

Table 2. Material cost of the project.

Equipment	Units	Individual Cost	Total Cost
Gateway IOT2050 Advanced	1	455,12 €	455,12 €
IOT2050: Electrical box	1	15,00 €	15,00 €
IOT2050: 32GB MicroSD card	1	6,90 €	6,90 €
MicroSD to USB adapter	1	8,29 €	8,29 €
Data diode: 4N35 Optocoupler	1	0,65 €	0,65 €
Data diode: Printed Circuit Board	1	0,31 €	0,31 €
Data diode: Components	-	1,50 €	1,50 €
Data diode: DIN case	1	1,66 €	1,66 €
RS-232 to USB convertor	2	1,01 €	2,02 €
RS-485 to USB convertor	1	3,87 €	3,87 €
Type A USB cable	1	1,60 €	1,60 €
Power supply connector	1	5,99 €	5,99 €
Power supply cable	1	7,79 €	7,79 €
Ethernet connector	1	7,35 €	7,35 €
Ethernet cable	2	3,80 €	7,60 €
USB flexible connector	2	4,37 €	8,74 €
Wires	-	2-4 €	4,00 €
XY-MD02 temperature and humidity sensor	2	5,13 €	10,26 €
UART serial converter	1	7,00 €	7,00 €
Cable connectors for M+O bus	2	1,05 €	2,10 €
TOTAL COST			**557,75 €**

- Additionally, costs associated to human resources include the salary of a junior engineer, for the development of the project, as well as the salary of a senior engineer, to supervise the work's progress.

Table 3. Labor cost of the project.

Employee	Hours	Cost per hour	Total Cost
Junior engineer	400 h	18 €/h	7200 €
Senior engineer	20 h	40 €/h	800 €
TOTAL COST			**8000 €**

Therefore, the total expenditure for the project amounts to **8557,75 €**.

Chapter 4. TECHNOLOGY DESCRIPTION

To grasp and understand the execution of the project, an overview of the most important technologies is undertaken. Thus, the primary objective of this chapter is to highlight the essential features of each technology and explain the selection process used to identify the most suitable ones.

The chapter starts by providing a detailed overview of the OPC UA standard, followed by an exploration of development tools and approaches that can be used to create the OPC UA aggregate server and clients. Considering the widespread adoption of the OPC UA standard and the various approaches that can be taken, a comprehensive analysis is conducted to determine the optimal solution for the project.

The chapter concludes with an overview of other technologies involved in the new factory's architecture, such as the Siemens' IOT2050 Gateway, as well as the communication protocols used for the M+O sensors (RS-485) and data diode (RS-232).

4.1 OPC UNIFIED ARCHITECTURE

4.1.1 OPC ORIGINS

OPC Unified Architecture (OPC UA) is a standard developed by the OPC Foundation, building on the legacy of Classic OPC. The goal of the first standard was to standardise the information flow from the process level to the management level (SCADA, MES, ERP, etc.), making it easier to integrate HMIs and SCADA systems and provide "real-time", "historical" and "event" data to management applications.

To achieve this, Classic OPC has three specifications: Data Access (DA), Alarm and Events (A&E), and Historical Data Access (HDA). The system uses a client-server approach, where servers store data from different sources of information, and clients access and consume the data.

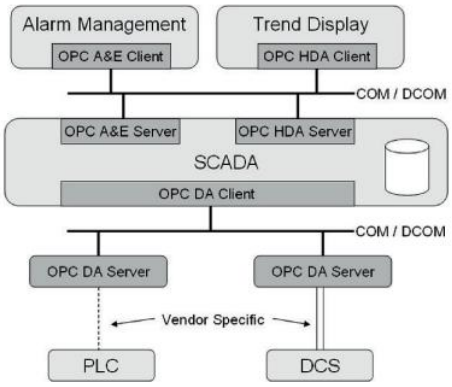


Figure 16. Typical example of clients and servers in Classic OPC (Mahnke, Leitner, & Damm, 2009).

The Classic OPC interfaces were originally constructed using COM (Component Object Model) and DCOM (Distributed COM) technologies from Windows, enabling the standard to have a rapid time-to-market as there was no need for the development of new network protocols or communication mechanisms. This initial approach was critical for the success of Classic OPC, but it also imposed important limitations on the standard.

Some of the main disadvantages of this first standard, in relation with its communication mechanism, include:

- Limited cross-platform compatibility: Classic OPC relies on Windows platform protocols (COM and DCOM), which restrict its usage to Windows-based computers and servers. This limitation hampers its practical applications, as it cannot be utilised in embedded devices or servers that predominantly operate on Linux operating systems. As a result, its flexibility and scope of use is significantly constrained.

- Compatibility issues with modern network environments: DCOM communications are not compatible with network environments containing Network Address Translation (NAT) or requiring communication through firewalls. This poses challenges, as it requires the opening of DCOM ports, which deviates from established IT standards and best practices.

- Inability to facilitate Internet communication: The COM and DCOM communications employed in Classic OPC lack support for communication over the Internet. This limitation hinders the standard's adaptability to new Internet of Things (IoT) solutions and connected industry scenarios, where communication and interoperability across diverse networks is crucial.

Additionally, Classic OPC suffers from limitations in its information modelling capabilities. This becomes apparent when analysing the data transmitted through OPC DA, A&E, and HDA, as it is confined to basic pieces of information that lack comprehensive details regarding the overall condition of the factory.

For instance, when utilising Classic OPC, a temperature sensor reading is communicated without accompanying information about the sensor's type, precision, or engineering units (e.g., Celsius, Fahrenheit, or Kelvin). Consequently, the transmitted information lacks the crucial context and features necessary for achieving seamless interconnectivity across the organisation's infrastructure. This restriction primarily confines the standard's functionality to basic applications such as SCADAs and HMIs, limiting its potential for more advanced use cases.

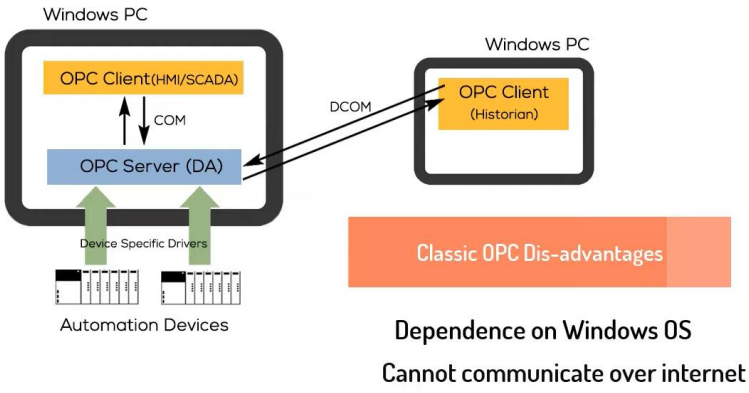


Figure 17. Main flaws of Classic OPC (Industry40tv, 2020a).

In response to these challenges, the OPC Unified Architecture was developed to create a platform-independent standard that could replace all COM-based specifications while maintaining the performance and features that made Classic OPC successful. This new standard was designed to enhance the reliability of its transport mechanism, enable communication over the Internet, and incorporate crucial security and access control features. Additionally, the new standard addressed the growing complexity of manufacturing systems through a significant improvement of its information models, achieving greater scalability, processing more intricate data, and enabling interoperability between manufacturers.

The requirements that OPC UA is designed to address are summarised in the following table.

Table 4. Improvements brought through OPC UA.

Communication between distributed system	Modelling Data
Reliability by: ▪ Robustness and fault tolerance ▪ Redundancy	Common model for all OPC data
	Object-oriented
	Extensible type system
Platform-independence	Meta information
Scalability	Complex data and methods
High performance	Scalability from simple to complex models
Internet and firewalls	Abstract base model
Security and access control	Base for other standard data models
Interoperability	

4.1.2 OPC UA STANDARD

The OPC UA standard comprises several layers that build upon two key components: the transport mechanism and the newly introduced data modelling. This chapter aims to provide a comprehensive understanding of OPC UA by first describing its layers and then exploring its two fundamental pillars.

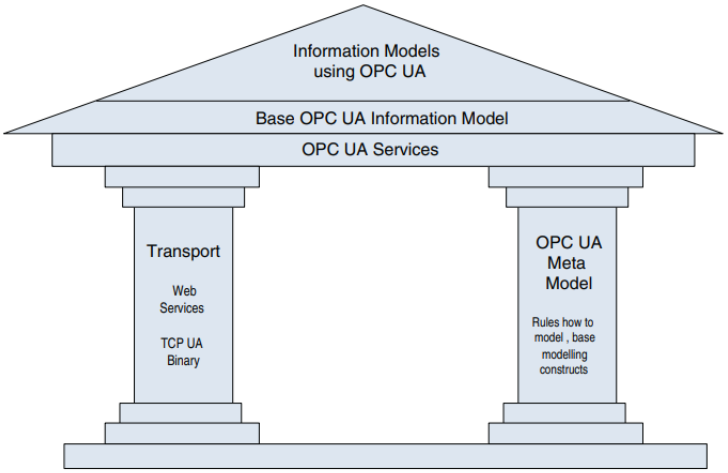


Figure 18. OPC UA Foundation (Mahnke, Leitner, & Damm, 2009).

4.1.2.1 Layer 1. OPC UA Services:

The bottom layer comprises the *OPC UA Services*, which are responsible for establishing the service-oriented architecture within the standard. By doing so, OPC UA servers can provide services that enable clients to access data rapidly and effectively, reducing the overall complexity of the data exchange process while improving its readability, reusability, and maintainability. Consequently, this architecture streamlines communications, adopting a higher degree of flexibility when compared to other industrial protocols[2].

[2] It is important to understand that OPC Unified Architecture is a communication standard. The protocols used by OPC UA are HTTP, TCP, or MQTT (for managing subscriptions), and when used in combination with the layers offered by OPC UA, they can significantly enhance communication in industrial systems compared to traditional protocols like Modbus.

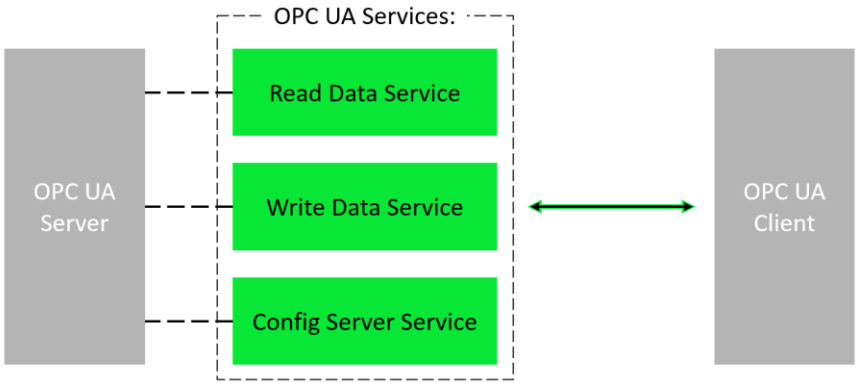


Figure 19. Services offered by OPC UA servers (Original work).

4.1.2.2 Layer 2. Base OPC Information models:

As information models must be consumed by various applications, the new standard includes consistent rules around how they must be built and presented, ensuring that all systems using OPC UA can understand them. Thus, this layer defines the basic modelling blocks from which more complex models can be constructed, ensuring consistency between different applications[3].

4.1.2.3 Layer 3. Information models using OPC UA:

The third layer encompass specific information models built on the OPC UA framework. These models define the structured format of data exchanged between servers and clients and they offer the flexibility to be customised and tailored to meet specific use cases and requirements.

4.1.2.4 Transportation mechanisms:

The transport mechanism plays a critical role in OPC UA by defining the protocols that establish connections between all elements within the framework. Its responsibility lies in ensuring secure and reliable communication, facilitating data exchange, and promoting interoperability across servers and clients.

[3] This is explained in more detail in chapter 5.1.2, which covers the development of the information model for ICAI's factory.

The OPC UA standard introduces various transport mechanism for data transfer:

- Firewall-friendly Web Services using SOAP and HTTP
- And an optimised binary TCP protocol designed to facilitate high-performance communications.

By leveraging these transport options, OPC UA enables more efficient and secure data exchange across industrial networks. The Web Services transport is well-suited for scenarios where firewalls are present, as it employs ports 80 and 443 respectively, allowing data transfer without requiring complex configuration changes or network setups.

On the other hand, the binary TCP protocol offers high-speed data transfer for use cases that require low-latency and real-time performance. By offering these flexible transport options, OPC UA can support a variety of industrial applications and use cases, making it a highly valuable standard for modern industrial communications.

Figure 20. OPC UA transport mechanisms (Inmation, 2021).

The platform-independent protocols of OPC UA enable it to operate on a vast range of hardware, software platforms and operating systems, making it incredibly adaptable and compatible with various industrial settings. This flexibility allows OPC UA to run on low-performance IoT devices, high-end servers, PLCs, DCS or SCADAS, while being easily integrated with enterprise-level software such as MES and ERP systems.

4.1.2.5 Data modelling:

With regards to the data modelling component, OPC UA adopts an object-oriented approach to create complex information models, enabling the accurate representation of intricate hierarchies and components. This foundation becomes particularly valuable in the context of conventional industrial environments that rely on the automation pyramid.

In these environments, communication is typically limited to adjacent layers due to the isolation imposed by distinct communication networks, such as Profinet, Fieldbus, or Modbus in the lower layers, Controlbus in the intermediate layer, and enterprise networks in the top two layers. However, the advent of IIoT standards like OPC UA has revolutionised the communication landscape eliminating these network boundaries and fostering a decentralised communication system that transcends traditional layer-based constraints.

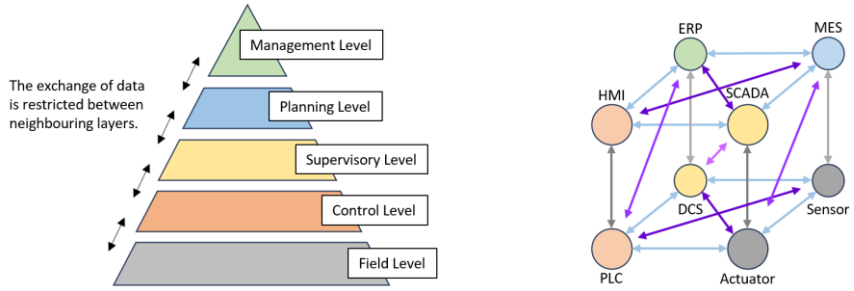


Figure 21. Traditional vs Decentralised communications (Represa, et al., 2023).

To achieve effective communication in these decentralised systems, it is crucial to ensure consistent and standardised data exchange. This entails representing information in a uniform manner across all five levels, from top-level enterprise software like ERP programs to basic elements like field-level sensors.

Thus, the objective of this information model is to provide every element in the factory's environment with a comprehensive view of all other components, understanding not only the basic data (temperature in case of a field-level temperature sensor) but also its context and situation within the factory (Industry40tv, 2020b).

4.2 DEVELOPMENT TOOLS

Once the OPC UA Standard has been thoroughly covered, the next phase consists of assessing and selecting the most suitable software tools for creating and designing the various components needed for the project. This encompasses the tools needed for designing the information models and building the OPC UA servers and clients.

4.2.1 DEVELOPMENT OF THE INFORMATION MODEL

This first section focuses on the development of the information model in compliance with the standards set by the OPC Foundation.

The OPC Foundation mandates that information models must be defined and packaged in an XML format that can be easily imported into any OPC UA application software. This XML file, referred to as a NodeSet2 XML document, can be created using one of three methods (Industry40tv, 2020c):

- Directly writing the NodeSet2 XML file.
- Designing a simple XML file and compiling it using the UAModelCompiler provided by the OPC Foundation.
- Using graphical tools provided by companies.

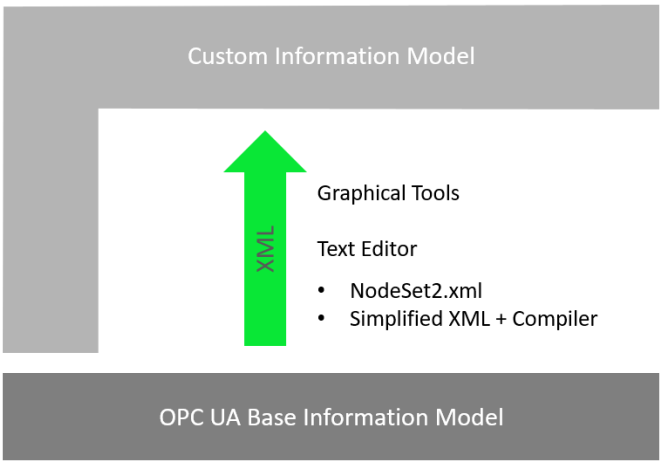


Figure 22. Methods to build a custom information model (Original work).

The first choice is not recommended because of its complexity and the high risk of errors, and, while the graphical option appears to be the logical route, the available tools are currently not sophisticated enough to construct intricate information models. As a result, the sole feasible alternative is to create a simplified XML file and compile it using the free compiler. This approach is also the one recommended by the foundation, as they provide a helpful XSD editor setup that aids in the creation of the XML schema[4].

4.2.2 DEVELOPMENT OF THE SERVER

Regarding the development of OPC UA servers, engineers have multiple options at their disposal. Various OPC UA SDKs are available in different programming languages, providing a solid foundation for server development. These SDKs offer pre-built components and libraries that handle the complexities of the OPC UA communication stack, security, and other essential functionalities.

For instance, the Unified Automation SDK, Prosys OPC UA SDK, and the OPC UA .NET Standard Stack are popular choices that provide comprehensive toolsets and APIs for OPC UA server development. These SDKs offer features like secure communication, node management, historical data access, event handling, and more. They enable developers to focus on implementing business logic while abstracting the low-level protocol details, thereby accelerating the server development process.

Figure 23. Available SDKs for server development.

Moreover, Python and Node-RED provide alternative options for developing OPC UA servers, offering simplicity and ease of use. Python's opcua-asyncio library and Node-RED's OPC UA

[4] The XSD file provides a set of rules that can be imported into Visual Studio to work with XML files, offering suggestions on which commands to use and highlighting any errors made.

nodes allow developers to create servers with customised functionality using their respective ecosystems.

However, in the context of this project, the chosen method for OPC UA server development is utilising the OPC Foundation's recommended approach. This involves using the OPC Foundation's SDK in combination with the .NET framework. The foundation's SDK provides robust support for OPC UA server development, ensuring compliance with OPC UA standards and facilitating interoperability with other OPC UA clients and servers.

By following the OPC Foundation's recommended approach, the project benefits from a proven and widely adopted development path. The SDK's pre-built components and libraries handle critical aspects such as secure communication, session management, and data access.

For the .NET framework, the project will use the latest long-term support version, .NET 6.0. Previous frameworks for OPC UA server development have been found vulnerable to cyberattacks according to ICS-CERT Advisories. However, the topic of cybersecurity will be covered in detail in Chapter 6.

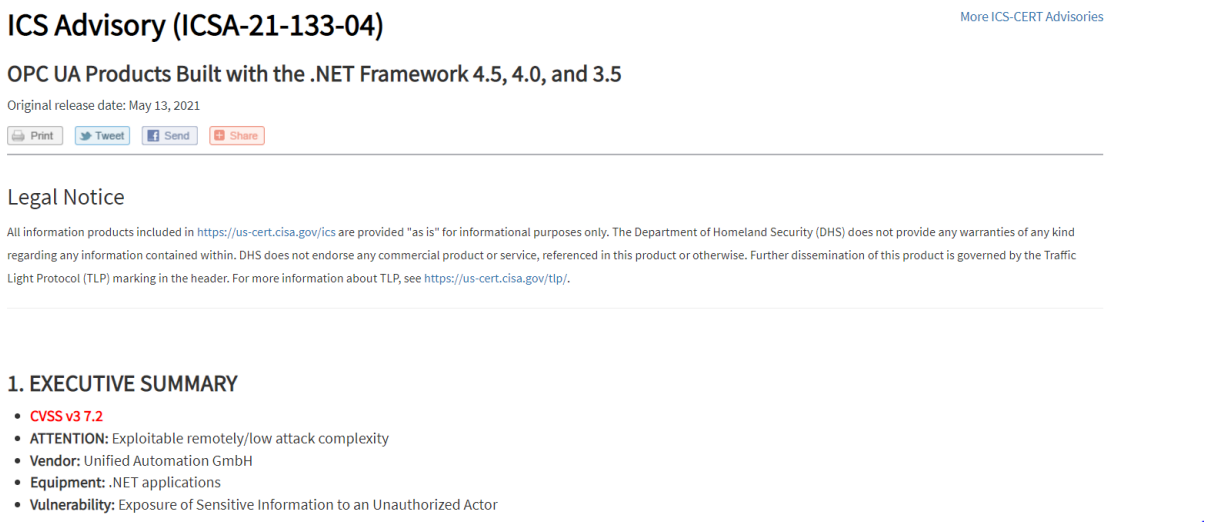


Figure 24. .NET frameworks vulnerabilities.

4.2.3 DEVELOPMENT OF THE CLIENTS

Developing OPC UA clients also requires careful consideration of the tools and frameworks available. OPC UA SDKs, similar to server development, provide a solid foundation for client development, offering various features and functionalities. However, there are some notable differences and considerations when compared to server development.

While the same OPC UA SDKs and programming languages can be used for both servers and clients, the focus and requirements differ. OPC UA servers need to be more robust and secure to mitigate risks such as DDoS attacks and unauthorised access. Therefore, the use of the OPC Foundation's SDK is often considered mandatory for server development due to its security features and adherence to OPC UA standards.

In contrast, OPC UA client development can benefit from alternative approaches. For this project, the chosen software for developing OPC UA clients is Node-RED. Node-RED is an open-source visual programming tool that offers a wide range of pre-built nodes and an intuitive interface for creating workflows. The decision to use Node-RED as the client development platform is based on its ease of use and several benefits it provides:

1. Intuitive Flow-based Programming: Node-RED employs a visual programming (low-code) approach that simplifies the creation of OPC UA client workflows, eliminating the need for extensive coding knowledge.

2. Ease of Modification: Its visual interface enables effortless and seamless modifications to the OPC UA client, promoting iterative developments and rapid prototyping.

3. Maintenance and Debugging: Node-RED's clear and intuitive workflows enable efficient maintenance throughout the lifespan of the application.

4. Dashboard Development: Node-RED offers built-in capabilities for creating interactive dashboards, feature that is particularly advantageous for the project as it allows for the development of a SCADA-like interface.

4.3 HARDWARE COMPONENTS AND COMMUNICATION PROTOCOLS

Finally, this last section provides an overview of the hardware components and communication protocols employed in the project. In this case, the hardware equipment revolves around the use of an IoT gateway, which plays a crucial role in connecting the factory's automation pyramid with NOA's acquisition system that operates in parallel.

4.3.1 IOT2050 GATEWAY

Siemens' IOT2050 is selected as the IoT gateway for this project. It is a compact and robust industrial gateway specifically designed for IoT applications that offers reliable connectivity, processing power, and advanced security.

The IOT2050 device features a wide array of ports and supports various communication protocols, allowing for comprehensive visibility into the factory's information landscape. This capability enables the integration of both modern digitalisation mechanisms and traditional protocols, bridging the gap between different technologies and facilitating smooth communication and data exchange within the factory's environment.

Some of the most relevant specifications of the IoT gateway are listed below, although the full specifications are displayed in Annex V. IOT2050.

Table 5. IOT2050 specifications (SIEMENS, 2021).

Attribute	Value
Communication Protocol	Ethernet
Number of COM Ports	1
COM Port Type	1x RS 232 / 422 / 485
On Board I/O	x20 Digital I/O, x6 Analog Inputs
Extension	Arduino, mPCIe

4.3.2 MONITORING AND OPTIMISATION SENSORS

The IoT gateway's versatility allows it to support a wide array of sensors with various communication protocols, empowering the Monitoring and Optimisation (M+O) system to gather valuable insights from the factory floor. However, while the gateway is capable of interfacing with multiple industrial protocols, the project only focuses on the utilisation of Modbus sensors.

Modbus RS-485 is a widely adopted serial communication protocol commonly employed in industrial automation and control systems. It operates on a master-slave architecture, where the master device, such as a PLC or IoT gateway, initiates communication by sending requests to the connected slave devices, including sensors and actuators.

The protocol is designed for half-duplex and multipoint communications, enabling the connection of up to 247 slave devices on the same bus. The master device supports two modes of communication:

- Unicast mode: In this mode, the master communicates with a single slave device by directing the message to its unique address[5]. This mode involves two messages: the master's request and the slave's response.
- Multicast mode: This mode allows the master device to broadcast a message to all slave devices without expecting any response from them. Consequently, only "write" messages can be sent.

Communication in Modbus RS-485 occurs via a twisted pair of wires utilising differential signalling. Data is transmitted as a voltage difference between two data lines, typically referred to as Line A and Line B, enhancing noise immunity and enabling long-distance transmission.

[5] Address 0 is reserved for broadcast messages, enabling communication with all slave devices simultaneously. Slave devices are assigned addresses ranging from 1 to 247, while addresses 248 to 255 are reserved. The master device itself does not have an address as it is the one who initiates and controls the communication.

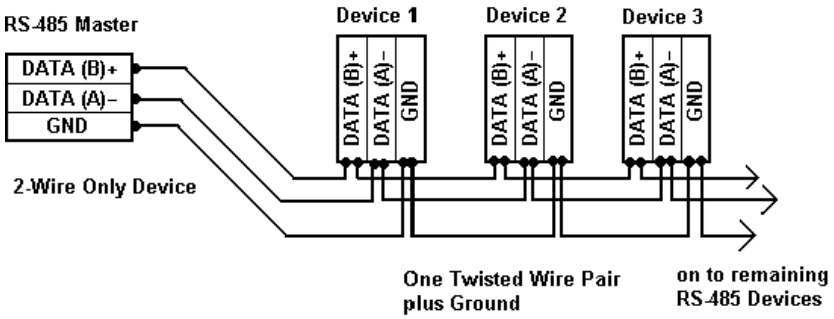


Figure 25. Modbus RS-485 schematic (Csanyi, 2016).

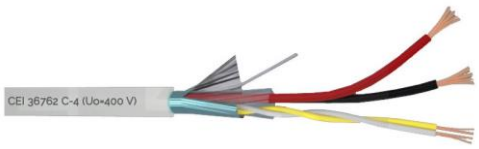


Figure 26. Twisted pair of wires used for RS-485 communications.

4.3.3 DATA DIODE

To ensure robust security measures, a data diode based on RS-232 communication is implemented, establishing a one-way communication channel between the IoT gateway and the outside world.

RS-232 is a widely used protocol for serial communication, and it is commonly employed in computer systems and industrial applications. Unlike RS-485, which utilises differential signalling, RS-232 employs a single-ended signalling method. This means that it utilises voltage levels with reference to a common ground, rather than a voltage difference between two lines.

Data is transmitted sequentially, one bit at a time, over a single data line, allowing communication between two devices: a sender (transmitter) and a receiver. Additionally, it operates in full-duplex mode, enabling simultaneous bidirectional communication.

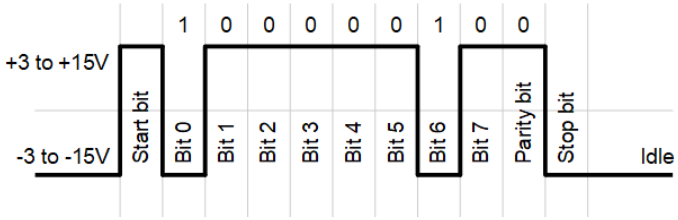


Figure 27. RS-232 data transmission (Anticyclone Systems, 2023).

DB-9 connectors are utilised to establish the physical connection, employing specific pin assignments for data transmission (Tx), data reception (Rx), and ground (GND) signals. The protocol's main drawback is its susceptibility to noise, restricting its use to short-range communications, covering only a few meters, at relatively low data rates.

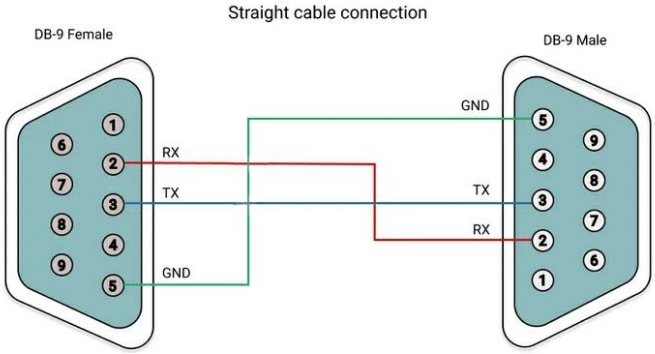


Figure 28. RS-232 connection between two devices (Weis, 2019).

Chapter 5. DEPLOYMENT OF NOA ARCHITECTURE

After thoroughly analysing the technology employed in the project, this fifth chapter focuses on presenting the extensive work carried out during its execution. The chapter is divided into several sections, each addressing crucial aspects essential for the successful implementation of the NOA architecture in ICAI's factory.

To begin with, special focus is given to the development of the information model and the integration of OPC UA servers and clients. These components serve as the foundation of the project, facilitating communication and data exchange among various elements within the factory.

Next, the section explores the integration of the Monitoring and Optimisation sensors. These sensors play a vital role in extracting valuable data from the factory, enabling future projects in predictive maintenance, and providing an accurate understanding of the factory's state and the health status of its components.

Furthermore, the configuration and installation of the hardware equipment required for extracting data from the PLCs and the newly implemented sensors is covered. As mentioned earlier, this data extraction process is facilitated through the utilisation of the IOT2050 Gateway.

Finally, the chapter discusses the development of the dashboard to monitor and visualise the factory's activities, showcasing a real application powered by NOA.

5.1 INFORMATION MODEL

The primary objective of an information model is to provide context to data, enabling every OPC UA application to comprehend the overall situation within the factory. Therefore, it must accurately represent the structure, behaviour, and semantics of the data, allowing client-server communication between the different devices that make up the factory and the NOA system[6].

5.1.1 ICAI FACTORY

ICAI's factory is a compact production facility consisting of four interconnected stations, each equipped with specialised sensors, actuators, and robotic arms to carry out various operations. The main conveyor belt, known as the "highway," forms the backbone of the entire production process, facilitating the movement of pallets between the four stations.

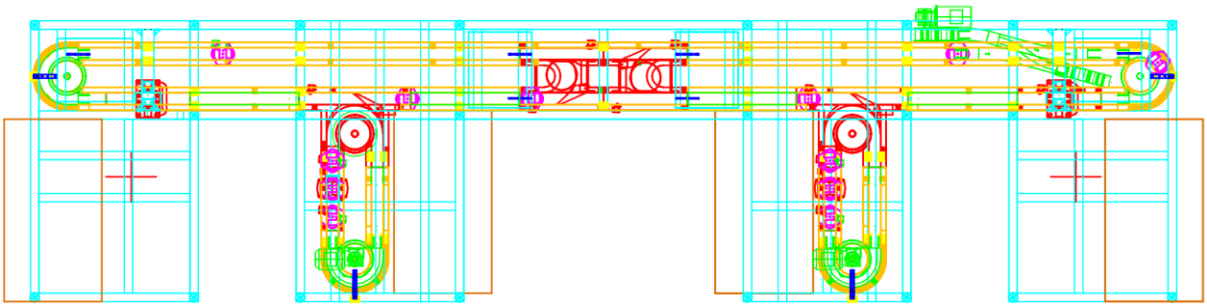


Figure 29. ICAI Factory blueprint.

These stations are strategically positioned to perform specific tasks in the manufacturing process. Stations 1 and 4, which are identical, are equipped with lifts that raise the pallets to the workspace where the robots perform their operations. Meanwhile, stations 2 and 3 have their own conveyor belts, referred to as satellite 2 and satellite 3, respectively. These belts enable the diversion of pallets from the main conveyor belt to the working space in each station, where pallets are locked in place by a positioner to perform operations.

[6] The basic concepts of information modelling are explained in Annex II. Information Modelling in OPC UA.

Each station is also fitted with two stoppers or retainers, which allow the operators to control the flow of material within each individual station by halting the movement of pallets. Collaborative robots are present in each station to carry out assembly operations in the production line, allowing them to work in tandem with human operators.

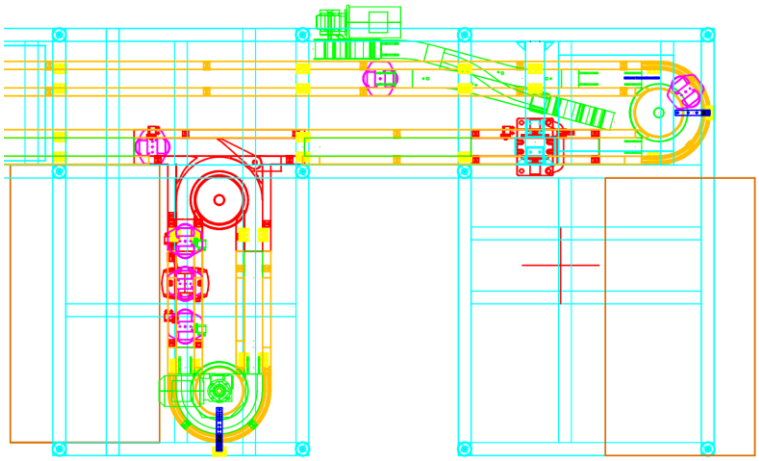


Figure 30. Stations 3 and 4 close-ups.

Additionally, the factory uses control mechanisms known as diverts and combines that are installed in the main conveyor belt to allow operators to control pallet flow. These mechanisms are equipped with two stoppers, similar to those found in the stations, that prevent pallets from crashing into control mechanisms or colliding with other pallets in the system. In addition, there is a general stopper in the highway that can halt all movement in the factory.

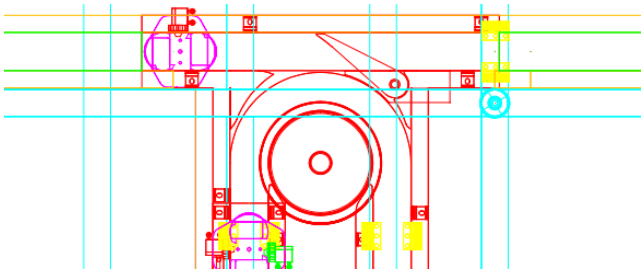


Figure 31. Conveyor Control System. Divert 34.

5.1.2 INFORMATION MODEL

In this section, the design of the factory's information model is undertaken in accordance with the graphical representation guidelines established by the OPC Foundation. The model is implemented in XML format[7], which will later be compiled to build the OPC UA server.

To ensure a streamlined development process, it is highly recommended to download the UAModelDesign.xsd file provided by the OPC Foundation. This resource aids in identifying potential coding errors and offers valuable insights during the development process.

5.1.2.1 Base Device Type

To ensure precise representation of objects and their essential attributes, the factory's information model incorporates a "base device" ObjectType. This serves as a standardised template for all Object and ObjectType nodes that will be subsequently defined. The base device node encompasses the following mandatory attributes.

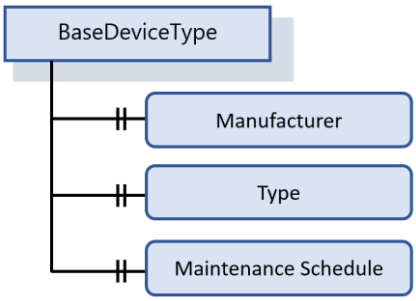


Figure 32. Base Device with mandatory attributes (Original work).

The "Manufacturer", "Type", and "Maintenance schedule" attributes are important and common to all devices in the factory as they provide information that is essential for efficient operation, maintenance, and planning.

The Manufacturer attribute identifies the manufacturer of the device or asset, which is important for quality assurance, warranty, and technical support. Identifying the manufacturer can also

[7] The full XML file can be found in Annex III. Information Model XML File.

help with troubleshooting, as companies have different specifications, requirements, and recommendations. On the other hand, the Type of asset attribute specifies the type of device, which is essential for inventory management, asset tracking, and resource allocation.

Finally, the Maintenance schedule attribute indicates the schedule for maintenance of the device or asset, which is crucial for ensuring optimal performance, minimising downtime, and extending the lifespan of the asset. This attribute helps operators plan and execute maintenance tasks proactively, reducing the risk of unexpected failures or breakdowns that could disrupt production or cause safety hazards.

The XML code for the base-type node is as follows.

```
<!--
Base Class for every device
-->
<opc:ObjectType SymbolicName="BaseDeviceType" BaseType="ua:BaseObjectType">
<opc:Description>
The Base Type for all devices in the factory is a foundational template that
specifies the essential attributes that every device must possess.
</opc:Description>
<opc:Children>
<opc:Property SymbolicName="Manufacturer" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite"/>
<opc:Property SymbolicName="Type" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite"/>
<opc:Property SymbolicName="MaintainanceSchedule" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Maintainance Schedule</opc:BrowseName>
</opc:Property>
</opc:Children>
</opc:ObjectType>
```

5.1.2.2 Basic sensor and actuator types

Having established the foundational device base type, the subsequent phase entails the definition of Node ObjectTypes (classes) for the actuators and sensors constituting the factory. For sensors, two fundamental node types will be utilised: analog and digital sensors. Analog sensors will encompass a numeric attribute incorporating properties such as Engineering Units and EURange. Conversely, digital sensors will possess a Boolean value indicating the detection status of an element.

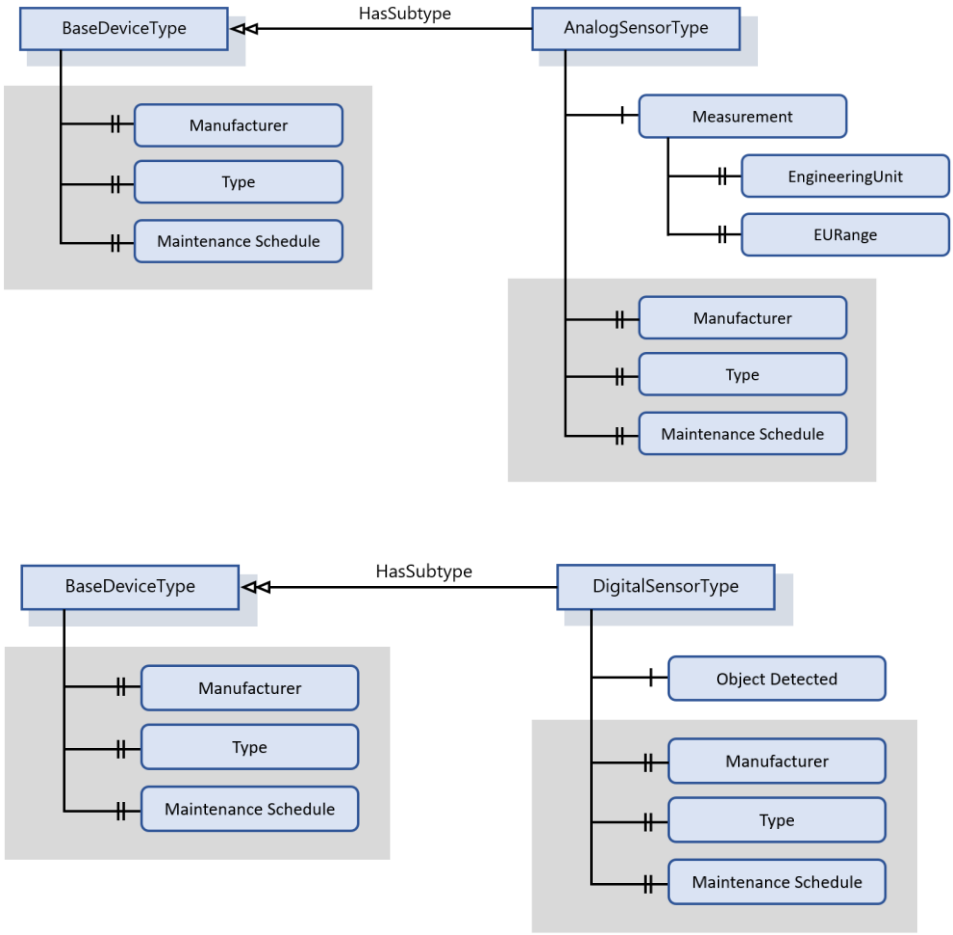


Figure 33. Basic analog and digital sensor schema (Original work).

On the other hand, actuators encompass a Boolean attribute denoting their activation status (ON/OFF). This attribute will be employed to represent both straightforward mechanisms such as pneumatic actuators and more intricate systems like robots, providing an indication of their operational state.

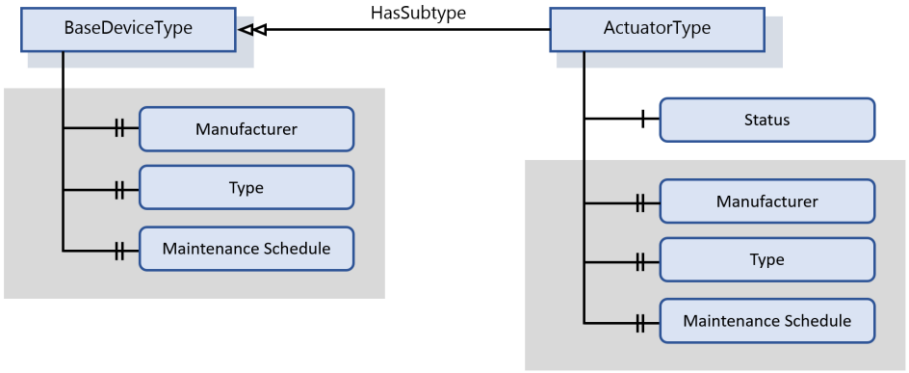


Figure 34. Basic actuator definition (Original work).

A noteworthy observation is that these newly introduced nodes inherit all the attributes defined in the base class node. This inheritance holds significant importance as it ensures consistency and standardisation throughout all devices present within the factory.

5.1.2.3 Specific sensors and actuators

After defining the basic sensors and actuators, the focus shifts towards modelling more specific devices. It is important to emphasise that, although these devices are visually represented as objects, in the underlying XML file they are defined as types (classes) to instantiate them as many times as required.

The initial stage involves defining the analog sensors installed within the factory, such as temperature and humidity sensors. These objects encompass distinct properties, including the sensor's measurement accuracy.

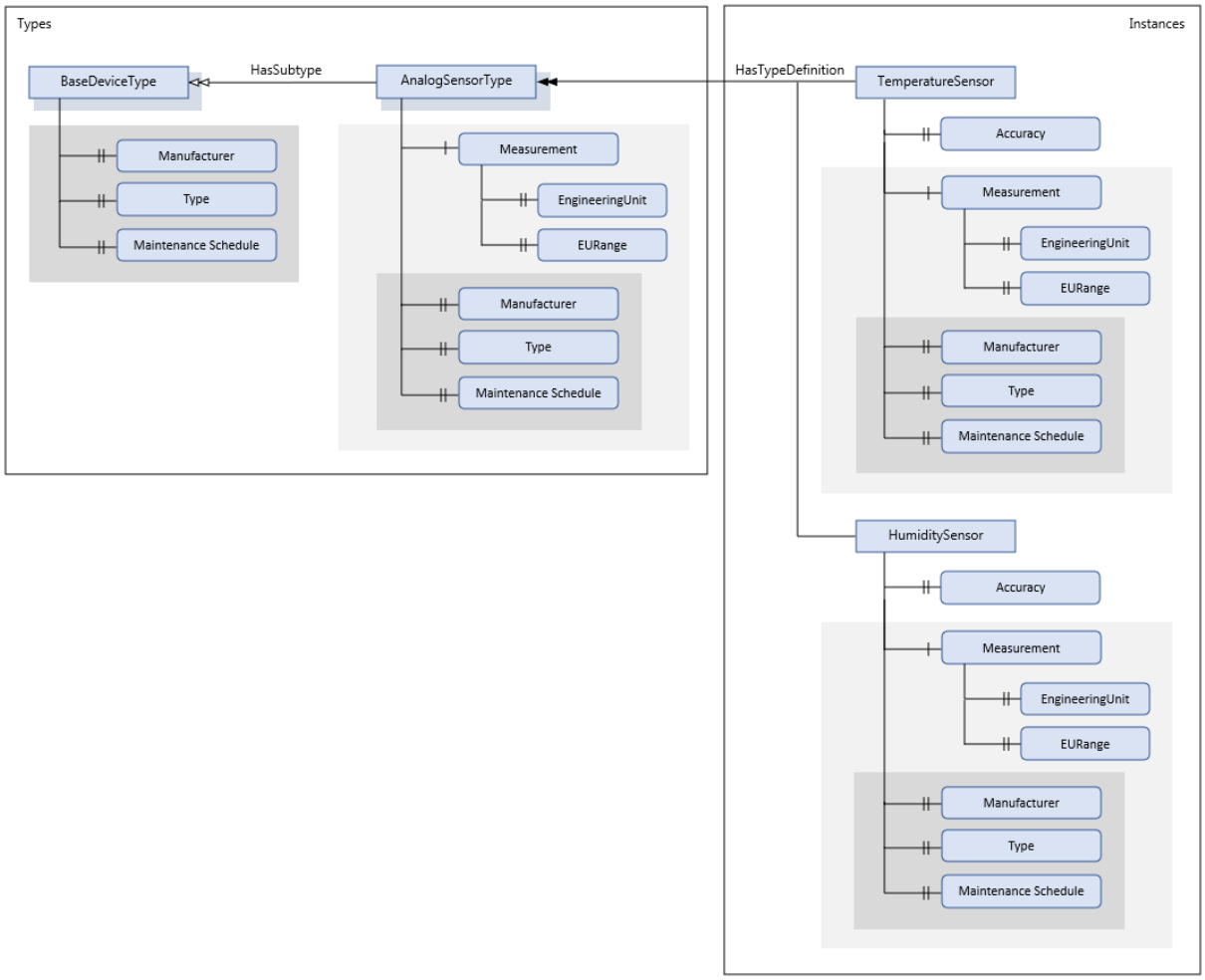


Figure 35. Temperature, Humidity and Vibration sensor definition (Original work).

Regarding digital sensors, a proximity sensor is modelled featuring an attribute that describes its overall detection range. Additionally, a limit switch sensor is included in the model.

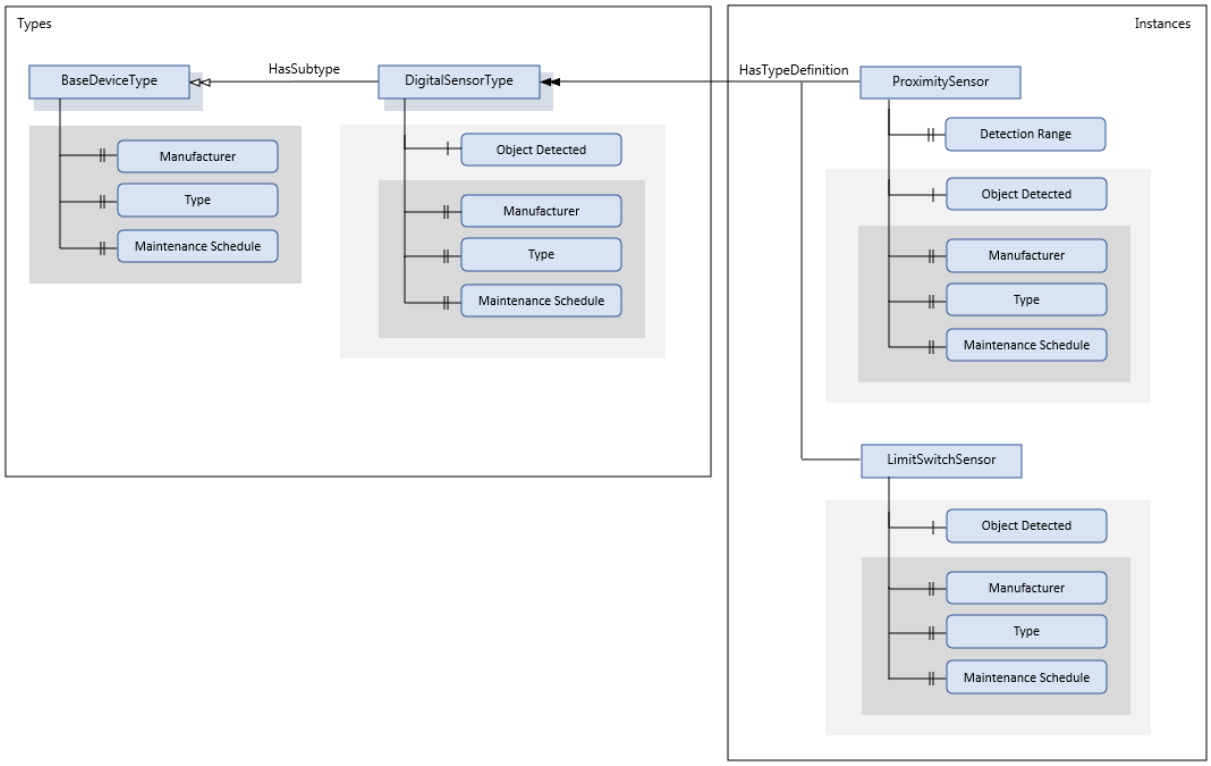


Figure 36. Specific digital sensors of the factory (Original work).

Furthermore, specific types of actuators are defined, such as pneumatic actuators, conveyors, and robots, each characterised by their distinctive attributes and properties. Notably, conveyors incorporate an additional variable to represent the speed of the belt, in addition to other properties like its status, which indicates whether the conveyor's engine is started. Like other analog variables, the Engineering Units (in this case, meters per second) and the EURange, which specifies the minimum and maximum speed of the conveyor, are included.

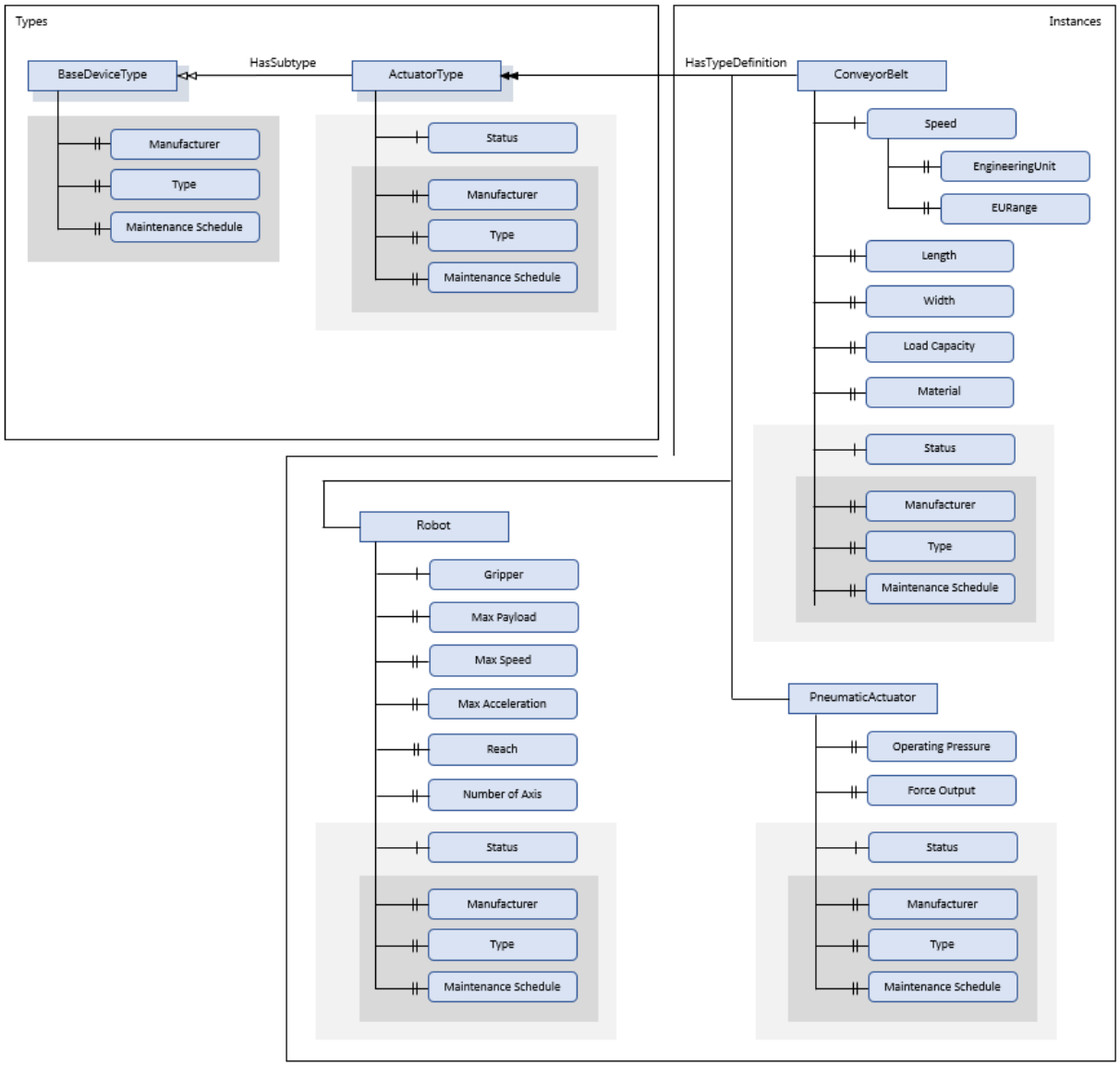


Figure 37. Specific actuators present in the factory (Original work).

5.1.2.4 Integrated Modules:

After defining the foundational elements of the factory, the next step is to create the more complex modules that comprise it. Drawing upon the details provided in section 5.1.1 which outlines the factory's blueprints and components, the schemas for stoppers, lifts, positioners, diverts, and combines are formulated. It is important to note that certain modules, like the diverts and combines, comprise both fundamental elements, such as proximity sensors, and complex modules (stoppers) defined in this same section.

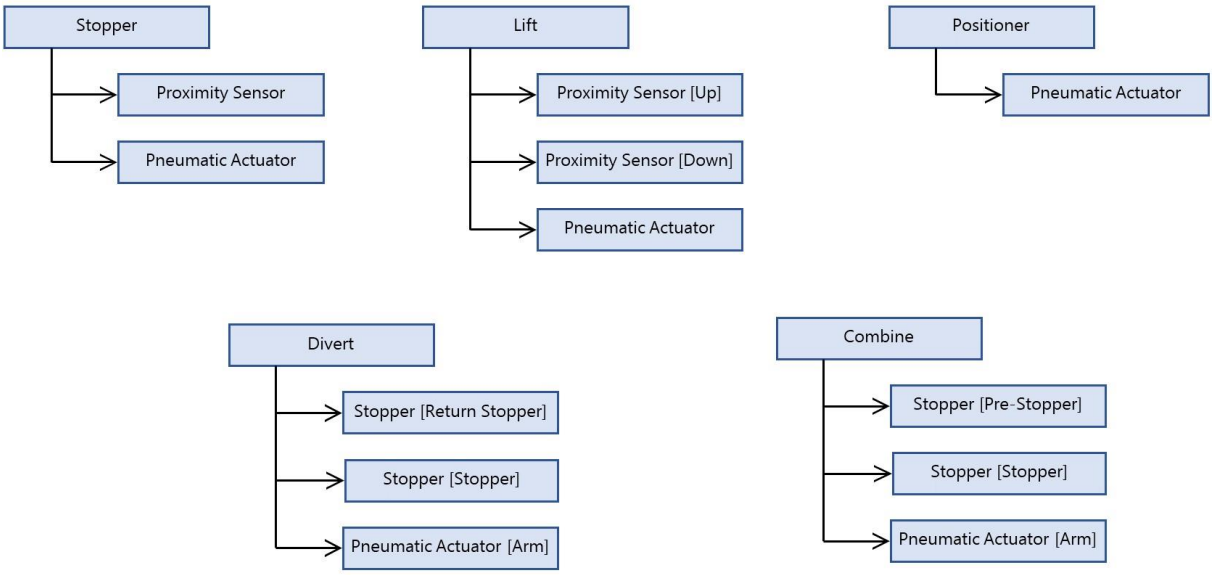


Figure 38. Factory's integrated modules (Original work).

To illustrate the inheritance of child nodes within these modules, a comprehensive hierarchy of the Divert and Combine objects is provided, showcasing the relationships and dependencies between the various components.

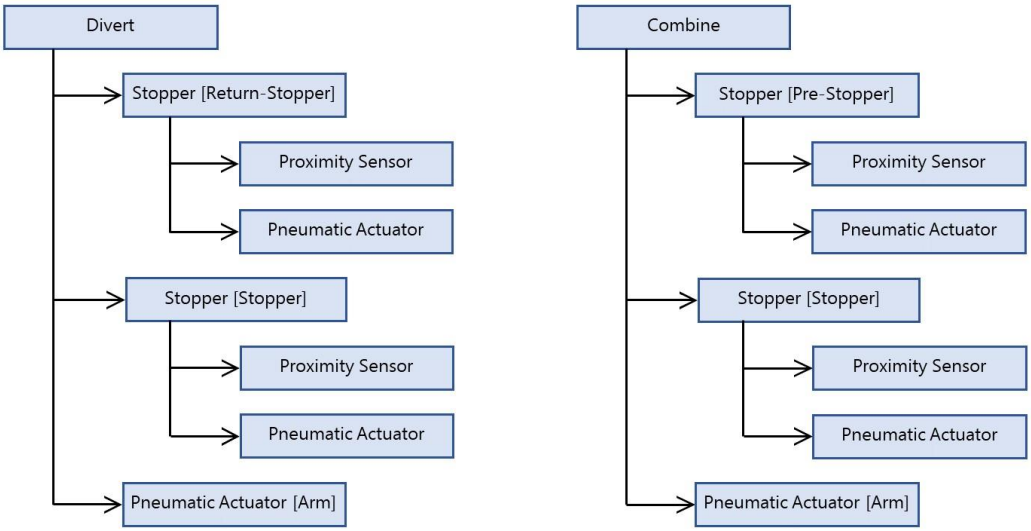


Figure 39. Detailed schema of the Divert and Combine objects (Original work).

5.1.2.5 Stations

With the completion of the modules, the next step involves integrating them into the information model for the four workstations situated within the factory. As previously mentioned, stations 1 and 4 are equipped with lifts that raise the pallets to the robot's working area, while stations 2 and 3 have a simpler configuration with a positioner that holds the pallets in place, allowing the robot to work on them.

Each station is further equipped with a robot and two stoppers that regulate the movement of pallets into the lift or positioner, depending on the station's configuration.

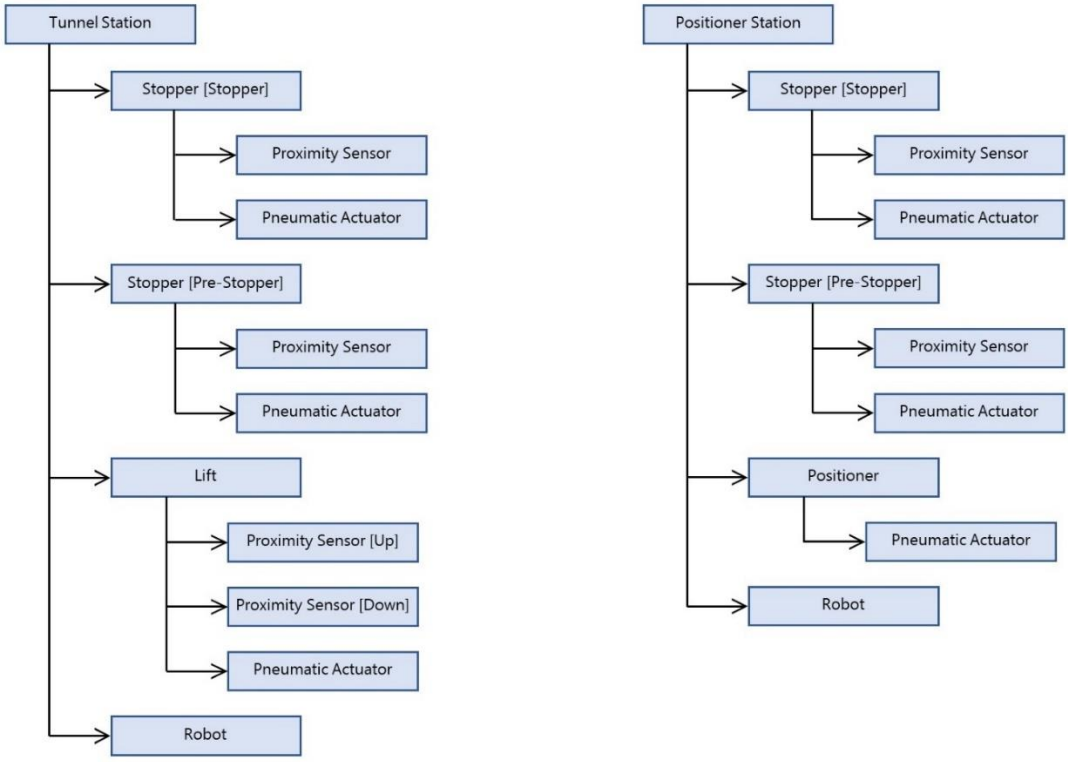


Figure 40. Station Types (Original work).

5.1.2.6 Factory

To complete the information model, the modules and stations are consolidated within a single folder-type object node called "Factory." To enhance the structural organisation of data within the model, the Factory node is categorised into two primary sectors: Stations and Conveyors. Additionally, the Conveyors sector is further subdivided into Conveyor Belts and Conveyor System Controls, providing a distinct separation between the physical belts and the actuators responsible for guiding the pallets throughout the factory.

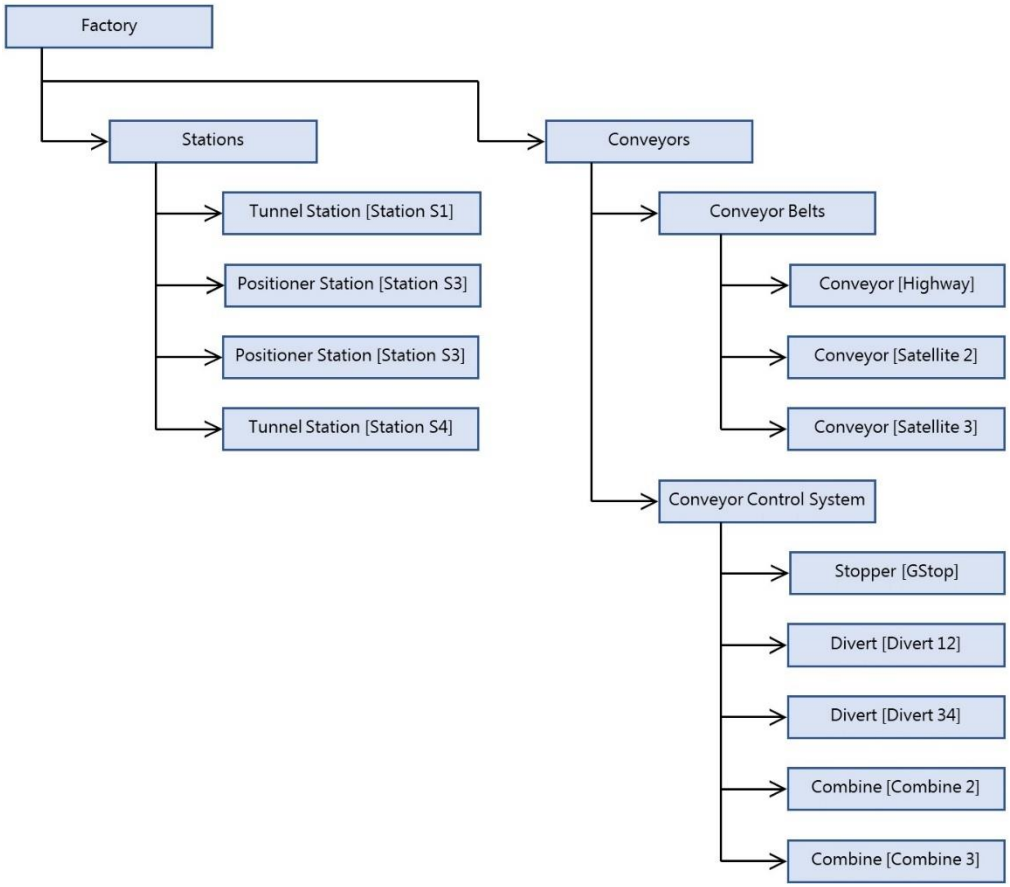


Figure 41. Overall ICAI Factory information model (Original work).

5.2 OPC UA AGGREGATE SERVER

Building upon the previous chapter, the focus now shifts towards the implementation of the OPC UA server. The current project focuses on the development of an aggregate server that acts as a centralised repository for all data generated within the factory. Regarding the sources of information, the project relies on OPC UA servers deployed on each of the factory's S7-1500 PLCs and the readings from the M+O sensors.

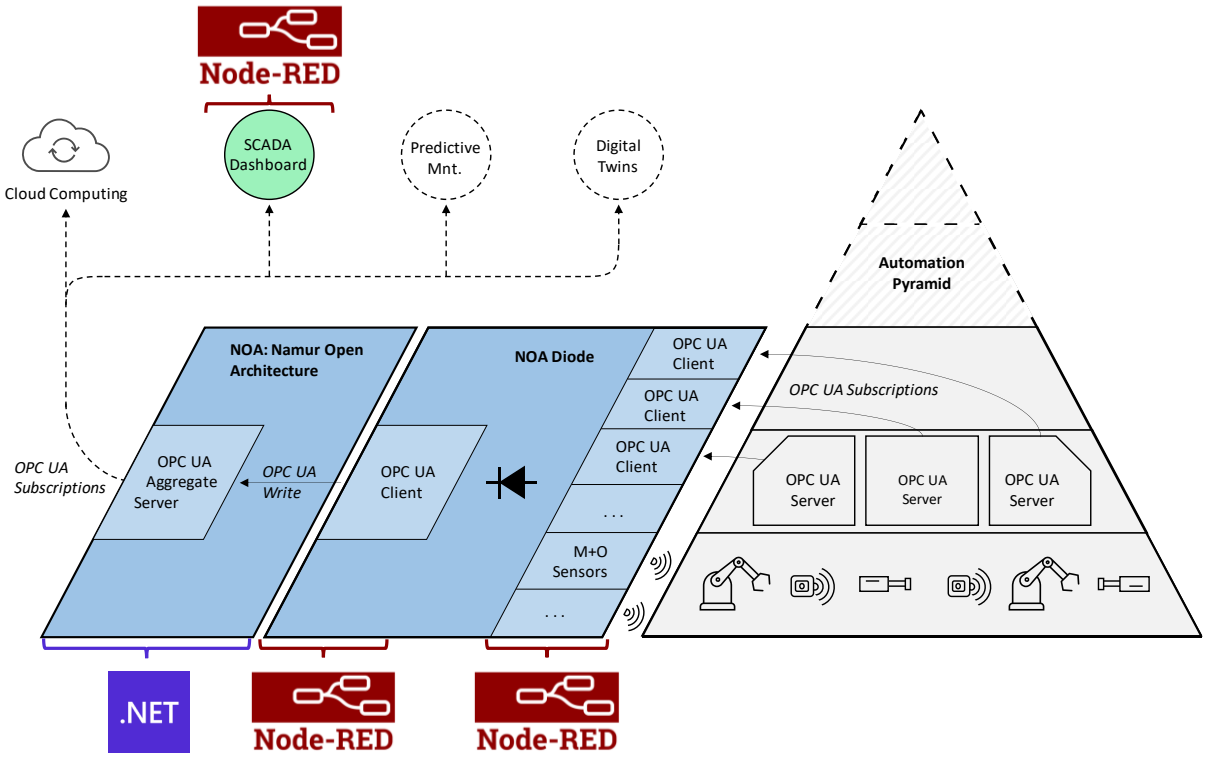


Figure 42. Development tools and environments used to build OPC UA Servers and Clients (Original work).

The diagram above showcases the various services provided by OPC UA servers, such as read, write, or subscribe, which clients can use to monitor, control, and visualise the data. A noteworthy characteristic of these services is the possibility for clients to subscribe to specific nodes and receive automatic notifications from the server when any changes occur. As a result, OPC UA systems can operate based on a hybrid approach that combines both client-server and publish-subscribe architectures.

A flowchart is presented below, illustrating the sequential processes undertaken by an OPC UA client when establishing a connection with a server to perform tasks such as writing, reading, calling methods, or subscribing to nodes.

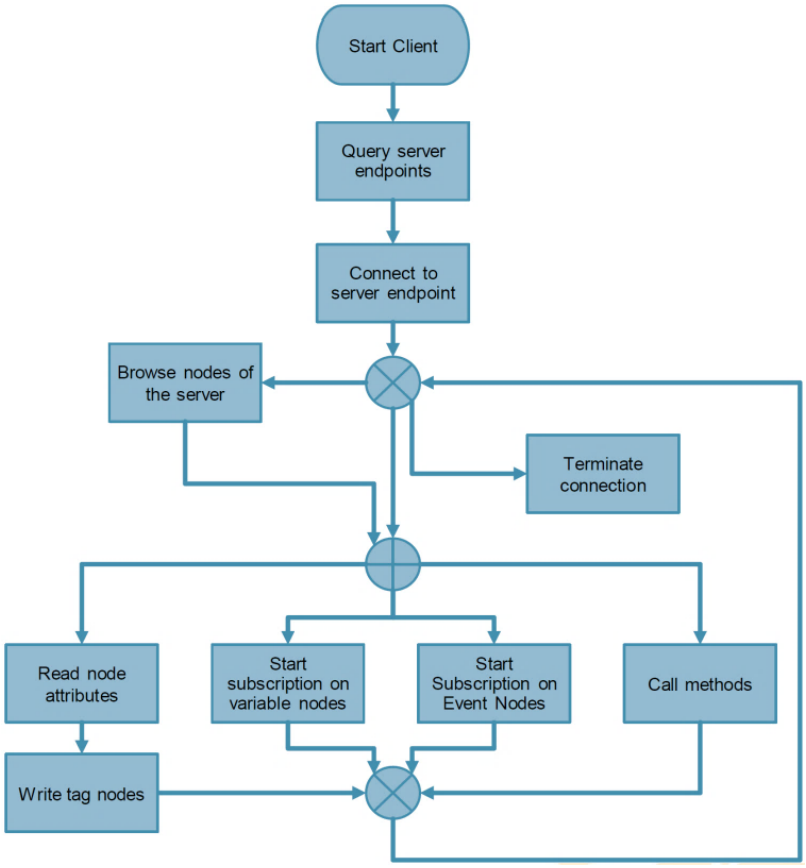


Figure 43. OPC UA client flowchart (Mondejar, 2023).

5.2.1 BASIC CONCEPTS OF OPC UA APPLICATIONS

As previously stated, OPC UA applications can communicate via binary TCP or HTTP/Soap protocols. To enable clients to use either of these channels, servers must support at least one of the two profiles provided by the OPC Foundation's UA communication stack.

As depicted in the diagram below, both communication stacks consist of three layers, namely the transport protocol, security protocol, and data encoding.

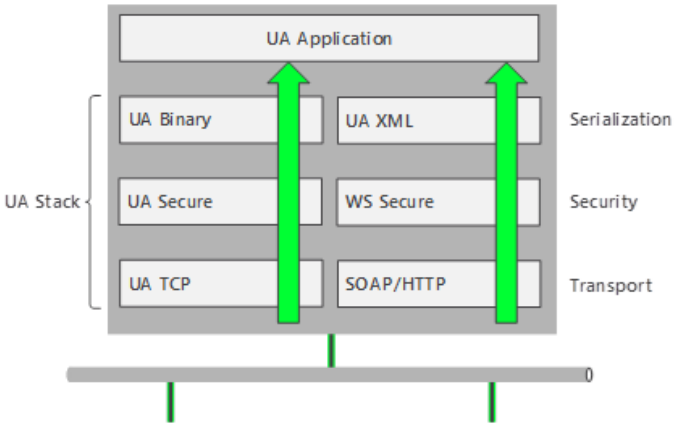


Figure 44.OPC UA communication stacks (Inmation, 2021).

- Transport layer: defines the communication protocol used to communicate the OPC UA servers and clients.
- Security layer: provides the security measures to protect the confidentiality and integrity of the data being transmitted.
- Serialisation layer: is responsible for encoding and decoding the data being transmitted.

5.2.1.1 Transport Layer

In this project, the aggregate server employs the TCP communication stack due to its superior performance and the fact that it solely needs to communicate with clients within the LAN, eliminating the need for communication over the internet.

5.2.1.2 Security Layer

Regarding the security layer, servers currently have the option of utilising the *SecurityModes* and *SecurityPolicies* provided by the OPC Foundation. The specific security measures implemented on a server may vary greatly depending on its intended use. For instance, a server can be configured with no security at all, making it suitable for use in academic research or in isolated network environments where cybersecurity is not a concern. Alternatively, a server can be set up with multiple layers of security, such as data encryption, authentication, or certificate trust. This second approach is typically employed in servers and applications that require transmission of confidential data.

The tables below summarise all Security Policies and Security Modes for the endpoints of the server.

Table 6. Security Policies in OPC UA applications.

Security Policy	Description
None	No security is applied, and communication is in plaintext.
Basic128Rsa15	This policy uses 128-bit encryption keys and RSA 15 encryption algorithm to encrypt communication
Basic256	This policy uses 256-bit encryption keys and AES encryption algorithm to encrypt communication.
Basic256Sha256	This policy uses 256-bit encryption keys and SHA-256 hashing algorithm to ensure message integrity and confidentiality.
Aes128Sha256RsaOaep	This policy uses Advanced Encryption Standard (AES) with 128-bit key size and SHA-256 hashing algorithm for message integrity, and RSA with Optimal Asymmetric Encryption Padding (OAEP) for key exchange. [Not available with the current OPC Foundation SDK]
Aes256Sha256RsaPss	This policy uses AES with 256-bit key size and SHA-256 hashing algorithm for message integrity, and RSA with Probabilistic Signature Scheme (PSS) for key exchange. [Not available with the current OPC Foundation SDK]

Table 7. Security Modes in OPC UA applications.

Security Mode	Description
None	No security is applied, and communication is in plaintext.
Sign	Sign security mode ensures message integrity by adding a digital signature to each message. The signature is generated using the private key of the sender and can be verified using the sender's public key. This ensures that the message has not been tampered with during transmission.
Sign & Encrypt	Sign&Encrypt security mode provides both message integrity and confidentiality. In addition to generating a digital signature, this mode encrypts the message to prevent unauthorised access or reading of the message content. The encryption is performed using a shared secret key that is negotiated between the sender and receiver during the communication process.

The aggregate server has been programmed to support all possible combinations of security policies and modes, allowing users to adjust the security configuration of the server according to their specific requirements. It is important to underline that, during boot-up, the server is set to operate with the most robust security measures by default: BasicSha256Sha256 with *Sign&Encrypt*.

The following image compares the aggregate server after bootup to one server that has enabled all security policies and security modes. As shown below, each server exposes a specific number of endpoints to which clients can connect.

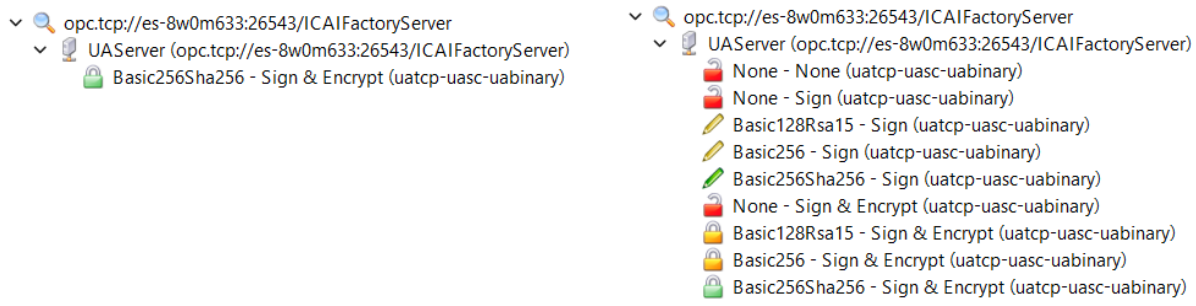


Figure 45. Server with one secure endpoint vs Server with all available endpoints (Original work).

5.2.1.2.1 Certificate Trust

When utilising the signing security mode in OPC UA communication, the encryption of messages is performed with a signature generated by the private key of OPC UA's applications certificates. To enable secure communication, both, client and server, must possess and trust each other's public keys, which requires the following handshake procedure.

1. When a client first contacts a server (via the *GetEndpoints* request), the server responds by sending its application certificate to the client, which contains its public key.
2. The client validates the server's certificate to ensure that it is trustworthy. If the certificate is trusted, the client can proceed to open a secure channel with the server through an *OpenSecureChannel* request.
3. As part of the *OpenSecureChannel* request, which is encrypted using the server's public key, the client also sends its own certificate back to the server, so that the server can verify the client's identity.

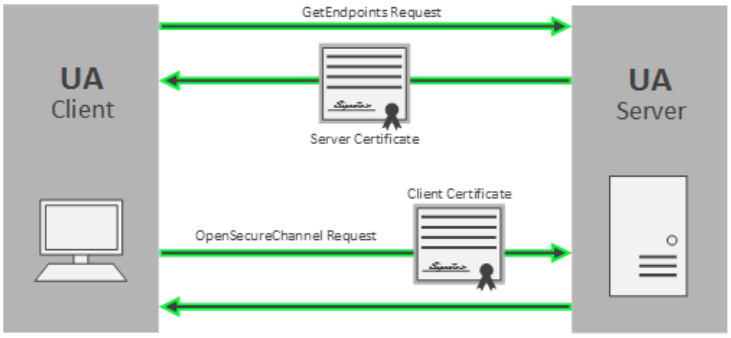


Figure 46. Handshake procedure between OPC UA clients and servers (Inmation, 2021).

OPC UA servers can be configured to only accept connections from trusted client applications. Therefore, when a new client application attempts to connect, its certificate is placed in an "untrusted certificate" folder. For the client to establish a connection, the server administrator must first place the certificate in the "trusted certificate" folder.

Alternatively, servers can be configured to allow connections from both trusted and untrusted client applications. This second approach should only be used during the development phase of a project, never in a production environment.

5.2.1.3 User Authentication

As previously mentioned, the UA specification utilises two security layers: transport layer security, which ensures message integrity using *applicationInstance* certificates, and a secure channel built between clients and servers through public key encryption.

In addition to transport layer security, OPC UA can use application-based security to ensure proper access control to the server, known as authentication. This mechanism is performed each time a new session between a client and a server is activated. During this process, the client is given a *userIdentityToken*, which allows the server to determine if the token is authorised to connect, preventing unauthorised access to the server. This introduces an additional layer of security, as not only the client certificate needs to be trusted, but also a valid token must be provided for user authentication.

There are several types of user authentication methods in OPC UA, including:

Table 8. Authentication Methods to access OPC UA servers.

Token	Description
Anonymous	This method allows clients to connect without providing any user identity.
UserNamePassword	Clients provide a username and password, which are authenticated by the server.
X509	Clients provide a digital certificate that is validated by the server.

By default, the aggregate server at ICAI is configured to utilise *UserNamePassword* authentication, which requires clients to provide valid login credentials when establishing a connection. However, the server can be configured to allow anonymous connections if the administrator deems it necessary.

5.2.2 OPC UA AGGREGATE SERVER

After providing an overview of the fundamental concepts of OPC UA applications, this section shifts its focus to the development of the OPC UA Aggregate Server. The primary purpose of this server is to collect and store all the data obtained from the factory's PLCs and non-intrusive sensors. To build this server, the information model developed in Chapter 5.1.2 is compiled using the free UAModelCompiler provided by the OPC Foundation.

This compiler, an essential component of the OPC Foundation's toolkit, guarantees adherence to OPC UA standards while simplifying the generation of the required files for the server's construction.

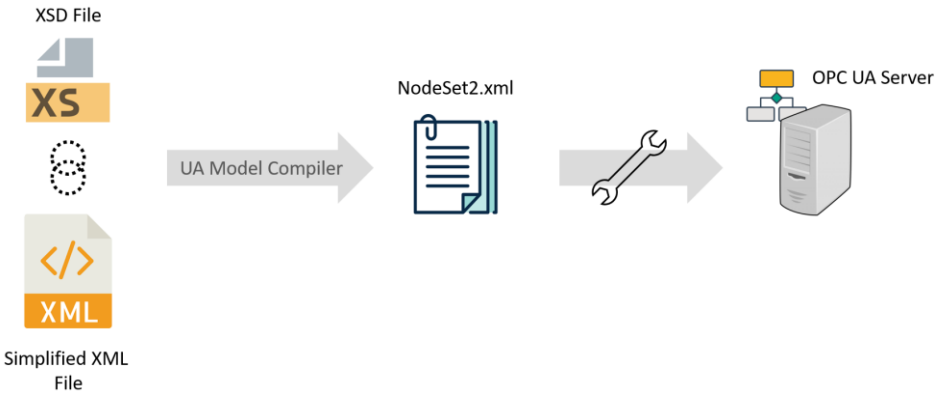


Figure 47. OPC UA Server Build Process Diagram (Original work).

5.2.2.1 Building the Server Application

Below are the outlined procedures for building an OPC UA server with the .NET framework and the OPC Foundation's SDK[8].

Step 1: Development Environment.

The initial step is to set up the development environment. This entails installing .NET (6.0) framework and Visual Studio IDE on the development machine.

[8] More in depth detailed steps can be found in Annex IV. OPC UA Server Files.

Step 2: Visual Studio Project.

Then, a new Visual Studio project must be created. The appropriate project template should be selected based on the specific requirements of the server, such as a Console Application or Windows Forms Application.

Step 3: OPC Foundation's SDK.

Step three involves incorporating the OPC Foundation's SDK into the project. This can be achieved by utilising the NuGet Package Manager or the Package Manager Console.

Step 4: Server Class

The creation of a server class (`server.cs`) serves as the backbone of the OPC UA server application. This class acts as the main interface for handling various server-related functionalities, such as initialisation, communication with clients, and managing server resources.

To create the server class, a new class needs to be added to the project. This class should inherit the `Opc.Ua.Server.StandardServer` base class provided by the OPC Foundation's SDK, gaining access to a set of pre-defined methods and events specifically designed for OPC UA server functionality.

Once the server class is created, it is necessary to override the required methods and events. These overridden methods and events will be responsible for handling server initialisation, processing client requests, and performing other server-related tasks. By customising these methods and events, the server behaviour can be tailored to the necessary requirements.

Step 5: NodeManager Class

The NodeManager class plays a crucial role in managing the information model of the OPC UA server. It is responsible for loading and handling OPC UA nodes and their properties, while managing the AddressSpace of the server.

Step 6: Server Endpoints and Security

During this step, an XML configuration file needs to be developed to configure the server's security settings. This file encompasses all information needed for setting up the server, including the endpoint URLs and ports, as well as the as authentication mechanisms, encryption protocols, and certificate management.

Step 7: Information Model Files and UAModelCompiler

In the final step, the UAModelCompiler tool is used to generate the necessary files to build the server's information model.

The UAModelCompiler tool can be sourced from the OPC Foundation's GitHub page (UA-ModelCompiler, 2023), and it can be launched from the command line, with the information model XML file specified as the input. Once the compiler is executed, the resulting output files, including the NodeSet2 XML file, can be added to the Visual Studio project.

To import the AddressSpace into the server, it is crucial to reference the newly created PredefinedNodes.uanodes file in the NodeManager class.

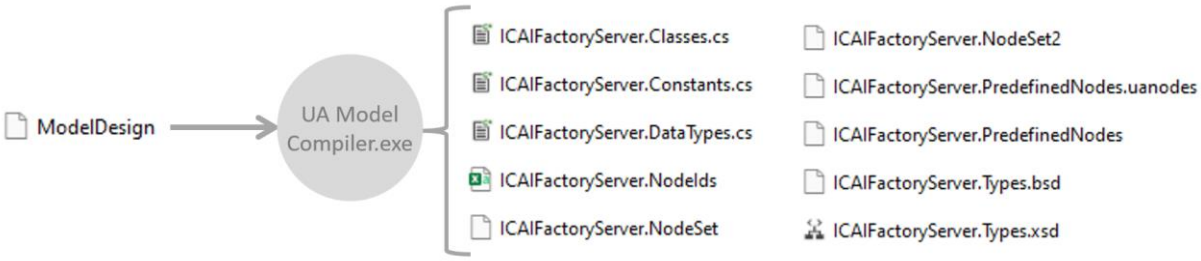


Figure 48. Files generated with the UAModelCompiler (Original work).

This last step can be performed at the beginning or at the end of the process. However, it is generally advisable to prioritise it as the initial step in server development. In this case, it has been included as the final step as it can be repeated as many times as needed to accommodate any new modifications performed on the information model.

5.2.2.2 Server Application

Lastly, a series of screenshots are included to highlight the features of the server application. To begin with, upon starting the server, users are greeted with a "home page" that offers a detailed description of the application and its role within NOA's architecture in ICAI's factory.

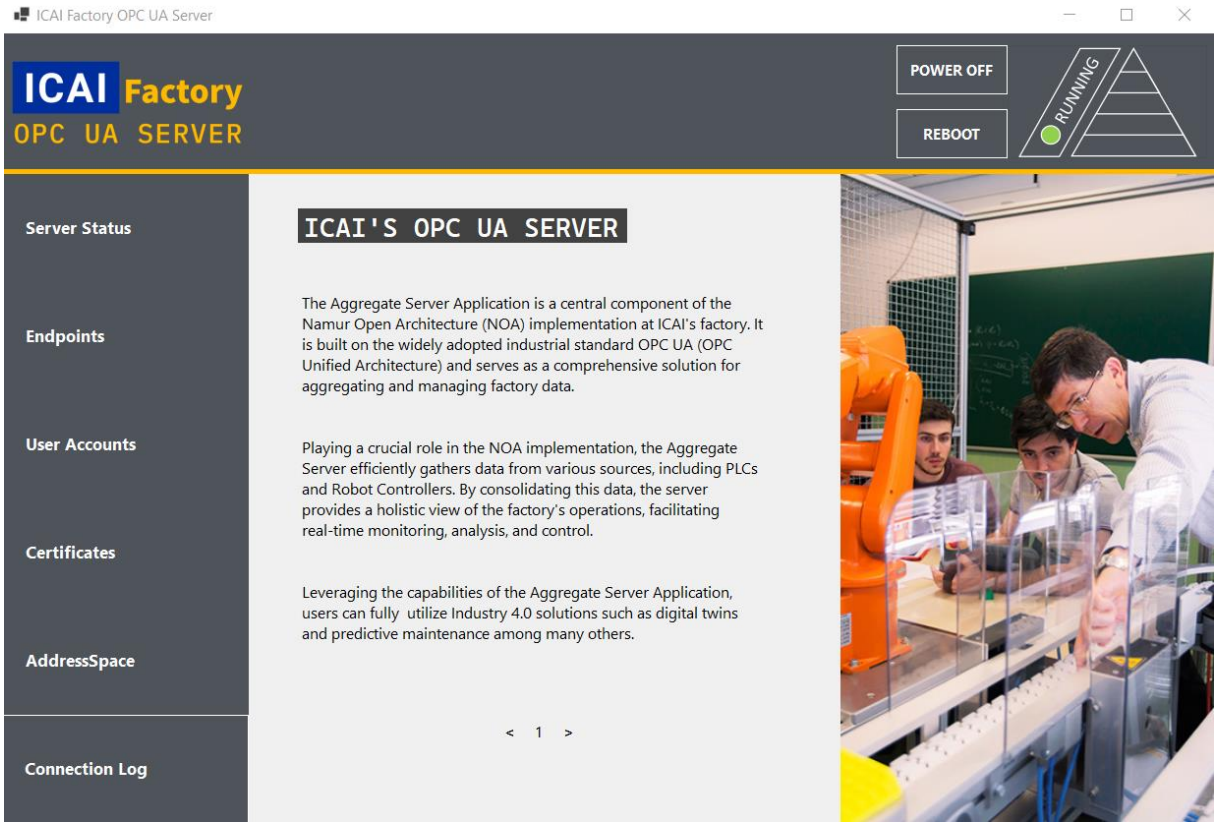


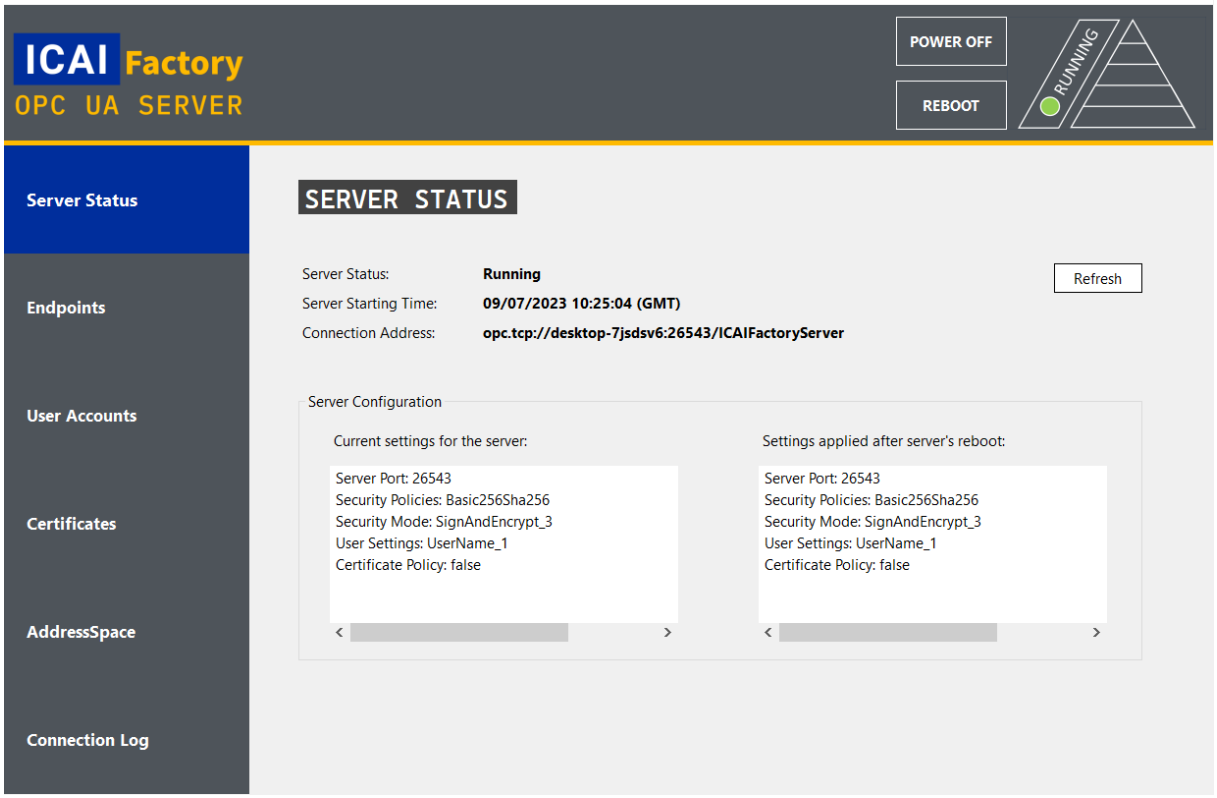
Figure 49. ICAI's aggregate server home page (Original work).

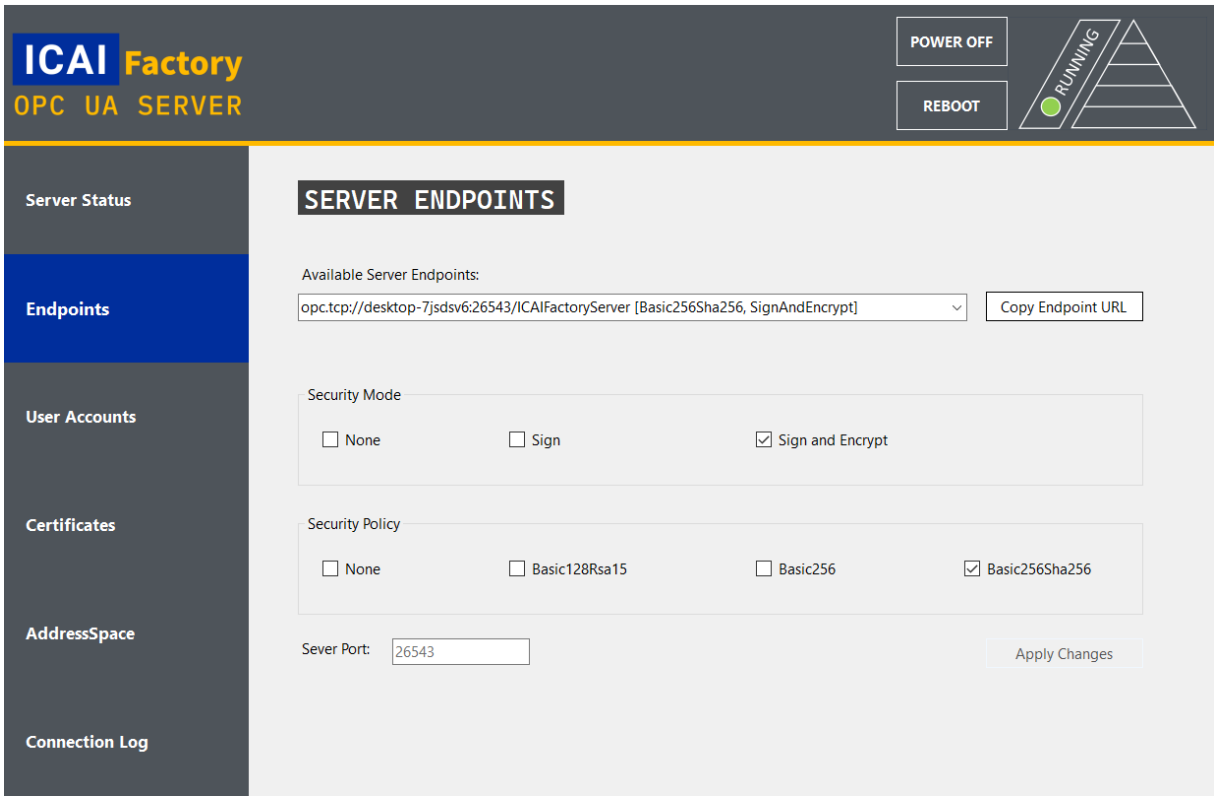
The application features two buttons located at the top right corner, enabling the user to turn the server ON or OFF and initiate a reboot. This functionality is noteworthy because configuration modifications require restarting the server for the changes to take effect.

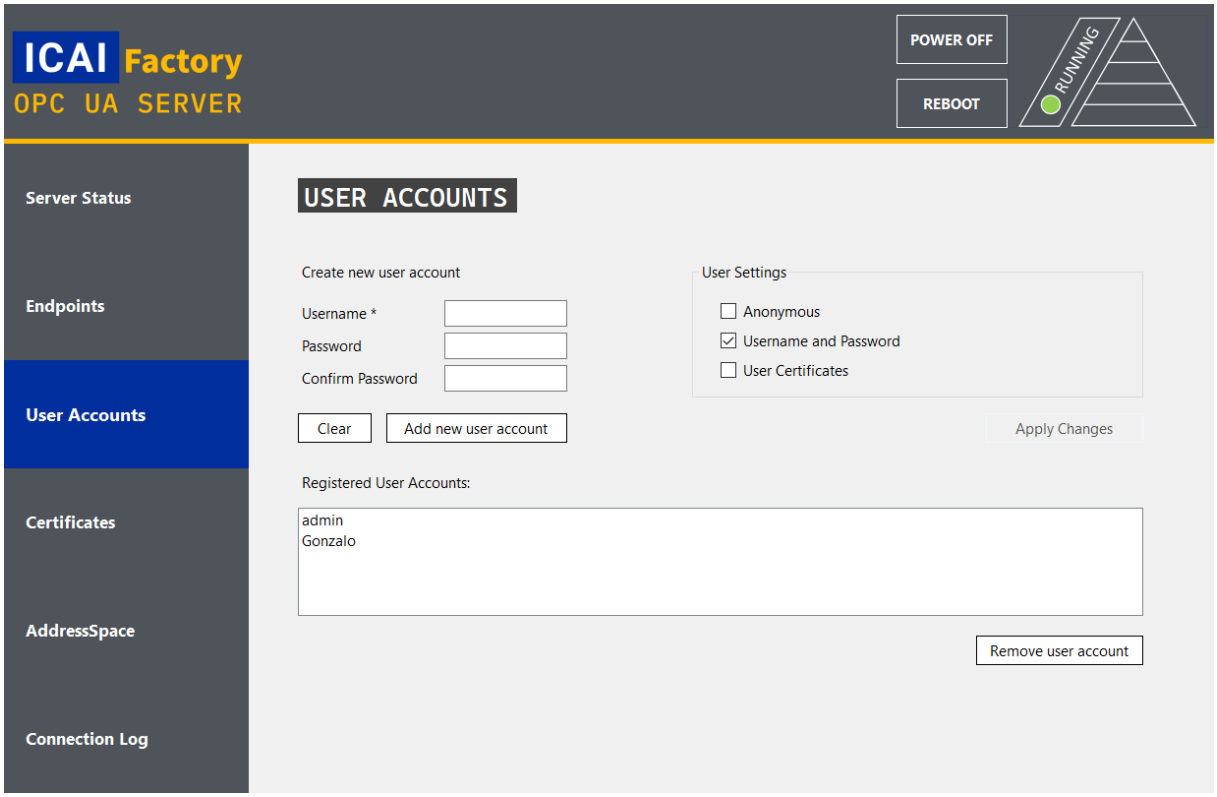
Additionally, a side panel is provided to navigate through its various menus:

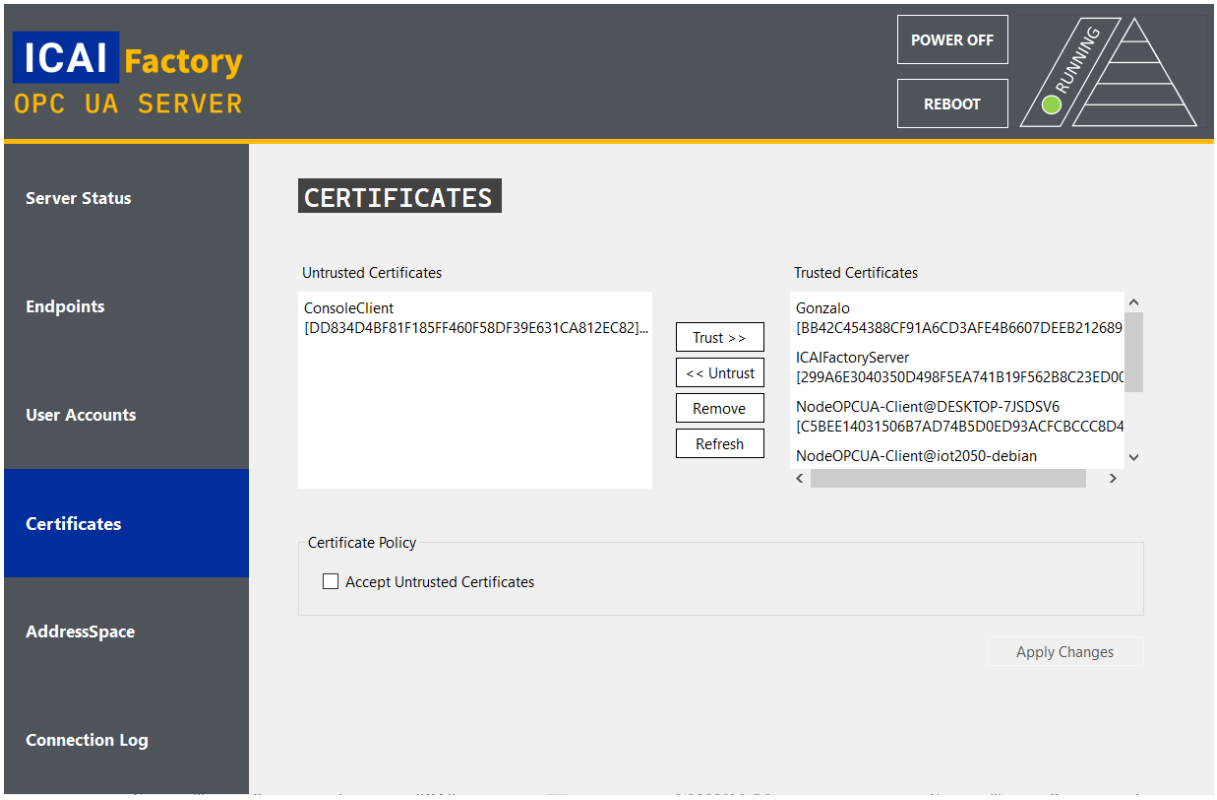
- Server Status: This menu presents the current server status (running, stopped, etc.), along with the server's start time and URL.
- Endpoints: Within this section, users can access a comprehensive overview of all available endpoints for the server and configure its security policies and modes.

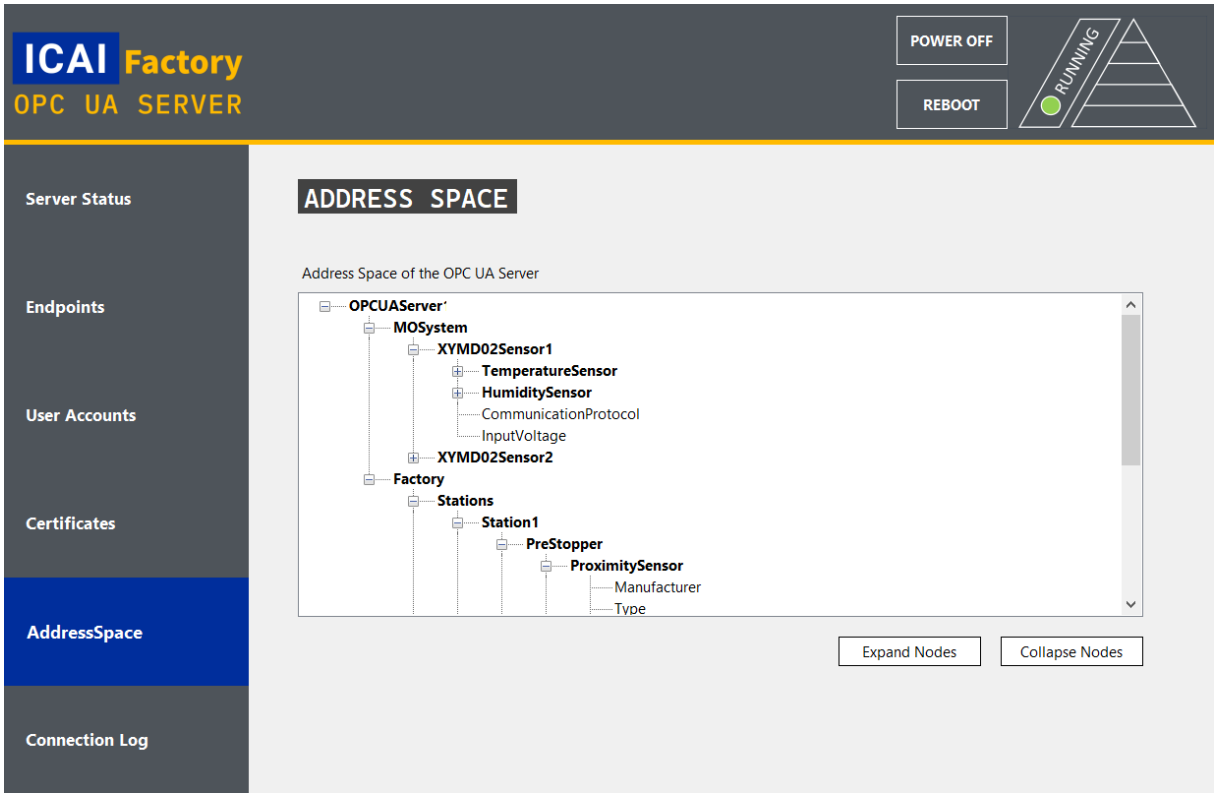
- User Accounts: This menu serves as a platform for managing authentication mechanisms. Server administrators can create, view, and delete user accounts, as well as configure the specific type of authentication required (Anonymous, Username and Password, X509 Certificate).
- Certificates: Both untrusted and trusted certificates are displayed, providing administrators with the capability to effectively manage certificate distribution. This includes controlling which clients are granted trust for establishing connections. Moreover, the app enables the deletion of rejected certificates and provides the option to accept untrusted certificates if deemed necessary.
- AddressSpace: Represented as a simple tree structure, this menu grants users access to a comprehensive display of all object nodes within the AddressSpace. It facilitates a clear understanding of the server's information model.
- Connection Log: It allows administrators to monitor and track login attempts. This log provides detailed information on successful connections to the server and authentication failures, serving as a valuable security tool and allowing users to export the log data as plain text files.

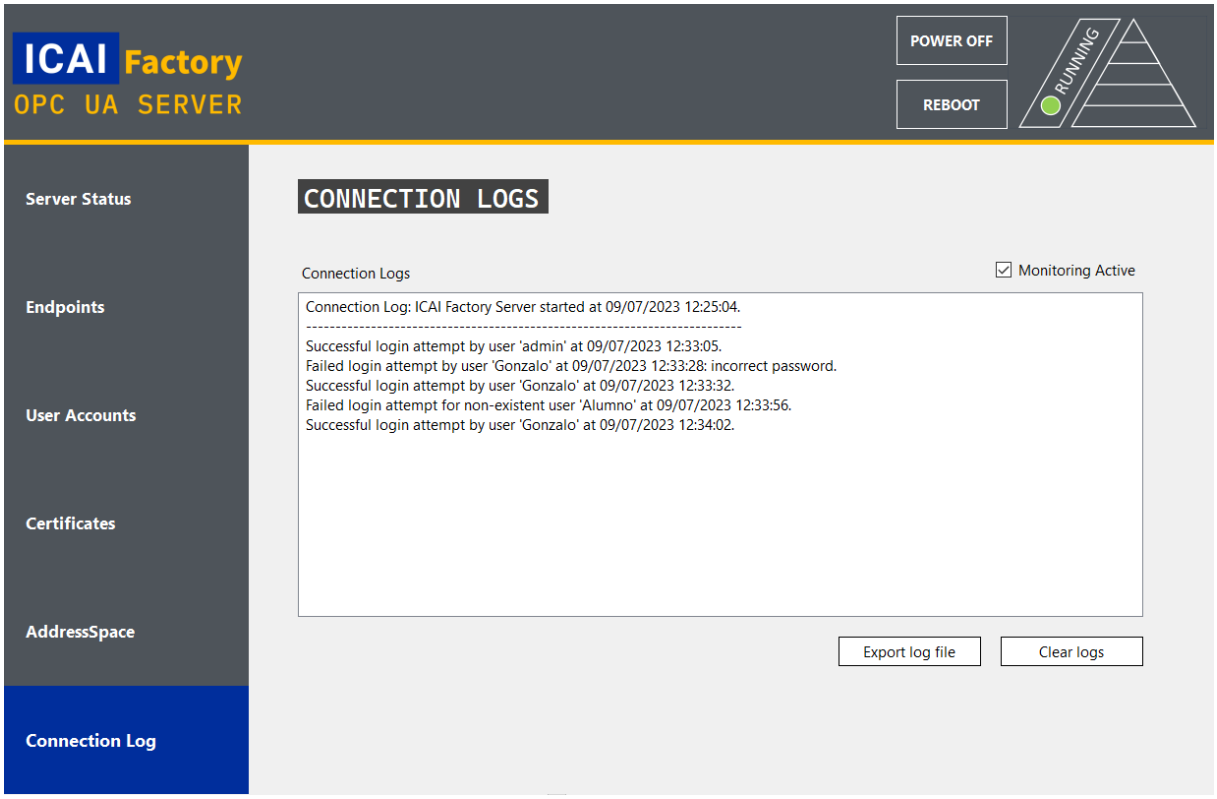

ICAI Factory
OPC UA SERVER
POWER OFF
REBOOT
RUNNING
Server Status
Endpoints
User Accounts
Certificates
AddressSpace
Connection Log
SERVER STATUS
Server Status: Running
Server Starting Time: 09/07/2023 10:25:04 (GMT)
Connection Address: opc.tcp://desktop-7jsdsv6:26543/ICAIFactoryServer
Refresh
Server Configuration
Current settings for the server:
Server Port: 26543
Security Policies: Basic256Sha256
Security Mode: SignAndEncrypt_3
User Settings: UserName_1
Certificate Policy: false
Settings applied after server's reboot:
Server Port: 26543
Security Policies: Basic256Sha256
Security Mode: SignAndEncrypt_3
User Settings: UserName_1
Certificate Policy: false


ICAI Factory
OPC UA SERVER
POWER OFF
REBOOT
RUNNING
Server Status
Endpoints
User Accounts
Certificates
AddressSpace
Connection Log
SERVER ENDPOINTS
Available Server Endpoints:
opc.tcp://desktop-7jsdsv6:26543/ICAIFactoryServer [Basic256Sha256, SignAndEncrypt]
Copy Endpoint URL
Security Mode
None
Sign
Sign and Encrypt
Security Policy
None
Basic128Rsa15
Basic256
Basic256Sha256
Sever Port:
26543
Apply Changes


ICAI Factory
OPC UA SERVER
POWER OFF
REBOOT
RUNNING
Server Status
Endpoints
User Accounts
Certificates
AddressSpace
Connection Log
USER ACCOUNTS
Create new user account
Username *
Password
Confirm Password
Clear
Add new user account
User Settings
Anonymous
Username and Password
User Certificates
Apply Changes
Registered User Accounts:
admin
Gonzalo
Remove user account


ICAI Factory
OPC UA SERVER
POWER OFF
REBOOT
RUNNING
Server Status
Endpoints
User Accounts
Certificates
AddressSpace
Connection Log
CERTIFICATES
Untrusted Certificates
ConsoleClient
[DD834D4BF81F185FF460F58DF39E631CA812EC82]...
Trust >>
<< Untrust
Remove
Refresh
Trusted Certificates
Gonzalo
[BB42C454388CF91A6CD3AFE4B6607DEEB212689
ICAIFactoryServer
[299A6E3040350D498F5EA741B19F562B8C23ED0
NodeOPCUA-Client@DESKTOP-7JSDSV6
[C5BEE14031506B7AD74B5D0ED93ACFCBCCC8D4
NodeOPCUA-Client@iot2050-debian
Certificate Policy
Accept Untrusted Certificates
Apply Changes


ICAI Factory
OPC UA SERVER
POWER OFF
REBOOT
RUNNING
Server Status
Endpoints
User Accounts
Certificates
AddressSpace
Connection Log
ADDRESS SPACE
Address Space of the OPC UA Server
OPCUAServer
MOSystem
XYMD02Sensor1
TemperatureSensor
HumiditySensor
CommunicationProtocol
InputVoltage
XYMD02Sensor2
Factory
Stations
Station1
PreStopper
ProximitySensor
Manufacturer
Type
Expand Nodes
Collapse Nodes


ICAI Factory
OPC UA SERVER
POWER OFF
REBOOT
RUNNING
Server Status
Endpoints
User Accounts
Certificates
AddressSpace
Connection Log
CONNECTION LOGS
Connection Logs
Monitoring Active
Connection Log: ICAI Factory Server started at 09/07/2023 12:25:04.
Successful login attempt by user 'admin' at 09/07/2023 12:33:05.
Failed login attempt by user 'Gonzalo' at 09/07/2023 12:33:28: incorrect password.
Successful login attempt by user 'Gonzalo' at 09/07/2023 12:33:32.
Failed login attempt for non-existent user 'Alumno' at 09/07/2023 12:33:56.
Successful login attempt by user 'Gonzalo' at 09/07/2023 12:34:02.
Export log file
Clear logs

5.3 NOA DIODE

Once the aggregate server has been set up and is fully operational, the next step involves implementing the NOA diode. This diode incorporates all the essential components needed to transfer data from the factory to the aggregate server through a secure and unidirectional connection.

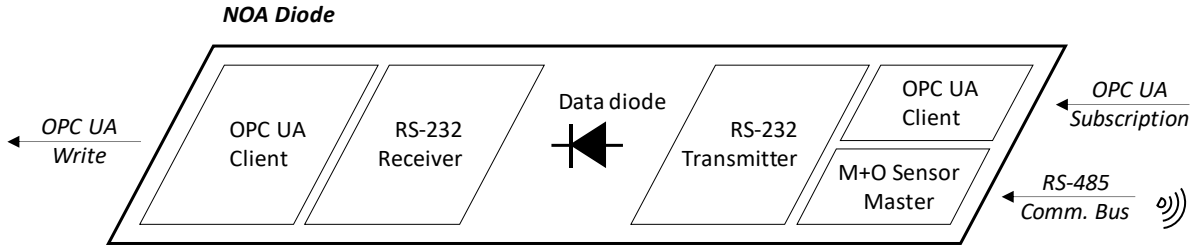


Figure 50. NOA diode diagram (Original work).

Therefore, this chapter is divided into three sections, each covering the development of its main elements:

- OPC UA clients and RS-232 transmitter and receiver.
- Monitoring and Optimisation sensors.
- Physical data diode.

5.3.1 OPC UA CLIENTS

The diagram above highlights the need for two distinct OPC UA client applications:

1. The first client application is responsible for extracting data from the automation pyramid and transmitting it through the data diode using the RS-232 communication protocol.
2. The second client application receives RS-232 messages and writes them into the aggregate server.

To facilitate the development of these clients using Node-RED, it is crucial to download the dedicated OPC UA library: "node-red-contrib-opcua". This library is specifically designed to support OPC UA functionalities within Node-RED and provides the necessary nodes to create server and client applications.

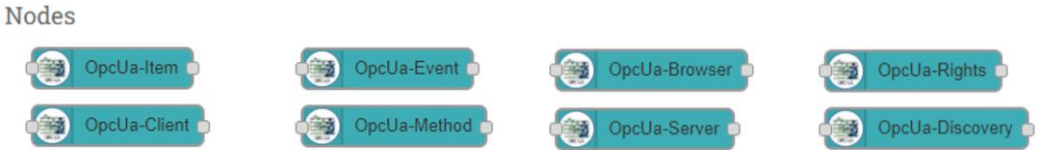


Figure 51. OPC UA nodes (Node-RED, s.f.).

5.3.1.1 OPC UA Client for data extraction / RS-232 Transmitter

To extract data from the factory's PLC and receive notifications whenever a value changes, the first client application utilises an "OPC UA Client" node configured to subscribe to the specific data that needs to be monitored.

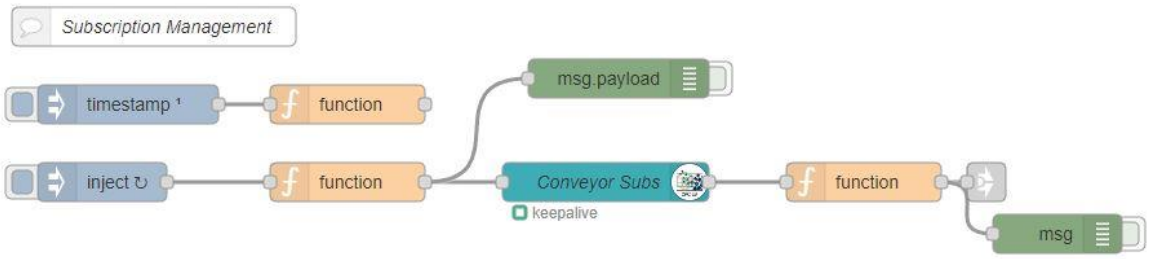


Figure 52. Client subscription to an OPC UA server running on one a factory PLC.

Upon detecting a change in a value, the client application transmits the data through a designated physical port where the RS-232 communication takes place.

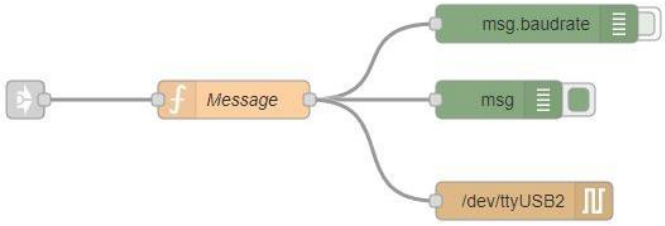


Figure 53. RS-232 Transmitter.

5.3.1.2 RS-232 Receiver / OPC UA Client for writing to the aggregate server

The second client application is responsible for receiving the RS-232 messages and utilises a "OPC UA Client" node to establish a connection with the aggregate server to write the newly received values.

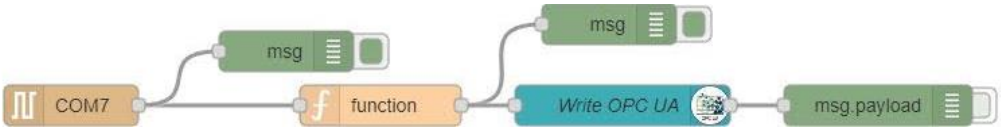


Figure 54. RS-232 Receiver and OPC UA client to write to the aggregate server.

5.3.2 MONITORING AND OPTIMISATION SENSORS

The Monitoring and Optimisation sensors play a vital role in facilitating future predictive maintenance projects and other optimisation initiatives within the factory. They serve as a crucial component for maintaining operational efficiency, reducing downtime, and ensuring the smooth functioning of critical equipment.

To understand the need for an M+O System, it is important to distinguish between two types of data: process data and equipment data.

- Process data is ubiquitous in all factories as it constitutes essential information required for operations and real-time processes. This data is utilised to control process equipment such as valves, heaters, or engines and is integrated into the automation pyramid as a critical component.

- On the other hand, equipment data focuses on understanding the health status of individual machinery and components within the factory. Unlike process data, equipment data is not necessary for immediate process control, but plays a crucial role in predictive maintenance and proactive equipment health management. Thus, by analysing equipment data, machine learning applications can detect signs of wear, identify potential faults, and inform operators to take proactive measures to prevent equipment failures before they occur[9].

To enable timely detection of equipment failure and enhance predictive maintenance efforts, dedicated equipment sensors are essential. These sensors, such as accelerometers, acoustic noise sensors, temperature sensors, or humidity sensors, provide specific data points that help identify signs of deterioration or impending failure.

[9] While process data extracted from plant sensors may also indicate equipment malfunctions, such problems are typically detected once damage has already occurred.

Table 9. Sensor Applications for the M+O System (Berge, 2022c).

Sensor	Reliability	Sustainability	Safety	Production
Vibration	✓			
Ultrasonic Thickness			✓	
Acoustic		✓		
Pressure	✓		✓	✓
Corrosive				✓
Differential Pressure	✓	✓		✓
Contact	✓	✓	✓	✓
Position			✓	✓
Guided Wave Radar				✓
Temperature	✓			
Multi-Temperature (Profile)		✓		✓
Turbine Flow		✓		
Level Switch			✓	✓
Gas Concentration			✓	

To adhere to NOA's principles of preserving the integrity of the automation pyramid, these new sensors must be non-intrusive add-on sensors, and their control mechanism cannot rely on the PLCs running the factory's processes. Instead, an external device capable of independent monitoring must be employed.

For the current project, the M+O System utilises Modbus RS-485 sensors, although it could also incorporate modern protocols such as MQTT, Profinet, HART, Zigbee or other wireless connectivity mechanisms. The utilisation of RS-485 sensors serves as a proof of concept to demonstrate that older equipment can also be included in digitalisation projects.

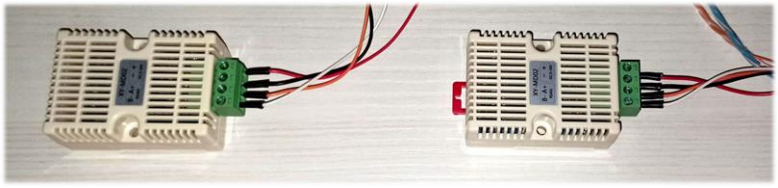


Figure 55. XY-MD02 Modbus temperature and humidity sensors.

The specific equipment consists of two XY-MD02 temperature and humidity sensors. These devices are cheap and suitable as a proof of concept but, for future predictive maintenance initiatives, hundreds or even thousands of different sensors should be added. To control the M+O sensors, the IOT2050 assumes the role of the Modbus master, periodically polling the sensors to retrieve the measured data.

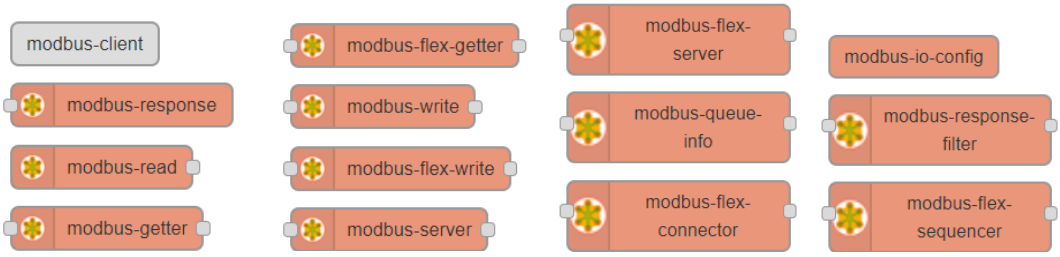


Figure 56. Available nodes provided by the "node-red-contrib-modbus" library (Node-RED, s.f.).

In this scenario, the Modbus master communicates with the RS-485 sensors through a communication bus to which all devices are connected. Upon receiving a message from the Modbus master, all devices connected to the bus process it and, if the ID matches, the designated sensor responds by sending back the requested data. This allows the Modbus master to cyclically retrieve the temperature and humidity measurements from the sensors.

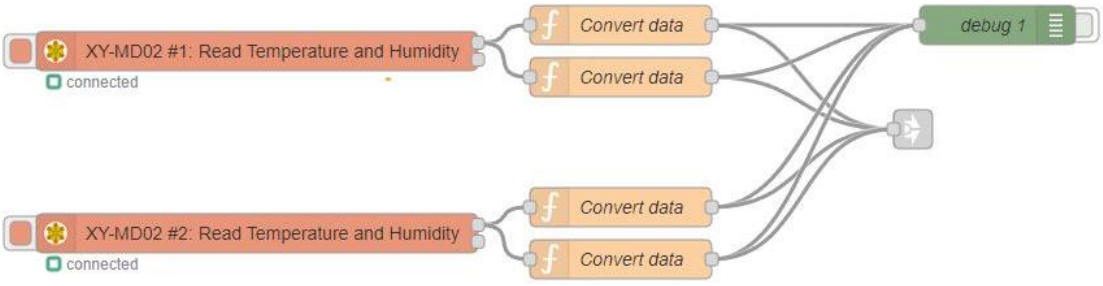


Figure 57:Modbus master program used to poll data from two non-intrusive sensors.

5.3.3 DATA DIODE

Designing and constructing the data diode is essential for implementing Namur's Open Architecture, as it safeguards the automation pyramid and prevents external access to field devices.

When deploying a data diode, it is essential to use connectionless protocols, which excludes OPC UA's TCP as a viable option due to its connection-oriented nature. To overcome this issue, commercial data diodes employ UDP, a unidirectional communication protocol that does not guarantee message delivery or protect against duplicate data packages.

Figure 58. Commercial data diode.

To mitigate the high costs associated with commercial data diodes, an alternative and more cost-effective solution can be implemented using serial communication. As mentioned earlier, this project employs the widely adopted RS-232 protocol.

When employing RS-232 for message transmission, three primary signals must be utilised:

1. Ground: This signal serves to establish a common reference voltage between the transmitting and receiving devices.
2. Tx (Transmit): The Tx pin is responsible for transmitting data from the sending device to the receiving device.
3. Rx (Receive): The Rx pin is used for receiving data from the transmitting device.

To construct a basic data diode, it is sufficient to connect both ground signals and the Tx and Rx terminals of the respective transmitting and receiving devices. By only utilising these two cables, the data diode ensures unidirectional communication, allowing data to flow from the

transmitting to the receiving device while preventing any communication in the opposite direction.

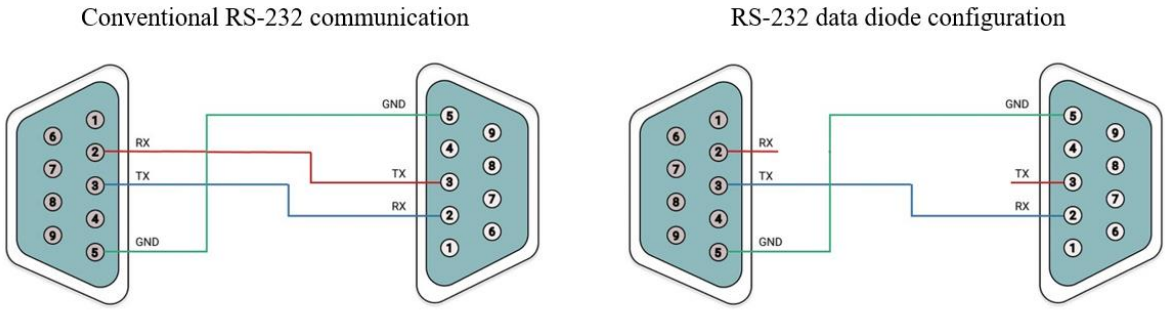


Figure 59. RS-232 conventional vs data diode communication (Original work).

While this configuration prevents data transmission from the receiver to the transmitter, it's important to emphasise that robust cybersecurity data diodes typically necessitate a physical separation between the two devices. This physical separation ensures that data transmission is impossible, regardless of the circumstances.

Commercial data diodes commonly employ fiber optic cables for transmitting UDP signals. However, in this project, an optocoupler[10] will be utilised instead.

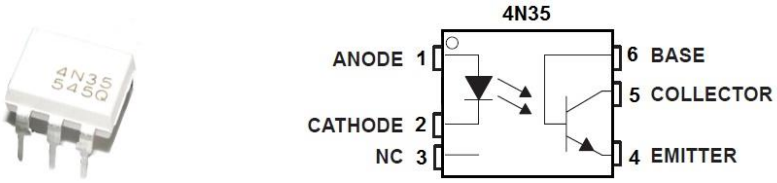


Figure 60. 4N35 Optocoupler.

[10] An optocoupler is a device that physically isolates the transmitting and receiving sides using LEDs and sensors, achieving a similar level of physical separation as fiber optics.

5.3.3.1 Design

Building upon this concept, a dedicated circuit has been meticulously designed to incorporate the optocoupler as the central component of the RS-232 data diode. Rather than utilising a DB9 connector directly, the circuit employs a USB to RS-232 converter to interface with the communication line.

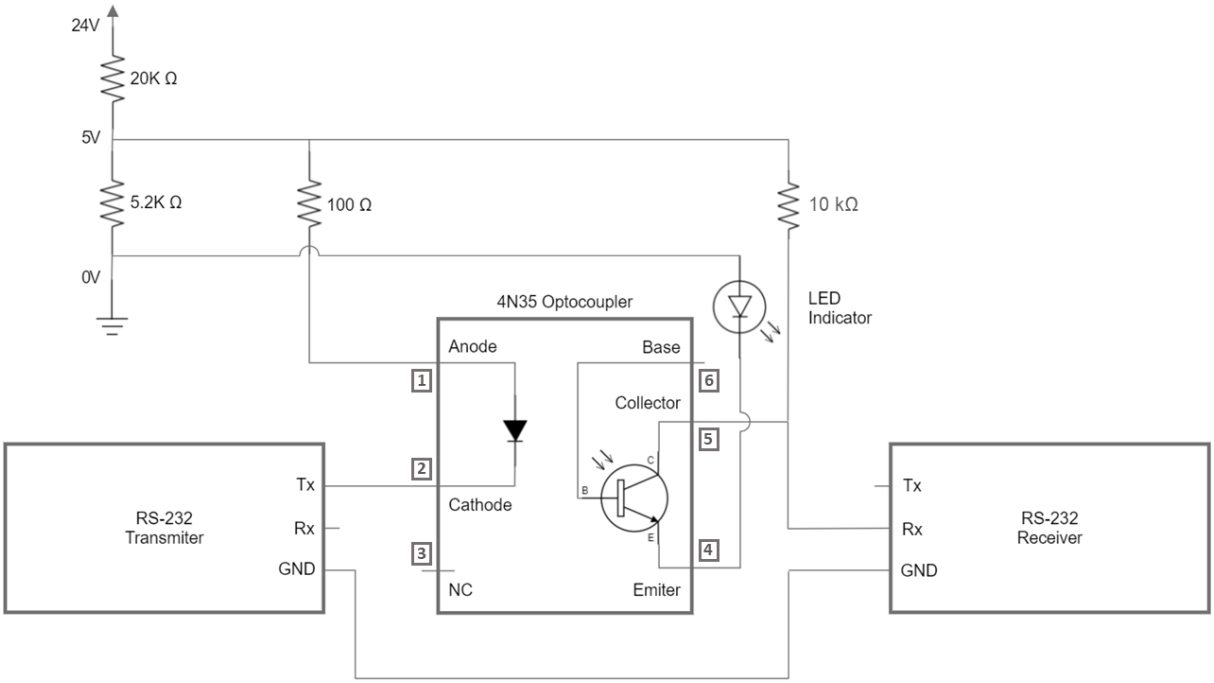


Figure 61. Data diode circuit diagram (Original work).

To comprehend this specific configuration, it is crucial to recognise that the RS-232 converter utilises a particular voltage encoding scheme. In this scheme, a "0" bit is represented by a voltage level of +5 V, while a "1" bit is represented by a voltage level of 0V.

5.3.3.2 Prototyping and Product Assembly

After completing the circuit design, the subsequent stage involves constructing an initial prototype to evaluate the circuit's functionality. For this purpose, a protoboard is employed, enabling the components to be connected and tested.

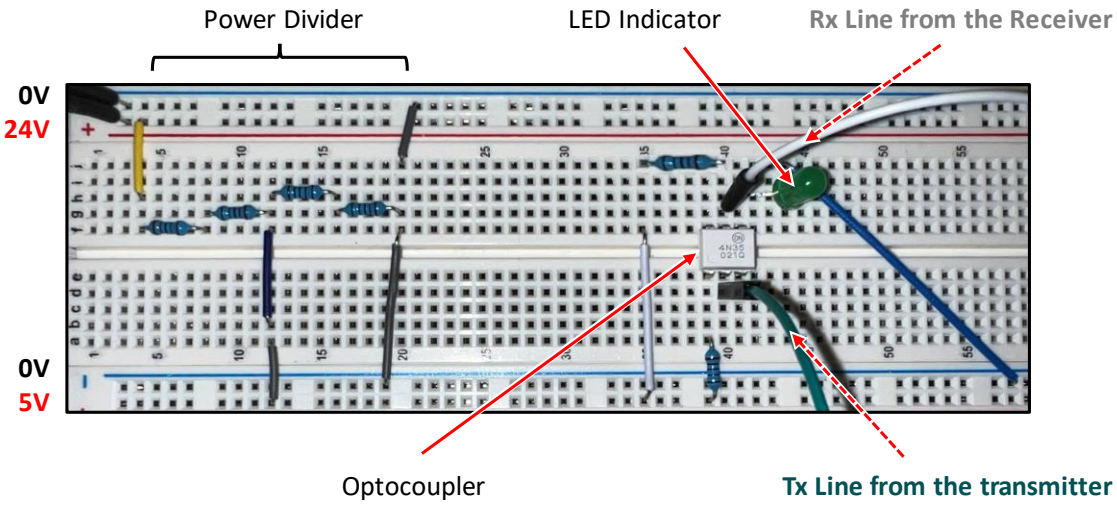


Figure 62. Data diode prototype.

Once the prototype has been successfully validated and meets the desired requirements, development progresses to the assembly of the final product. The final product is assembled using a Printed Circuit Board (PCB) and a specially adapted plastic case designed to be mounted on DIN rails. This feature is particularly significant as it enables the data diode to be conveniently installed alongside the IoT gateway within the factory.

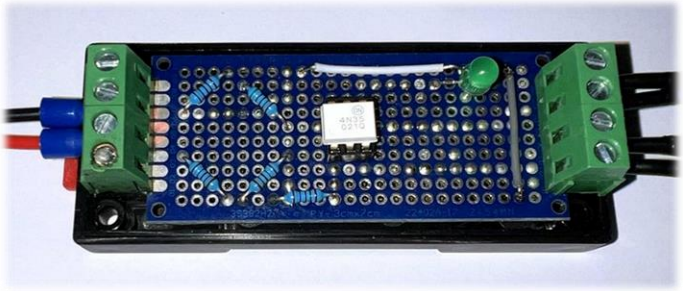


Figure 63. Assembled data diode.

5.3.3.3 Data Diode Limitations

The utilisation of RS-232 communication entails a notable limitation in terms of data bandwidth due to the constraints imposed by the serial protocol. Consequently, in Chapter 7.3.2, a series of tests are conducted to ensure that the current configuration can effectively handle the data transmitted through the diode with ease.

Furthermore, due to the unidirectional nature of the communication, there is no confirmation of whether messages have been received correctly. This means that any messages containing bit errors will not be detected or notified to the transmitting device, resulting in potential data loss. To mitigate this issue, it is necessary to implement measures that reduce the impact of this problem. One approach is to introduce redundancy by sending multiple copies of the same message, increasing the likelihood that at least one of them will be received correctly.

Another limitation to consider is data confidentiality and the risk of false messaging. Since RS-232 communications are sent without encryption, messages can be easily intercepted by connecting a device to the communication line, compromising data confidentiality. Moreover, an attacker could exploit the data diode to send false information, tampering the data sent to the server and undermining its validity.

To address these limitations related to data confidentiality and false messaging, it is crucial to implement encryption. In this case, XOR encryption is employed in conjunction with four private keys that are known only to the IOT2050 (transmitter) and the receiver at the other end.

5.3.3.3.1 XOR Encryption

XOR encryption is a simple cryptographic technique that operates on binary data. It works by applying the XOR operation between the bits of the plaintext message and the bits of a secret key.

To demonstrate XOR encryption, an example is presented using a simple private key and message:

Table 10. Example to illustrate XOR encryption I.

	Plain text	ASCII representation
Private key	key	01101011 - 01100101 - 01111001
Message	hello	01101000 - 01100101 - 01101100 - 01101100 - 01101111

The XOR operation is performed between the bits of the message and the corresponding bits of the private key, resulting in the following encrypted message:

Table 11. Example to illustrate XOR encryption II.

Private key	01101011 - 01100101 - 01111001
Message	01101000 - 01100101 - 01101100 - 01101100 - 01101111
Encrypted Message (XOR)	00000011 - 00000000 - 00010101 - 00010101 - 00010110

In the context of the data diode, four private keys are utilised. This approach is chosen because XOR encryption can be easily deciphered, but by alternating between different private keys, the likelihood of decryption is greatly reduced.

The structure of the encrypted messages transmitted through the data diode is as follows:

1. Two bits indicating which encryption key to use.[11]
2. Encrypted timestamp.
3. Encrypted message.

The timestamp serves as a preventive measure against message copying and replay attacks, where an attacker tries to resend previously intercepted messages. This feature is implemented by the receiver, who keeps a record of the most recent timestamp. Consequently, any messages with timestamps preceding or overlapping with the last received message are discarded.

[11] This part of the message is sent in plain text to specify the encryption key that should be used for decryption. For example, "00" represents key 1, "01" represents key 2, "10" represents key 3, and "11" represents key 4.

The node-RED flow utilised for encrypting and transmitting the messages is as follows.

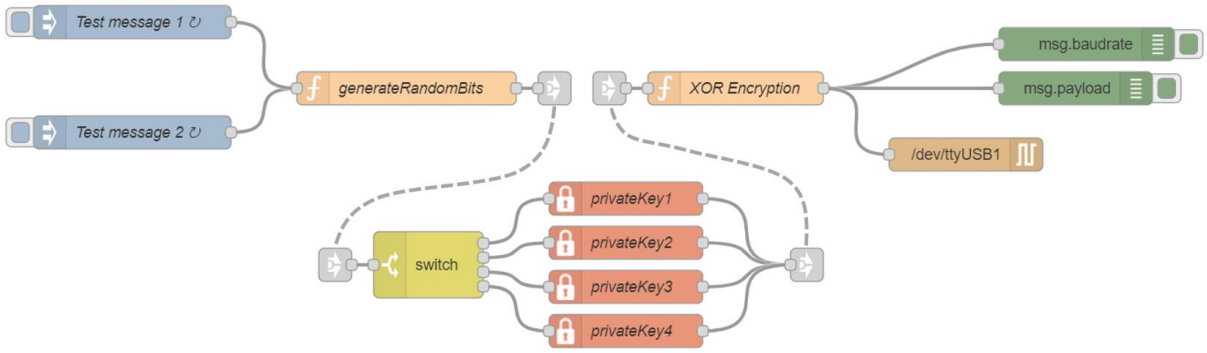


Figure 64. Node-RED flow for XOR Encryption.

The JavaScript code employed by the transmitter to encrypt the messages is provided below.

```
// Message to be encrypted
var message = msg.message;
// Convert the message to an array of ASCII codes
var messageCodes = message.split('').map(char => char.charCodeAt(0));
// Generate a timestamp
var timestamp = Date.now().toString();
// Get the XOR key based on the first two bits
var xorKey = msg.privateKey;

// Perform XOR encryption on each character code with the XOR key
var encryptedCodes = messageCodes.map((code, index) => code ^
xorKey.charCodeAt(index % xorKey.length));
// Convert the encrypted ASCII codes back to characters
var encryptedMessage = String.fromCharCode(...encryptedCodes);

// Encrypt the timestamp
var encryptedTimestamp = '';
for (var i = 0; i < timestamp.length; i++) {
  encryptedTimestamp += String.fromCharCode(timestamp.charCodeAt(i) ^
xorKey.charCodeAt(i % xorKey.length));
}

// Construct the payload with the first two bits, encrypted timestamp, and the
encrypted message
var binaryString = msg.Bits;
var payload = "." + binaryString + encryptedTimestamp + encryptedMessage;
msg.payload = payload;
return msg;
```

5.4 IOT GATEWAY

To successfully deploy this solution in a real-world environment, the implementation process must be carefully considered. In this regard, the IOT2050 Gateway, manufactured by Siemens, serves as the preferred hardware platform to host all software components used to extract data from the factory.

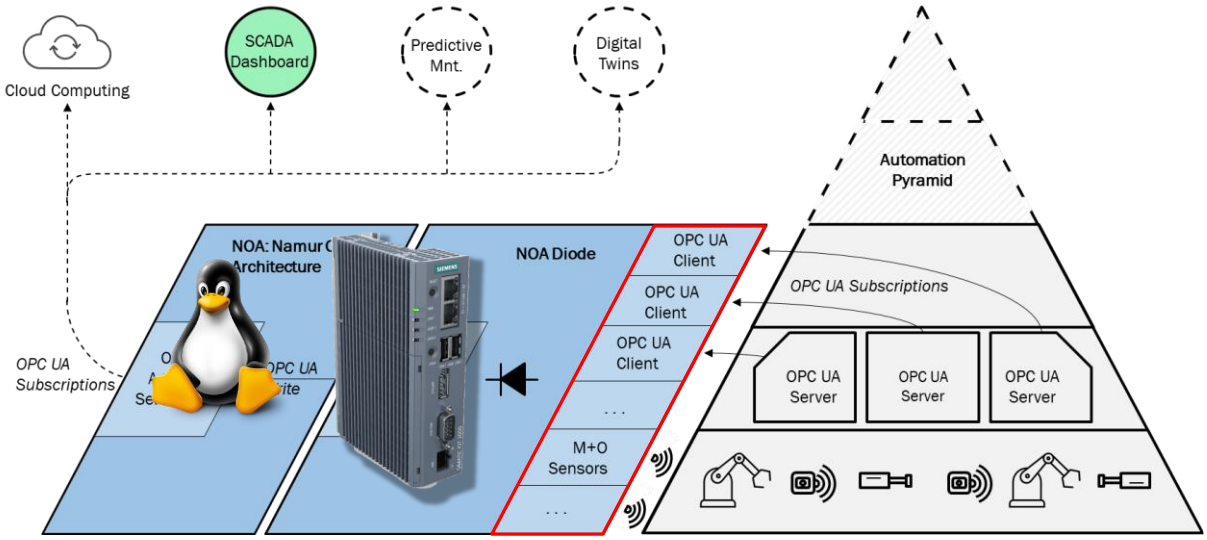


Figure 65. IOT2050 Gateway hosting all components used to extract data from the factory (Original work).

5.4.1 IOT2050 SPECIFICATIONS

The IOT2050 stands as a robust IoT gateway extensively utilised for linking OT equipment with IT networks across diverse industrial settings.

Figure 66. IOT2050 Gateway.

It is equipped with various communication interfaces to enhance connectivity across its supported protocols. These interfaces comprise a serial communication port (COM), a DP port, two USB ports, and two Ethernet ports.

5.4.2 IOT2050 SETUP

To ensure both safety and functionality in ICAI's factory, the IOT gateway is securely placed inside an electrical case mounted on a DIN rail. This enclosure protects the equipment and safeguards users from potential hazards such as electrocution or severe burns, caused by the high temperatures of the gateway's heat sink.

One notable advantage of the setup is that the IOT2050 and the power supply are installed together in a single box. This consolidation of equipment offers the convenience of easy transportation and provides a compact and organised solution. Furthermore, the box includes power supply connectors, that mitigate the risk of electrocution, along with Ethernet ports that allow direct connectivity to the case. This eliminates the need to thread RJ-45 connectors into the gateway ports, ensuring a simplified and efficient setup. Additionally, flexible USB connectors are incorporated to facilitate easy access to the ports of the device.

Figure 67. IOT2050 Gateway installed alongside the power supply inside an electrical box.

Further details regarding the gateway specifications, OS image installation, initial bootup, and configuration are described in Annex V. IOT2050.

5.5 SCADA DASHBOARD

After successfully setting up the aggregate server and the NOA diode, the subsequent task is to develop a Supervisory Control and Data Acquisition (SCADA) dashboard to showcase a real-life application powered by NOA. This dashboard serves as an intuitive and comprehensive interface that allows operators to access and monitor all information stored in the aggregate server. Through the SCADA dashboard, operators can visualise and analyse process and equipment data, providing them with valuable insights for effective decision-making.

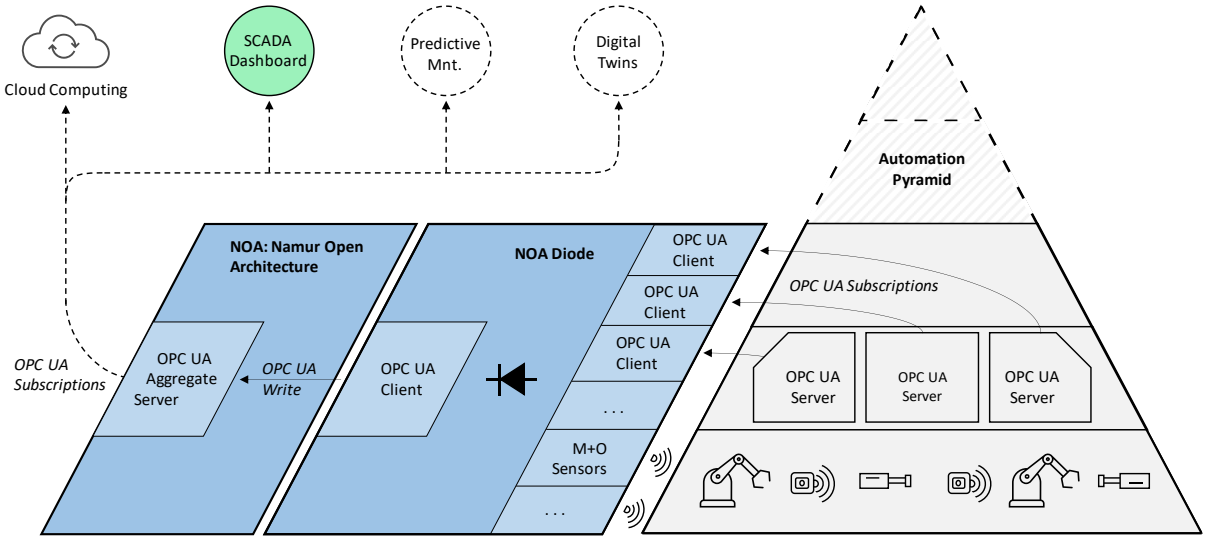


Figure 68. SCADA application depicted in the system's architecture.

Once again, Node-RED is used to develop this dashboard[12], utilising two specific libraries:

1. node-red-dashboard: that enables the creation of dashboards with customisable widgets that can display data in various formats, including charts, tables, and gauges.
2. node-red-contrib-ui-svg: used to display an SVG image of the factory's operating elements, along with animations that reveal their status in the factory.

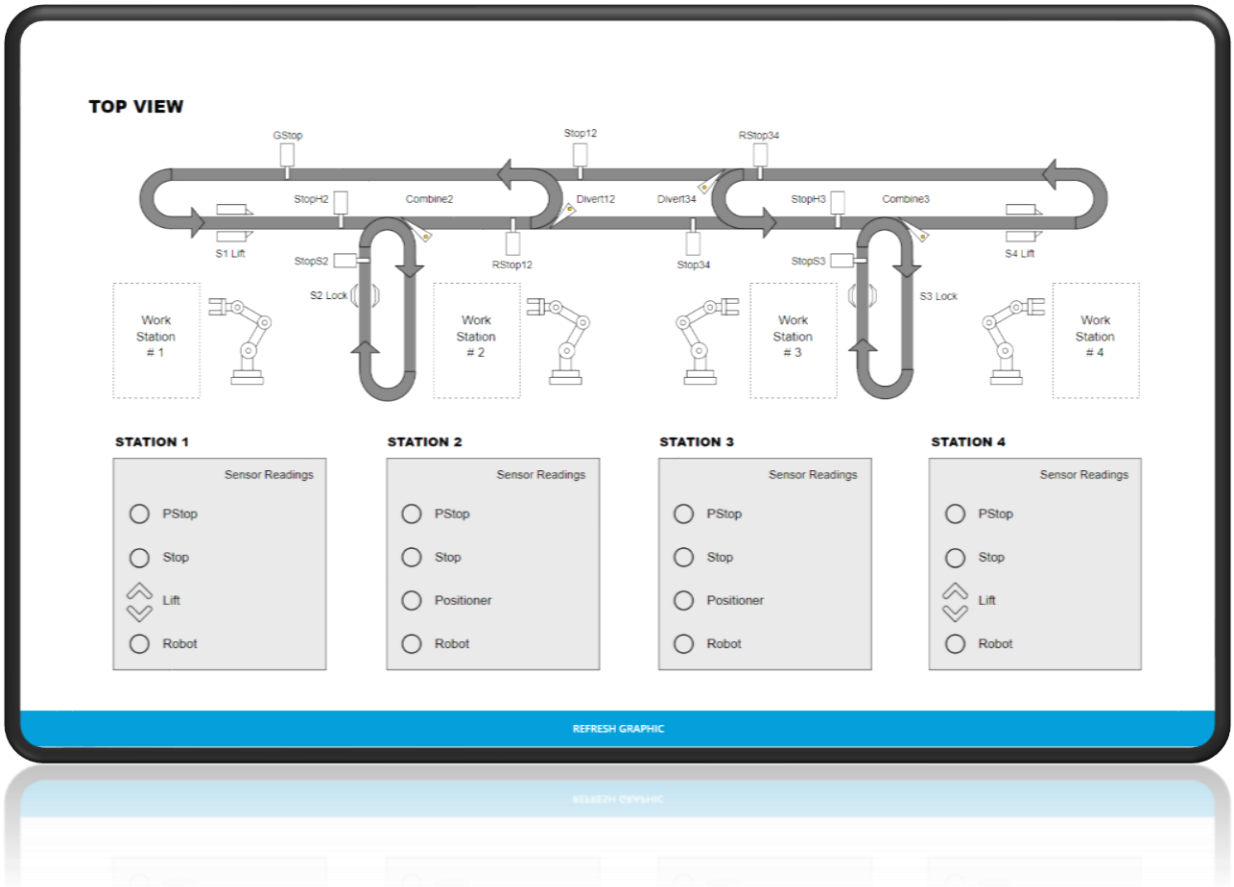
[12] The complete project can be found in Annex VII. Node-RED Dashboard Project.

To interact with the aggregate server and retrieve the information required for visual display, the dashboard utilises an OPC UA client along with a subscription service. This enables it to receive notifications whenever changes occur in the data, ensuring real-time updates.

5.5.1 FACTORY LAYOUT DISPLAY

Utilising the SVG library in Node-RED, a layout display for the factory floor is created based on the provided blueprints (Section 5.1.1). This layout aims to accurately depict the physical arrangement of the factory and its associated hardware. To enhance the visual representation, animations are incorporated into the layout to showcase the movement of pneumatic actuators and the operation of the conveyor control system, specifically the divert and combine needles.

Figure 69. Factory floor layout.



5.5.2 CONTROL PANELS

In addition to the layout display, control panels are designed to provide a summarised and simplified representation of the data. These control panels are organised based on the various subsystems present in the factory, allowing operators to easily filter and access the relevant information as needed.

These panels include:

- Stations: This control panel offers details about the sensors, actuators, and robots present in each station. It allows operators to monitor the status and performance of these components, ensuring smooth operations and identifying any potential issues that may arise.
- Conveyor Belts: The control panel dedicated to conveyor belts provides real-time updates on the status and speed of the three main conveyor belts: the highway, satellite 2, and satellite 3. It also includes information about the general stopper that controls the flow of materials throughout the factory, enabling operators to monitor the conveyor belt system efficiently.
- Diverts and Combines: This control panel presents information about the needles within the conveyor control systems. It also includes details about the stoppers that prevent pallets from colliding with each other and the corresponding control mechanisms.
- Asset Management: The control panel for asset management provides comprehensive information about the assets present in the factory. It includes details such as the manufacturer, payload capacity, maximum speed, asset type, and maintenance schedule. Operators can track and manage the assets effectively, ensuring proper maintenance and optimising their utilisation.
- M+O Sensors: This control panel utilises gauges and a chart to represent temperature and humidity readings from the XY-MD02 sensors installed throughout the factory. Operators can monitor these environmental factors in real-time, allowing for proactive measures to maintain optimal conditions within the facility.

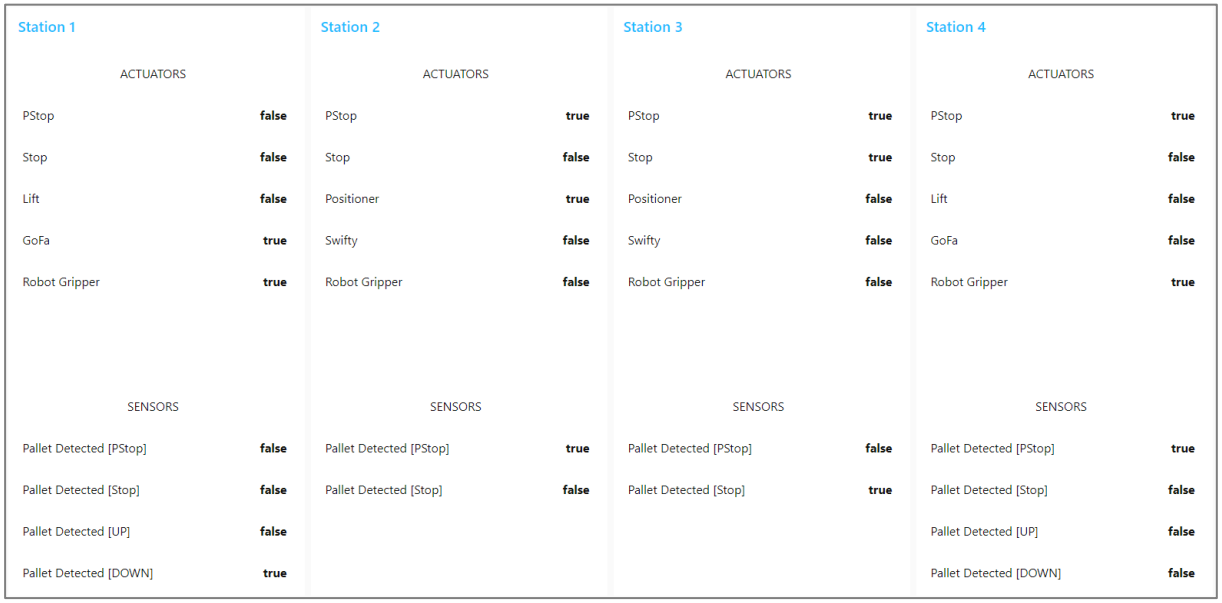


Figure 70. Stations panel.

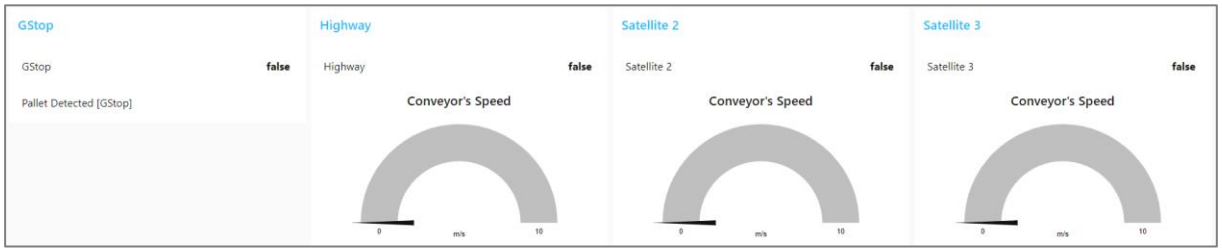


Figure 71. Conveyors panel.

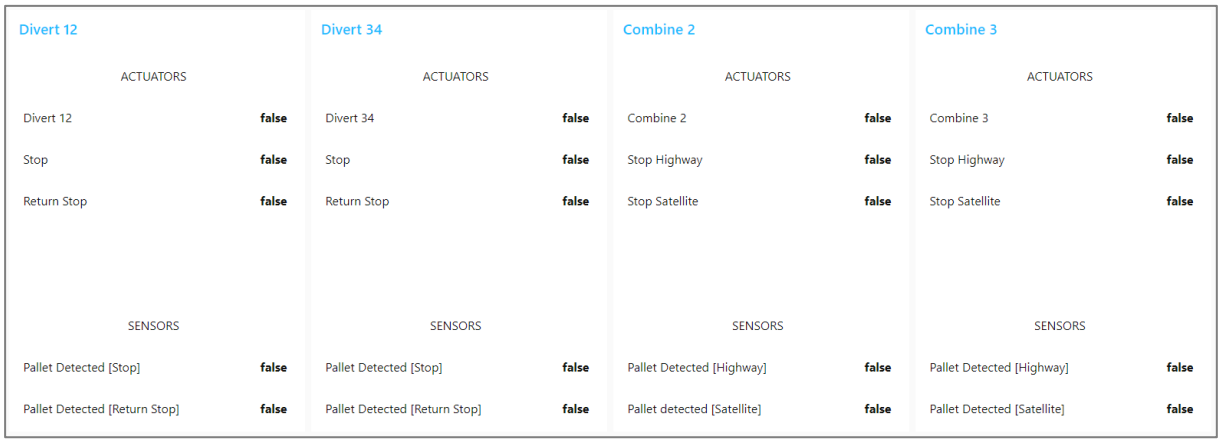


Figure 72. Diverts & Combines panel.

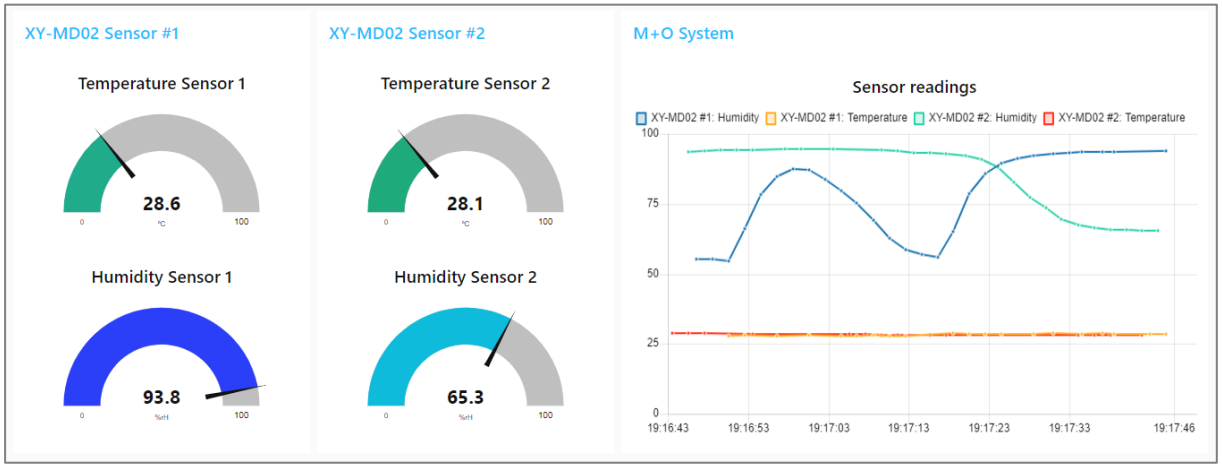


Figure 73. M+O Sensors panel.

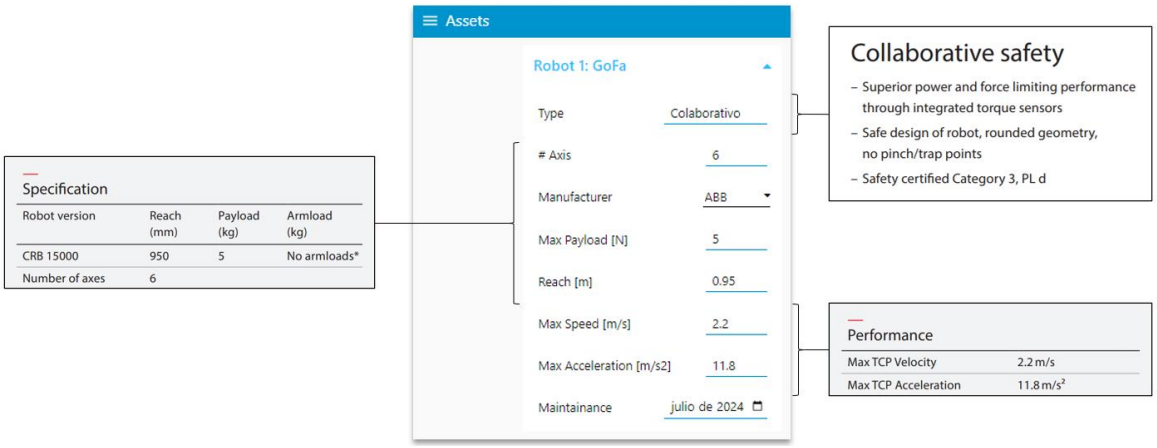


Figure 74. Asset Management panel.

Chapter 6. CYBERSECURITY STUDY

In this final chapter, a comprehensive cybersecurity analysis is undertaken to assess the risks and vulnerabilities confronting ICAI's factory, understanding the impact of a cyberattack on both the factory and the University. To address these concerns, an in-depth analysis is conducted drawing upon the OPSEC and NIST frameworks.

Furthermore, a robust and secure network architecture is designed to assess the security needs of the factory and the new NOA implementation. This process entails incorporating various software tools, such as intrusion detection systems, along with the integration of new hardware devices like firewalls and the previously developed data diode.

6.1 COST ANALYSIS OF A CYBER ATTACK

To comprehend the necessity of these cybersecurity studies, it is imperative to analyse the economic consequences that an attack could inflict upon the University and the factory. Therefore, a simplified cost analysis is conducted to assess the financial implications associated with such cyberattacks.

Firstly, with a focus on the factory, an attack could lead to a complete halt in operations, resulting in a significant economic impact. Assuming the factory produces goods valued at 25€ per unit and has a production capacity of 15 units per hour, a one-hour factory shutdown would result in a loss of 375 €. Extrapolating this to a week of work, accounting for 8 hours per day, the economic losses would amount to 15,000 €.

Additionally, the University's academic activities would also suffer the consequences of an attack. Considering that a class of students, consisting of 20-30 individuals, pays an average of 376 € per hour for lab usage, and assuming the lab is utilised for approximately 7.2 hours per day, the weekly costs would reach 13,536 €. It is important to note that this impact does not encompass the potential losses related to damage to the University's reputation, which could lead to a decline in future student enrolment.

Table 12. Economic impact of hypothetical cyberattacks.

Type of Loss	Cost
Manufacturing goods	15,000 €
Academic lectures	13,536 €
Total	28,536 €

6.2 OPSEC & NIST ANALYSIS

Once the economic reasoning has been established, this section focuses on employing the OPSEC and NIST frameworks to conduct the cybersecurity analysis.

6.2.1 OPSEC ANALYSIS

The analysis begins with the OPSEC (Operations Security) framework, which is designed to identify and safeguard critical information by methodically examining potential threats and vulnerabilities.

The OPSEC framework consists of five phases:

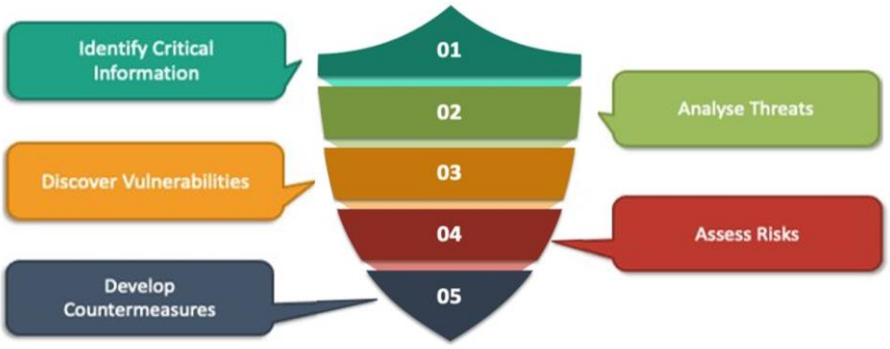


Figure 75. Phases of the OPSEC framework.

6.2.1.1 Identify Critical Information

Firstly, critical information that requires protection must be identified. This information includes key assets, processes, systems, and data that, if compromised, would have a significant impact on operations.

Table 13. Phase 1: Identify critical information.

What are the objectives of the adversaries?	What information do they need to achieve their goal?
▪ Discontinue production. ▪ Damage the reputation of the University. ▪ Steal information and credentials. ▪ Ask for a rescue: Ransomware. ▪ Interrupt academic sessions in the laboratory. ▪ Damage and destroy material.	▪ Network architecture ▪ Physical access to the facility ▪ Usernames and passwords ▪ Software and hardware version and brand

6.2.1.2 Threat Analysis

The second phase involves identifying potential threats that could exploit vulnerabilities and compromise the critical information previously identified.

Table 14. Phase 2: Analyse threats.

Threat sources	Sources of information
▪ Students, teachers, and other employees (aware, unconsciously) ▪ Criminal ▪ Disgruntled former employees ▪ Third party	▪ Moodle ▪ Internet ▪ Trojan horse ▪ Keylogger ▪ Social Media ▪ Post office ▪ Published lists of vulnerabilities (software, hardware, libraries used, ...)

6.2.1.3 Detect Vulnerabilities

Then, vulnerabilities and weaknesses within the organisation's systems, processes, and practices are assessed. This includes evaluating areas where critical information may be susceptible to unauthorised access, exploitation, or disruption.

Table 15. Phase 3: Detect vulnerabilities.

Vulnerabilities	ID
Information on control devices	1
Two-factor authentication	2
Laboratory hours	3
Laboratory access	4
Access to computer classrooms	5
Lack of training	6
Access to the production line	7

6.2.1.4 Assess Risks

After detecting the main threats and vulnerabilities, the next phase involves assessing the risks associated with each one of them. This includes evaluating the likelihood of an attack and the potential impact it could have on both the factory and University.

Table 16. Phase 4: Assess risks.

ID	Vulnerabilities	Likelihood	Impact	Risk
1	Information on control devices	2	3	6
2	Two-factor authentication	3	4	12
3	Laboratory hours	5	1	5
4	Laboratory access	1	5	5
5	Access to computer classrooms	1	4	4
6	Lack of training	5	4	20
7	Access to the production line	1	5	5

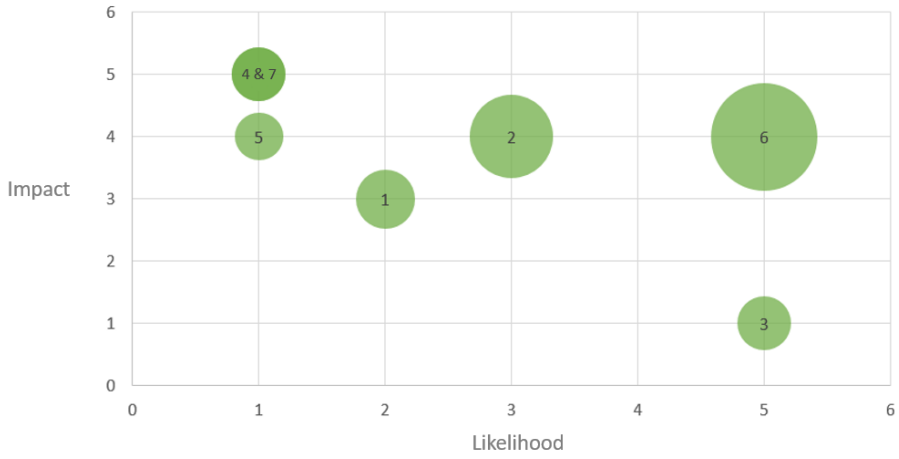


Figure 76. Risk assessment graph.

6.2.1.5 Develop Countermeasures

The final step focuses on designing and implementing countermeasures to mitigate risks and protect critical information.

Table 17: Phase 5: Develop countermeasures.

ID	Risks	Countermeasures
1	Hide device details	▪ Conceal model and version information of devices from the public.
2	Two-factor authentication	▪ Divide the factory into two parts: production and educational. Isolate the production area. ▪ Restrict computer access to registered students only. ▪ Implement multifactor authentication with the use of security tokens.
3	Laboratory hours	▪ Ensure that staff supervises the laboratory during non-teaching hours.
4	Laboratory access	▪ Require individuals to present their University card to gain access to the laboratory.
5	Access to computer classrooms	▪ Restrict remote access to lab classrooms from other devices connected to the same network.
6	Lack of training	▪ Conduct awareness talks for students and workers to enhance their understanding of cybersecurity.
7	Access to the production line	▪ Enforce the requirement of a special permit for accessing the production line outside of class hours.

6.2.2 NIST ANALYSIS

The second framework used to conduct the cybersecurity study is the NIST framework, which provides comprehensive guidelines, standards, and best practices for cybersecurity risk management. It follows a five-step approach—identify, protect, detect, respond, and recover—helping to establish vulnerabilities, detect anomalies, respond to incidents, and recover from disruptions.

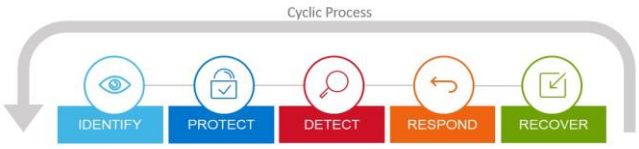


Figure 77. Steps involved in the NIST framework.

Within the NIST framework, two notable guidelines are utilised for this analysis[13]. The first one is the "Manufacturing Profile" of the Cybersecurity Framework, which serves as a roadmap to help manufacturers reduce cybersecurity risks in accordance with industry best practices and sector-specific goals. Additionally, the analysis incorporates the “Ransomware Profile”, which focuses on identifying, protecting, detecting, responding, and recovering from ransomware events.

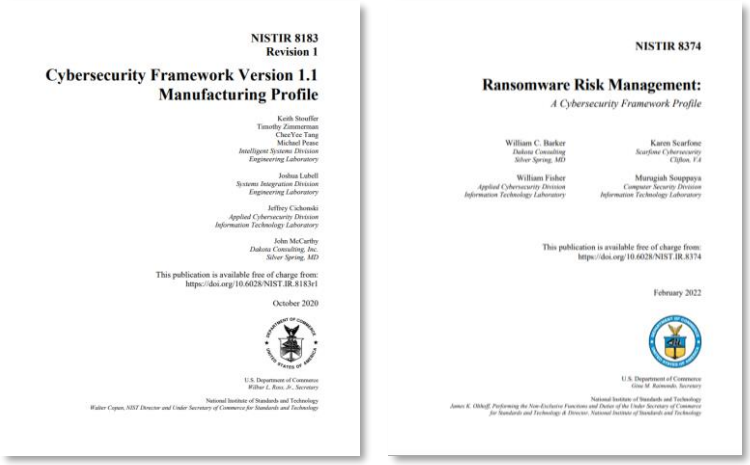


Figure 78. NIST guidelines used for the study.

[13] The guidelines given by the two reports have been highlighted accordingly to their significance in ICAI’s factory and are covered in Annex VI. NIST Study.

6.3 NETWORK TOPOLOGY DESIGN

After completing both studies and obtaining a comprehensive understanding of the vulnerabilities faced by the factory, this section covers the development of a secure network topology that combines Namur's Open Architecture with the unique peculiarities of ICAI's factory. This is necessary to asses the risks faced by the factory due to its location inside a university lab, where other academic activities take place simultaneously.

A schematic diagram illustrates the division of the entire infrastructure into three distinct zones: the factory, the laboratory, and the University. This division serves to clearly delineate the roles of each element and the staff responsible for their future installation and maintenance.

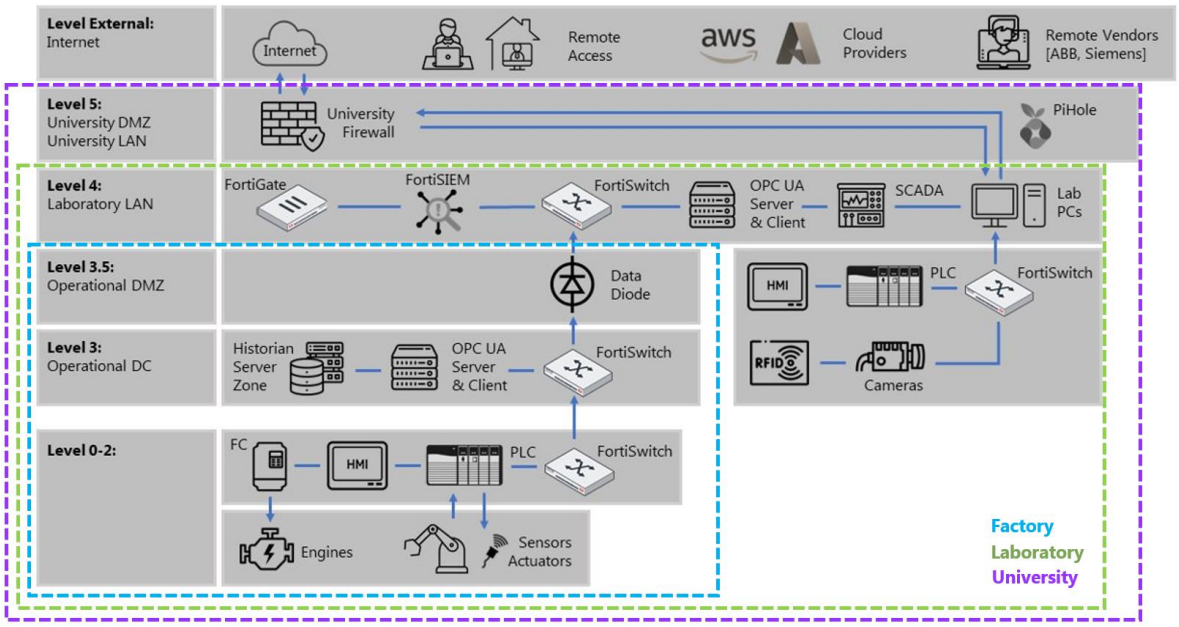


Figure 79. Secure architecture for ICAI's University and factory.

In the University zone, which encompasses the widest range, standard IT hardware such as firewalls and switches are utilised. Additionally, a PiHole has been incorporated as a DNS server to effectively control access to the internet and prevent connections to malicious websites.

The laboratory zone has been fortified with additional security instruments, including firewalls and SIEMs, to strengthen communication security among the internal devices. This area is solely dedicated to academic purposes and houses specific equipment like PCs, PLCs, HMIs, RFID sensors, and Industrial Cameras. Within this zone, the aggregate OPC UA server and the dashboard are situated, serving as resources for students and researchers to undertake and develop further digitalisation projects.

Lastly, the factory area has been isolated from external access by implementing a data diode, allowing for secure data extraction while preventing unauthorised entry. To ensure redundancy and reliability, a second OPC UA server, in conjunction with a Historian Server, have been incorporated.

Chapter 7. RESULT ANALYSIS

This section presents the project results along with a brief discussion of the findings. The most relevant outcomes of the project are analysed, and a critical assessment of them is conducted.

7.1 COMMISSIONING

Firstly, to understand the project's results, all software and hardware components must be installed and running on ICAI's factory. Therefore, this section covers the commissioning phase of the project, describing the steps and tools that are needed.

7.1.1 SOFTWARE

In the implementation stage, the first objective is to convert one of the lab computers into a dedicated host for all NOA elements, including the aggregate server, the dashboard, and the OPC UA client. To achieve this, several software installations are necessary.

7.1.1.1 Version Control

To ensure the smooth operation of NOA components on the designated factory PC, an effective version control system is essential. GitHub serves as a convenient platform for downloading and accessing the latest versions of the project files, allowing for easy management and synchronisation across multiple devices. By utilising GitHub, the project remains up to date, and any future modifications or enhancements can be easily implemented.

7.1.1.2 .NET and Visual Studio

The installation process continues with the OPC UA server. To ensure its proper functioning, the .NET 6.0 framework and Visual Studio's IDE must be installed on the lab's computer. This setup not only enables the execution of the current project but also provides the tools for future improvements, as it grants access to the Visual Studio solution. Once the installation is complete, the project can be imported and executed.

7.1.1.3 Node-RED flows

After the server installation, the subsequent stage involves the implementation of two crucial components: the OPC UA client, responsible for receiving the RS-232 messages and writing it to the aggregate server, and the dashboard.

To run these components Node.js is installed, providing the necessary runtime environment, followed by the installation of Node-RED. Then, the project flows, containing essential configurations and logic, are imported into the computer. By executing the appropriate commands, both Node-RED projects can run simultaneously on different ports.

```
node-red projects/dashboard -p 1880
```

```
node-red projects/rs232_Receiver -p 1881
```

7.1.2 HARDWARE

Hardware implementation primarily focuses on the IOT2050 Gateway, which serves as the bridge connecting the factory's PLCs to the newly prepared PC hosting the NOA system.

Since the IOT gateway has been previously programmed, its installation in the factory primarily involves connecting it to power and using an Ethernet cable to establish a connection with the factory's LAN.

Figure 80. IOT2050 Gateway commissioning.

To physically connect the gateway and the PC, the data diode and two USB to RS-232 converters are also added[14].

Figure 81. Data diode and USB to RS-232 converter.

Once the connection between the gateway and the computer via the data diode is established, the final step involves installing the hardware designed for the M+O sensors. Due to limited space inside the electrical box, using a COM port adapter was not a viable option. However, a USB to RS-485 converter proves to be a suitable alternative, effectively addressing the space constraint.

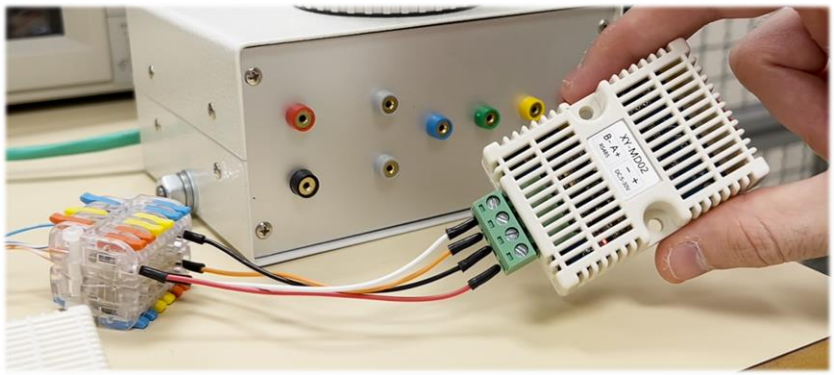


Figure 82. RS-485 communication bus with XY-MD02 temperature and humidity sensor.

[14] USB converters are chosen over serial COM ports to ensure compatibility with laptops that lack built-in DB-9 ports.

7.2 RESULTS

Once all software and hardware components are installed and operational, the project results can finally be examined. This subsection presents the objective findings and achievements of the project, demonstrating its progress towards its intended goals.

During the technology study and selection phase, a thorough evaluation was conducted to identify and implement the most fitting technologies for the factory. The achievement of this goal is confirmed by the adoption of the .NET framework and the OPC Foundation's SDK for server development, alongside the integration of Node-RED as a low-code tool for client construction, dashboard design, and setting up of the M+O sensors. Furthermore, the selection of the IOT2050 Gateway as the preferred hardware solution to establish connectivity proves the successful utilisation of suitable cutting-edge technology for the development of the project.

Regarding the implementation of NOA, the factory successfully incorporates the principles and concepts established by Namur's Open Architecture. To achieve this, the placement of the data diode was fundamental to extract the data securely, while the creation of a sophisticated information model has offered a holistic view of the factory, enabling a clear understanding of the contextual situation for each individual data point. This powerful combination of secure data extraction and advanced information model has unlocked unlimited possibilities for new digitalisation projects and the implementation of groundbreaking technologies into the factory.

Furthermore, the successful integration of non-intrusive sensors and the configuration of the Siemens IOT2050 Gateway as a Modbus master, have laid the foundation for the Monitoring and Optimisation System. Although this initial implementation involves only two sensors, it serves as a valuable proof of concept for future studies on predictive maintenance and other optimisation initiatives.

The design of the SCADA dashboard, with its accurate graphical representation of the factory's elements, serves as an effective monitoring and decision-making tool, providing operators with real-time insights into the factory's operations. Additionally, it offers a comprehensive display

of the factory's status, allowing for the identification of potential issues and proactive measures to optimise performance.

Lastly, the project has incorporated industry-leading security frameworks such as OPSEC and NIST to establish a robust security foundation. Additionally, the implementation of the data diode has played a crucial role in enhancing the security posture of the system.

In conclusion, the project's results showcase the successful achievement of each objective and represent a significant milestone in ICAI's digital transformation journey. The implementation of the NOA architecture and utilisation of the OPC UA standard have set the foundations for future projects, enabling ICAI to embrace new technologies, optimise operations, and unleash the full potential of a connected and data-driven factory environment.

Figure 83. Project's results (Original work).

7.3 DISCUSSION AND ANALYSIS

After assessing the overall results, this section highlights any limitations or challenges encountered during the project. Furthermore, potential areas for improvement or further investigation are discussed.

7.3.1 SERVER LIMITATIONS

During the project's commissioning, limitations related to the server were identified. Problems arose when trying to connect to the OPC UA aggregate server from other machines in the same network. While this limitation can be seen as positive security measure, as it restricts access to the server, it also limits the system's usability. Currently, connecting to the server from a different computer requires conducting an SSH tunnel and port forwarding. Therefore, exploring the reasons behind this connectivity issue is of great importance to maximise the potential of the aggregate server.

7.3.2 DATA DIODE CHALLENGES

The use of a data diode for secure communication introduced challenges in the exchange of data. A prominent example of such challenges arises when the aggregate server necessitates an update of all variables within the factory's PLCs. In this scenario, the sequencing of instructions cannot be directly ordered by either the dashboard or the server. Instead, it becomes imperative for an operator to undertake this task by accessing the IoT gateway positioned prior to the data diode.

Furthermore, the limited bandwidth of the RS-232 communications poses a scalability constraint. As the factory and M+O sensor system will progressively grow, the increased data volume may surpass the capacity of the data diode to handle it effectively. This limitation could hinder the system's ability to accommodate future expansion and the integration of additional sensors. Therefore, it is crucial to consider this issue and assess the need for upgrading the data diode to a commercial-grade solution that can handle higher volumes of data.

7.3.3 CYBERSECURITY CONSIDERATIONS

An essential aspect of the project execution has involved conducting in-depth studies using the OPSEC and NIST frameworks to assess cybersecurity risks and develop a secure network architecture. However, these studies primarily focused on theoretical approaches, and no actual changes were implemented in the factory. While theoretical understanding is essential, practical implementation and real-world testing are critical to ensure effective security measures. Consequently, future efforts should prioritise the implementation of the recommended security measures to mitigate potential vulnerabilities and protect the factory's operations.

7.3.4 FACTORY DEPLOYMENT AND TESTING

Finally, it is important to note that the project's tests and results only apply to a small part of the factory that was tested. The factory is not fully deployed, which limits the scope of the findings. Thus, further testing and validation on a larger scale is necessary to ensure the effectiveness and reliability of the implemented solution.

In conclusion, while the project has achieved significant milestones and demonstrated successful outcomes, several areas require attention and continuous improvement to maximise its impact. Enhancing the flexibility of communication exchange, implementing practical cybersecurity measures, improving accessibility to the OPC UA server, and conducting comprehensive testing on a larger scale are crucial for optimising the project's effectiveness and adaptability. By addressing these areas and emphasising a culture of continuous improvement, the project will unlock the full potential of advanced technologies and connectivity in the factory, resulting in improved operational efficiency, informed decision-making, and a future-ready environment for growth and innovation.

Chapter 8. CONCLUSIONS AND FUTURE WORK

8.1 CONCLUSION

In conclusion, the project successfully achieved its main objective of deploying the NOA architecture in the ICAI factory. This deployment has facilitated the digitalisation and integration of the factory into the connected era while ensuring the preservation of its security and original integrity. Through the implementation of OPC UA servers and clients, M+O sensors, an IoT gateway with a data diode, and a SCADA-like dashboard, the project has enhanced operational capabilities, improved efficiency, and enabled real-time data-driven decision-making.

These accomplishments have set the bases for future digital transformation and optimisation, positioning the factory for upcoming advancements in areas such as digital twins, predictive maintenance, big data analysis, and cloud computing. Thus, the project's success underscores the commitment to modernise the factory and embrace the opportunities of Industry 4.0, ultimately driving innovation and competitiveness in its operations.

8.2 FUTURE WORK

Based on these notable achievements, ICAI's factory presents several promising areas for future advancements and developments.

One such area is the implementation of digital twins, which serve as virtual replicas of physical assets. Digital twins offer the ability to simulate and monitor machine and process performance in real-time, thereby empowering proactive maintenance and optimisation strategies within the factory.

Predictive maintenance is another area that holds great potential. Through advanced analytics and machine learning algorithms, predictive maintenance can anticipate equipment failures and

schedule maintenance activities accordingly. This proactive approach maximises uptime, minimises maintenance costs, and optimises the utilisation of resources within the factory.

In the field of big data, ICAI can leverage the abundant volume of data generated by diverse systems and sensors within the factory, enabling the acquisition of valuable insights and facilitating data-driven decision-making. Making use of ICAI's cluster, numerous projects can be developed to further explore and analyse this data.

Furthermore, the secure connection to the internet, unlocked by NOA, enables the utilisation of cloud computing resources, such as scalable storage, computational power, and advanced analytics.

Chapter 9. BIBLIOGRAPHY

ABB. (2022). *Module Type Package (MTP).* Retrieved from Module Type Package (MTP): https://new.abb.com/control-systems/modular-automation/module-type-package

Anticyclone Systems. (7 de january de 2023). *An introduction to RS232 serial port communication*. Obtenido de Anticyclone Systems: https://www.anticyclone-systems.co.uk/rs232.php

Asghar, M. R., Hu, Q., & Zeadally, S. (8 de october de 2019). *Cybersecurity in industrial control systems: Issues, technologies, and challenges.* Obtenido de Google Scholar: https://www.sciencedirect.com/science/article/pii/S1389128619306292

Berge, J. (2017a, july 23). *Digital Transformation - What it Actually Means for a Plant.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/digital-transformation-what-actually-means-plant-jonas-berge

Berge, J. (2017b, december 19). *Starting Digital Transformation - Threat and Opportunity of the New Industrial Revolution.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/starting-digital-transformation-threat-opportunity-new-jonas-berge

Berge, J. (2018a, september 11). *Virtual Reality for Accelerated Field Operator Learning and More.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/virtual-reality-accelerated-field-operator-learning-more-jonas-berge

Berge, J. (2018b, october 21). *Standards for Automation - No Regrets Digital Transformation.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/standards-automation-regrets-digital-transformation-jonas-berge

Berge, J. (2019a, november 20). *Implementing the NAMUR Open Architecture (NOA).* Retrieved from LinkedIn: https://www.linkedin.com/pulse/implementing-namur-open-architecture-noa-jonas-berge

Berge, J. (2019b, november 27). *If it won't break, don't sense it.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/wont-break-dont-sense-jonas-berge

Berge, J. (2020a, march 10). *Enterprise OT - Structured Integration for Digital Transformation.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/enterprise-ot-structured-integration-digital-jonas-berge

Berge, J. (2020b, may 05). *Implementing HTML 5 in Plants - Web Technology User Guide.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/implementing-html-5-plants-web-technology-user-guide-jonas-berge

Berge, J. (2021a, march 4). *Steer Your DX Project Through the Gate Process.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/steer-your-dx-project-through-gate-process-jonas-berge

Berge, J. (2021b, october 14). *7 Standards for HyperAutomation.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/7-standards-hyperautomation-jonas-berge

Berge, J. (2021c, december 27). *#BigIdeas2022 - Survive & Thrive HyperAutomation.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/bigideas2022-survive-thrive-hyperautomation-jonas-berge

Berge, J. (2022a, march 10). *Architecture: Flat or Structured ISA 95/Purdue.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/architecture-flat-structured-isa-95purdue-jonas-berge

Berge, J. (2022b, july 26). *Reliability Risks Under Control with Automation.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/reliability-risks-under-control-automation-jonas-berge

Berge, J. (2022c, september 20). *M+O Sensors: Monitoring & Optimization.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/mo-sensors-monitoring-optimization-jonas-berge

Berge, J. (2022d, october 10). *The Technology Freedom Paradox.* Retrieved from LinkedIn: https://www.linkedin.com/pulse/technology-freedom-paradox-jonas-berge

Csanyi, E. (7 de december de 2016). *9 Rules for correct cabling of the Modbus RS485 communication systems*. Obtenido de Eectrical Eineering Portal: https://electrical-engineering-portal.com/correct-cabling-modbus-rs485

Digitalindustrie, D. E.-u. (2021). *NOA – Mastering the complexity of connectivity*. Retrieved from YouTube: https://www.youtube.com/watch?v=iVC_9qriM5A&t=172s

González, I. J., & BorgesRivero, R. A. (2022). *Free software HMI application developed on Python to monitoring a hydraulic subsystem at the pediatric hospital La Balear of Havana.* Obtenido de RENIA: https://renia.cujae.edu.cu/index.php/renia/article/view/38

Industry40tv. (2020a, september 2). *What is OPC UA and How it Works?* Retrieved from YouTube: https://www.youtube.com/watch?v=vRk42W_4R0o&list=PLIrJJXPVFRvjHsA9tta8yULOB8nPUO_G7

Industry40tv. (2020b, september 15). *OPC UA Information Model - How an OPC UA Information Model Works*. Retrieved from YouTube: https://www.youtube.com/watch?v=f5oPEVhZFug&list=PLIrJJXPVFRvjHsA9tta8yULOB8nPUO_G7&index=2

Industry40tv. (2020c, october 19). *Understanding OPC UA Base Information Model and Companion Specifications*. Retrieved from YouTube: https://www.youtube.com/watch?v=cL5Tq7a1gwo&list=PLIrJJXPVFRvjHsA9tta8yULOB8nPUO_G7&index=3

Industry40tv. (19 de enero de 2021a). *OPC UA .NET Tutorial - Creating Information Model and OPC UA Server Using NET*. Obtenido de YouTube: https://www.youtube.com/watch?v=gxA7SDNLHgc&list=PLIrJJXPVFRvjHsA9tta8yULOB8nPUO_G7&index=4

Industry40tv. (2021b, june 10). *How OPC UA Client Server Communication Works*. Retrieved from YouTube: https://www.youtube.com/watch?v=vgE9P6KNC7g&list=PLIrJJXPVFRvjHsA9tta8yULOB8nPUO_G7&index=5

Industry40tv. (2021c, october 13). *OPC UA Technology Mapping: Data Encoding, Data Security and Transport Protocols*. Retrieved from YouTube: https://www.youtube.com/watch?v=zYjKK0F3RDg&list=PLIrJJXPVFRvjHsA9tta8yULOB8nPUO_G7&index=6

Industry40tv. (2 de agosto de 2022). *OPC UA PubSub: What it is and How it Works?* Obtenido de YouTube: https://www.youtube.com/watch?v=eTAK9noFHWE&list=PLIrJJXPVFRvjHsA9tta8yULOB8nPUO_G7&index=7

Inmation. (2021). *OPC Unified Architecture*. Retrieved from Inmation: https://docs.inmation.com/system/1.90/opc-connectivity/opc-unified-architecture.html

Mahnke, W., Leitner, S.-H., & Damm, M. (2009). *OPC Unified Architecture.* Springer.

Matyas, K., Nemeth, T., Kovacs, K., & Glawar, R. (27 de april de 2017). *A procedural approach for realizing prescriptive maintenance planning in manufacturing industries.* Obtenido de Google Scholar: https://www.sciencedirect.com/science/article/pii/S0007850617300070

Mondejar, J. A. (2023). Automatización Avanzada. ICAI Comillas.

NAMUR Open Architecture (NOA). (2021, december 5). Retrieved from Samson: https://www.samsongroup.com/es/actualidad/actualidad/detalles/news/fachwissen/namur-open-architecture/

NIST. (s.f.). *Cybersecurity Framework: Manufacturing Profile.*

NIST. (s.f.). *Cybersecurity Framework: Ransomware Profile.*

Node-RED. (s.f.). Obtenido de https://flows.nodered.org/

OPC Foundation. (2023). *UA-ModelCompiler*. Obtenido de GitHub: https://github.com/OPCFoundation/UA-ModelCompiler

Piromalis, D., & Kantaros, A. (7 de july de 2022). *Digital Twins in the Automotive Industry: The Road toward.* Obtenido de MDPI: https://www.mdpi.com/2571-5577/5/4/65

Pontarolli, R. P., Bigheti, J. A., Sá, L. B., & Godoy, E. P. (27 de january de 2023). *Microservice-Oriented Architecture for Industry 4.0.* Obtenido de MDPI: https://www.mdpi.com/2673-4117/4/2/69

Represa, J. G., Larrinaga, F., Varga, P., Ochoa, W., Perez, A., Kozma, D., & Delsing, J. (2023, january 31). *Investigation of Microservice-Based Workflow Management Solutions for Industrial Automation.* Retrieved from MDPI: https://www.mdpi.com/2076-3417/13/3/1835

Rodríguez-Morcillo, C., Alexandres, S., Matanza, J., & Adán, E. M. (2022). *Comunicaciones Industriales.* Madrid: ICAI Comillas.

SIEMENS. (2021, october). *SIMATIC IOT2050*. Retrieved from RS Componentes : https://my.rs-online.com/web/p/iot-gateways/2017732

Weis, O. (17 de december de 2019). *Interfaz de comunicación serie. Pinout RS232.* Obtenido de Virtual Serial Port: https://www.virtual-serial-port.org/es/article/what-is-serial-port/rs232-pinout/

Zonta, T., Costa, C. A., Righi, R. d., Lima, M. J., Trindade, E. S., & Li, G. P. (2020, october 6). *Predictive maintenance in the Industry 4.0: A systematic literature review.* Retrieved from Google Scholar: https://www.sciencedirect.com/science/article/pii/S0360835220305787

Chapter 10. ABBREVIATIONS

IT: Information Technologies

OT: Operation Technologies

NOA: Namur Open Architecture

O-PAS: Open-Process Automation Standard

OPC UA: OPC Unified Architecture

SCADA: Supervisory Control and Data Acquisition

SDK: Software Developer Kit

API: Application Programming Interface

HTTP: Hypertext Transfer Protocol

SOAP: Simple Object Access Protocol

TCP: Transmission Control Protocol

UDP: User Datagram Protocol

MQTT: Message Queuing Telemetry Transport

COM: Component Object Model

DCOM: Distributed Component Object Model

IoT: Internet of Things

IIoT: Industrial Internet of Things

DDoS: Distributed Denial of Service

UART: Universal Asynchronous Receiver-Transmitter

SSH: Secure Shell

JSON: JavaScript Object Notation

SVG: Scalable Vector Graphics

OPSEC: Operations Security

NIST: National Institute of Standards and Technology

DNS: Domain Name System

SIEM: Security Information and Event Management

PLC: Programmable Logic Controller

HMI: Human-Machine Interface

RFID: Radio Frequency Identification

DIN: Deutsches Institut für Normung (German Institute for Standardization)

WBS: Work Breakdown Structure

Chapter 11. GLOSSARY

Cloud Computing: The delivery of computing services, including storage, processing power, and software applications, over the internet. It allows users to access and utilise resources on-demand from remote servers, eliminating the need for local infrastructure and providing scalability and flexibility.

Big Data: Large and complex sets of data that cannot be effectively managed or analysed using traditional data processing methods. Big data typically includes vast amounts of information from various sources, such as social media, sensors, and transactions, requiring advanced tools and techniques for extraction, storage, and analysis.

Digital Twins: Digital twins are virtual replicas or models of physical objects, systems, or processes that simulate their behaviour, characteristics, and performance in real-time. They are created by collecting data from sensors, IoT devices, and other sources to create a digital counterpart that mirrors the physical entity. Digital twins can be used for predictive maintenance, process optimisation, scenario analysis and simulation.

SCADA: System used to remotely monitor and control industrial processes and collect data in real-time for analysis and management. SCADA systems provide operators with a graphical interface to visualise and manage industrial processes, enabling them to monitor equipment status, adjust settings, and receive alarms or notifications.

Software Developer Kit (SDK): An SDK is a set of software tools and libraries that developers use to create applications for specific platforms or systems. It provides a framework and resources to simplify and streamline the development process.

DDoS: Malicious attack aimed at disrupting the normal functioning of a network or website by overwhelming it with a flood of traffic from multiple sources. It makes the targeted service unavailable to legitimate users.

ANNEX I. ALIGNMENT WITH SDGS

The digitalisation of industrial processes, including the implementation of the NOA architecture in ICAI's factory, aligns with the following Sustainable Development Goals (SDGs):

SDG 3: Good Health and Well-being

In manufacturing processes, many manual tasks are necessary to maintain plant operation, such as maintenance work. These duties, often known as "4D jobs", are characterised by being Dull (manual collection of parameters), Dirty, Dangerous, and Distant (they require workers to travel to remote locations where the factories are located). Thus, digitalisation seeks to eliminate these "4D" jobs and replace them with safer, less monotonous work that can be done remotely.

SDG 8: Decent Work and Economic Growth

Digitalisation of industrial processes can lead to the creation of new jobs and better working conditions, particularly as it relates to eliminating dangerous, tedious, and monotonous tasks. This can contribute to economic growth and the promotion of decent work.

SDG 9: Industry, Innovation, and Infrastructure

The digitalisation of industrial processes enables innovation and the development of new technologies, that can improve efficiency, reduce emissions, and make manufacturing more sustainable. In this project, the NOA architecture will help streamline operations and facilitate the development of new "smart" applications such as predictive maintenance or the development of a digital twin.

SDG 12: Responsible Consumption and Production

Digitalisation can help reduce waste and improve resource efficiency in manufacturing, making production more responsible and sustainable. By automating tasks and optimising processes, it's possible to reduce energy consumption and minimise its environmental impact.

SDG 13: Climate Action

Finally, digitalisation projects can play a critical role reducing emissions and mitigating climate change. By enabling a more precise monitoring of emissions, regulatory entities can measure all emissions produced by factories and industrial facilities in real time, forcing companies to comply with environmental regulations. Therefore, digital transformation does not only reduce emissions, but also provides the tools necessary, such as IIoT, to legislate them.

By aligning with these SDGs, the implementation of the NOA architecture in ICAI not only improves efficiency, energy consumption, and safety, but it also contributes to broader sustainability goals. Thus, this project will help to ease maintenance work, improve the academic lectures given at University, and create a platform for the development of new applications, promoting further innovation.

Figure 84. Sustainable Development Goals.

ANNEX II. INFORMATION MODELLING IN OPC UA

OPC UA INFORMATION MODEL FOR INDUSTRIAL COMMUNICATION.

In industrial automation, it is crucial for different devices and systems to communicate with each other seamlessly, requiring a uniform way of representing information. OPC UA addresses this requirement with a standardised information model, an approach that allows data exchange between diverse systems and devices, leading to better interoperability, flexibility, and scalability in industrial environments.

One of the main advantages of using an information model, as is the case with OPC UA, is that it provides a high level of abstraction. OPC UA's information modelling and its service-oriented architecture (mentioned in chapter 4. Technologies) allows more efficient and flexible data exchange compared to other protocols such as Modbus. With Modbus, data exchange is done through sequences of bits/bytes that are transmitted between devices. Consequently, for a Modbus Master to receive specific information from a device, it must send a fixed bit sequence in the correct format to request the desired information from the slave, limiting its usefulness in complex systems. Therefore, OPC UA, through its information model, manages a service-oriented architecture where servers expose services that clients can use to access information quickly and efficiently. These services include reading, writing, or configuring the server, unlocking a more efficient data exchange and reducing the likelihood of coding errors.

In addition, OPC UA's approach provides greater scalability and ease of management. With Modbus, adding new devices or modifying existing ones can be a time-consuming and error-prone process. In contrast, OPC UA's information model allows for easy and seamless integration of new devices and systems, reducing maintenance costs and downtimes.

Moreover, OPC UA provides a secure and dependable communication channel, employing sophisticated encryption and authentication mechanisms to guarantee the secure transmission of data, thereby safeguarding its confidentiality, integrity, availability, and authenticity. This

aspect holds particular significance in industrial applications, where the security and integrity of data is of utmost importance.

BASIC CONCEPTS.

Address Space:

To ensure the accurate representation of intricate and complex data within the information model, the concept of AddressSpace is fundamental. The AddressSpace is a core concept in OPC UA and it serves as a hierarchical structure that organises all the objects, properties, and methods in the system. It comprises all the information that the server makes available to clients, including the objects that represent physical and/or virtual entities, object types, their relationships, and the available services. This AddressSpace is mainly made up of nodes, references, and attributes, providing a unified view of the data that allows clients to easily browse and access specific information.

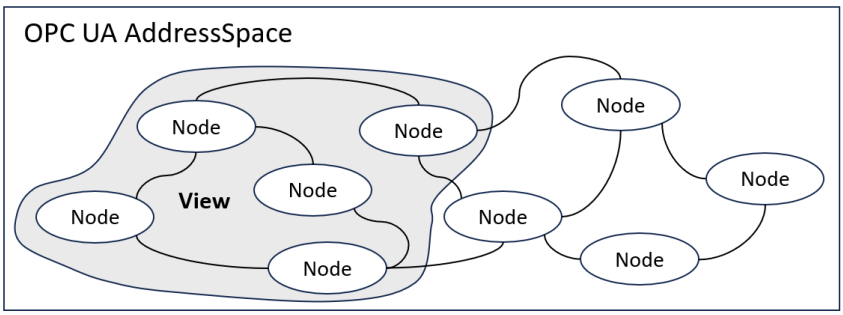


Figure 85. OPC UA Address Space.

Nodes and References:

In OPC UA modelling, the most important elements are *Nodes* and *References*. Nodes serve as the main building blocks within the AddressSpace and effectively characterise entities present within the modelled system. These entities can encompass a wide range of elements, including physical objects like machines or sensors, as well as abstract concepts such as data types or software components. Each node possesses a unique identification known as a NodeId, which serves as a globally distinctive identifier that can be used to refer to the node from anywhere within the system.

On the other hand, References establish the relationships and connections between Nodes, defining the associations and dependencies within the AddressSpace. These References categorise the various connections, such as hierarchical references that denote parent-child relationships, non-hierarchical references that represent associations, and cross-references that establish connections between Nodes that are not hierarchically related. Therefore, References provide the means to navigate and traverse the interconnected Nodes within the AddressSpace and enable clients to discover and access relevant information, establish dependencies, and comprehend the relationships between different entities within the system.

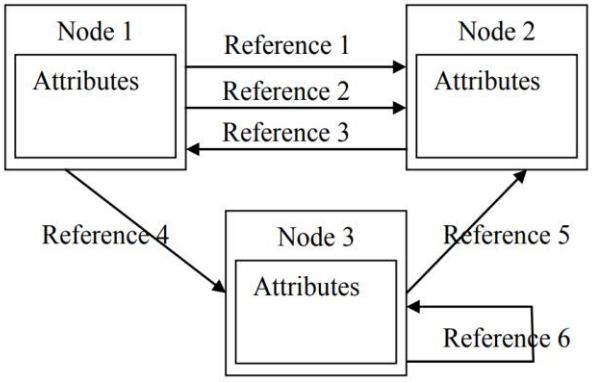


Figure 86. Nodes and References.

OPC UA SPECIFICATIONS:

To ensure the widespread use of information models across diverse applications, it is important to establish consistent rules around how they must be built and presented, ensuring that all products using OPC UA can interpret them. To achieve this, the OPC Foundation created the "Base OPC Information models" mentioned in chapter 4. Within this layer, the foundation defines the fundamental modelling components that serve as the building blocks for constructing more intricate models, ensuring that consistency is maintained across all OPC UA applications.

The OPC UA specifications play a crucial role in establishing a standardised framework for representing and exchanging information in a consistent manner. These specifications offer a collection of standard templates, such as the "Base Node Class," which encompasses essential attributes that the OPC Foundation deems necessary for all nodes. These attributes include the Node Id, a display name, the node class to which it belongs, and a browse name, while they additionally incorporate optional attributes such as the node's description.

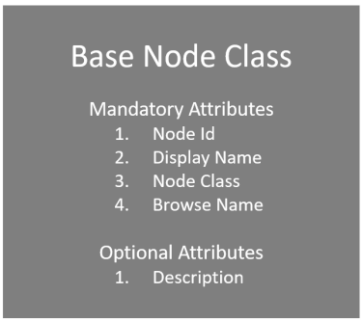


Figure 87. Base Node Class (Industry40tv, 2020c).

Building upon the base template, the OPC Foundation specifies eight special types of base node classes that users can inherit from when creating complex information models for specific use case applications.

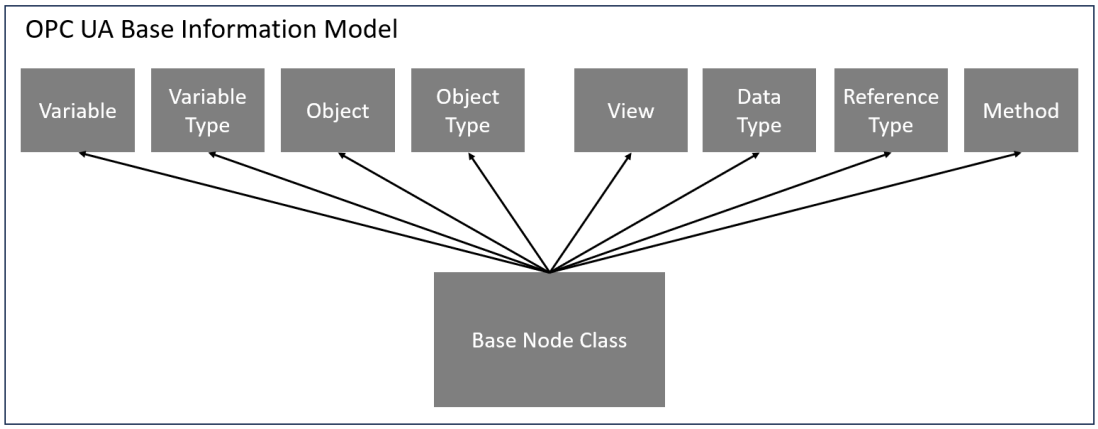


Figure 88. OPC UA Base Information Nodes (Industry40tv, 2020c).

These derived nodes include Object nodes, for representing physical assets that have a state (variable) and behaviour (method), Variable nodes, to represent a value that can be read and/or written, or Method nodes, which can be called by the clients to run specific processes.

The most important nodes from the OPC UA Base Information Model are defined bellow.

DataType Node: A DataType Node defines the characteristics and structure of a data type used in the system. It contains information about the type, such as its name, encoding, and representation. DataType Nodes are typically used to define custom data types or to extend existing ones.

VariableType Node: A VariableType Node defines a template for a Variable Node, which is used to store data in the system. It specifies the data type of the Variable Node, as well as other attributes such as the access level, the minimum and maximum values, and the engineering units.

Variable Node: A Variable Node represents an instance of a VariableType Node. It is used to store a specific value of a data type, which can be read or written by clients.

ObjectType Node: An ObjectType Node defines the characteristics and behaviour of a type of object in the system. It contains information about the object, such as its name, description, and

the methods and properties that are available. ObjectType Nodes are typically used to define custom objects or to extend existing ones.

Object Node: An Object Node represents an instance of an ObjectType Node. It is used to represent a specific object in the system, which can have its own properties, methods, and child nodes. Object Nodes can represent physical devices, logical components, or any other object in the system.

ReferenceType Node: A ReferenceType Node defines a type of reference that can be used to establish relationships between nodes in the Address Space. It contains information about the reference and are typically used to define custom relationships between nodes or to extend existing ones.

Method Node: A Method Node defines a callable operation that can be performed on an Object Node or a Variable Node. It contains information about the method, such as its name, input arguments, output arguments, and the behaviour of the method. When a client invokes a Method Node, the server performs the specified operation and returns the result to the client.

Below is a summary of the graphical standard used to represent the various elements within the AddressSpace.

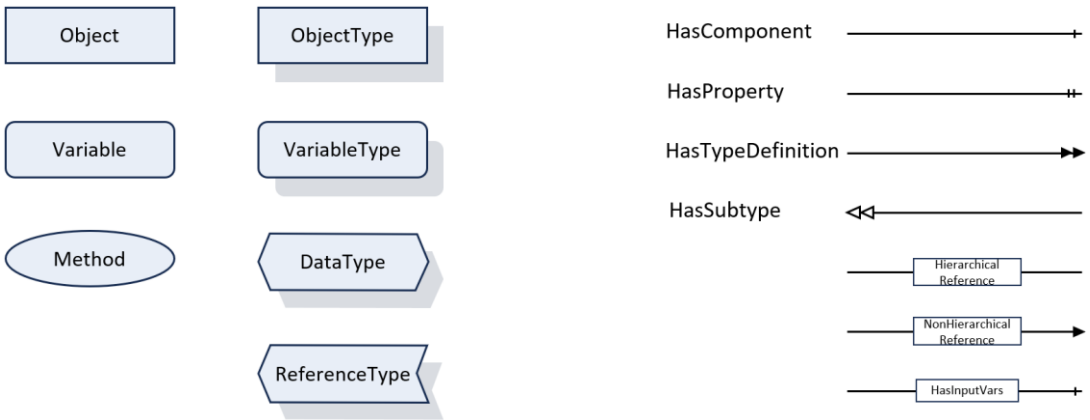


Figure 89. Graphical elements used for information modelling (Original work).

BUILT-IN MODELS AND COMPANION SPECIFICATIONS

In conjunction with OPC UA specifications, information models often incorporate other standardised components to enhance their functionality and interoperability. One such component is the "OPC UA Built-in Information models" that are integrated into servers. These models allow users to incorporate Classic OPC functionalities like Data Acquisition (DA), Historical Data Access (HDA), and Alarms & Events (AE) into the server's information model. Additionally, a standardised OPC UA Program built-in model (Prog) is available, offering a template for executing sets of instructions.

Additionally, to further drive standardisation and promote true interoperability, the OPC Foundation has established "Companion Specifications." These specifications are developed in collaboration with various industries to standardise specific elements present in their respective industrial environments. As a result, software and device manufacturers in industries like pharmaceuticals, robotics, or power generation can employ a shared companion specification model to construct out of the box interoperable systems.

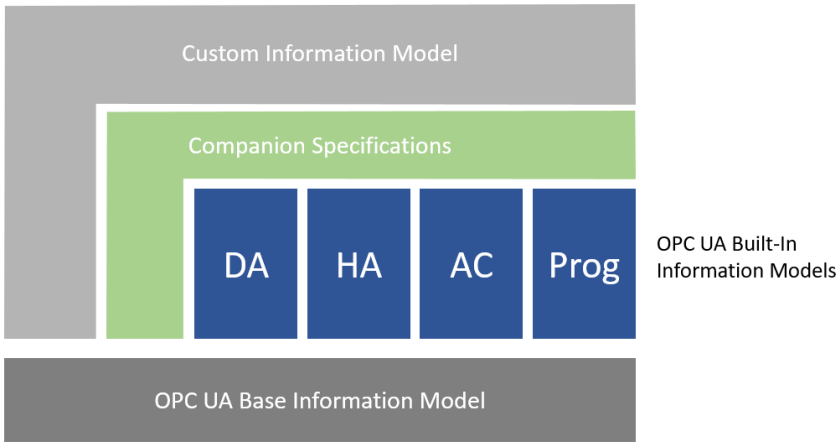


Figure 90. Built-in models and companion specifications (Mahnke, Leitner, & Damm, 2009).

ANNEX III. INFORMATION MODEL XML FILE

This annex provides the XML code for all the nodes described in section 5.1.5.

Base Device Type:

```
<!--
Base Class for every device
-->
<opc:ObjectType SymbolicName="BaseDeviceType" BaseType="ua:BaseObjectType">
<opc:Description>
The Base Type for all devices in the factory is a foundational template that
specifies the essential attributes that every device must possess.
</opc:Description>
<opc:Children>
<opc:Property SymbolicName="Manufacturer" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite"/>
<opc:Property SymbolicName="Type" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite"/>
<opc:Property SymbolicName="MaintenanceSchedule" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Maintenance Schedule</opc:BrowseName>
</opc:Property>
</opc:Children>
</opc:ObjectType>
```

Basic sensor and actuator types:

```
<!--
Object Types: Analog and Digital Sensors.
Analog sensors must include EngineeringUnit and EURange properties (the latter is added by default
when TypeDefinition="ua:AnalogItemType").
Digital sensors only return a Boolean value.
-->
<opc:ObjectType SymbolicName="AnalogSensorType" BaseType="BaseDeviceType">
<opc:Description>Generic analog sensor.</opc:Description>
<opc:Children>
<opc:Variable SymbolicName="Measurement" DataType="ua:Double" ValueRank="Scalar"
TypeDefinition="ua:AnalogItemType" AccessLevel="ReadWrite">
<opc:Children>
<opc:Property SymbolicName="EngineeringUnits" DataType="ua:EUInformation"
ModellingRule="Mandatory" AccessLevel="Read"/>
</opc:Children>
</opc:Variable>
</opc:Children>
</opc:ObjectType>

<opc:ObjectType SymbolicName="DigitalSensorType" BaseType="BaseDeviceType">
<opc:Description>Generic digital sensor.</opc:Description>
<opc:Children>
<opc:Variable SymbolicName="ObjectDetected" DataType="ua:Boolean" ValueRank="Scalar"
AccessLevel="ReadWrite">
<opc:BrowseName>Object Detected</opc:BrowseName>
```

```
</opc:Variable>
</opc:Children>
</opc:ObjectType>

<!--
Object Types: Actuator
State attribute represents if the actuator is active or not (ON/OFF)
-->
<opc:ObjectType SymbolicName="ActuatorType" BaseType="BaseDeviceType">
<opc:Description>Generic industrial actuator.</opc:Description>
<opc:Children>
<opc:Variable SymbolicName="Status" DataType="ua:Boolean" ValueRank="Scalar"
AccessLevel="ReadWrite"/>
</opc:Children>
</opc:ObjectType>
```

Specific sensors:

```
<opc:ObjectType SymbolicName="TemperatureSensorType" BaseType="AnalogSensorType">
<opc:Description>Temperature sensor type node.</opc:Description>
<opc:Children>
<opc:Property SymbolicName="Accuracy" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Accuracy [ºC]</opc:BrowseName>
</opc:Property>
</opc:Children>
</opc:ObjectType>

</opc:ObjectType>
<opc:ObjectType SymbolicName="HumiditySensorType" BaseType="AnalogSensorType">
<opc:Description>Humidity sensor type node.</opc:Description>
<opc:Children>
<opc:Property SymbolicName="HumidityAccuracy" DataType="ua:String"
ModellingRule="Mandatory" AccessLevel="ReadWrite">
<opc:BrowseName>Accuracy [%rH]</opc:BrowseName>
</opc:Property>
</opc:Children>
</opc:ObjectType>
```

As the current factory configuration does not include any limit switch sensors, it has not been modelled in the XML file. However, they can easily be added at a later stage if necessary.

```
<opc:ObjectType SymbolicName="ProximitySensorType" BaseType="DigitalSensorType">
<opc:Description>Proximity sensor type node.</opc:Description>
<opc:Children>
<opc:Property SymbolicName="DetectionRange" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Detection Range [mm]</opc:BrowseName>
</opc:Property>
</opc:Children>
</opc:ObjectType>
```

Specific actuators:

```
<opc:ObjectType SymbolicName="PneumaticActuatorType" BaseType="ActuatorType">
<opc:Description>Pneumatic actuator type node.</opc:Description>
<opc:Children>
<opc:Property SymbolicName="OperatingPressure" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Operating Pressure [Pa]</opc:BrowseName>
</opc:Property>
<opc:Property SymbolicName="ForceOutput" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Force Output [N]</opc:BrowseName>
</opc:Property>
</opc:Children>
</opc:ObjectType>
<opc:ObjectType SymbolicName="ConveyorBeltType" BaseType="ActuatorType">
<opc:Description>Conveyor belt type node.</opc:Description>
<opc:Children>
<opc:Variable SymbolicName="Speed" DataType="ua:Double" ValueRank="Scalar"
TypeDefinition="ua:AnalogItemType" AccessLevel="ReadWrite">
<opc:Children>
<opc:Property SymbolicName="EngineeringUnits" DataType="ua:EUInformation"
ModellingRule="Mandatory" AccessLevel="Read"/>
</opc:Children>
</opc:Variable>
<opc:Property SymbolicName="Length" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Length [m]</opc:BrowseName>
</opc:Property>
<opc:Property SymbolicName="Width" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Width [m]</opc:BrowseName>
</opc:Property>
<opc:Property SymbolicName="MaxLoad" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Max Load [N]</opc:BrowseName>
</opc:Property>
<opc:Property SymbolicName="Material" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite"/>
</opc:Children>
</opc:ObjectType>

<opc:ObjectType SymbolicName="RobotType" BaseType="ActuatorType">
<opc:Description>Robot type node.</opc:Description>
<opc:Children>
<opc:Variable SymbolicName="Gripper" DataType="ua:Boolean" ValueRank="Scalar"
AccessLevel="ReadWrite"/>
<opc:Property SymbolicName="MaxPayload" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>MaxPayload [N]</opc:BrowseName>
</opc:Property>
<opc:Property SymbolicName="MaxSpeed" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>MaxSpeed [m/s]</opc:BrowseName>
</opc:Property>
<opc:Property SymbolicName="MaxAcceleration" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Max Acceleration [m/s2]</opc:BrowseName>
</opc:Property>
<opc:Property SymbolicName="Reach" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Reach [m]</opc:BrowseName>
</opc:Property>
<opc:Property SymbolicName="NumAxis" DataType="ua:Integer" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Number of Axis</opc:BrowseName>
```

```
</opc:Property>
</opc:Children>
</opc:ObjectType>
```

Integrated Modules:

```
<!--
Object Types: Factory Modules.
Stoppers: They halt/stop the pallets.  =  Proximity Sensor + Pneumatic Actuator
Lifts: Used to grab the pallets and elevate them  =  Proximity Sensor x 2 + Pneumatic Actuator
- Positioners: Used to grab and immobilise the pallets  =  Pneumatic Actuator
Divert: Redirects the flow of objects on the conveyor belt.  =  Stopper x 2 + Pneumatic Actuator
Combine: Merges two or more product lines into a single stream.  =  Stopper x 2 + Pneumatic Act.
-->

<opc:ObjectType SymbolicName="StopperType" BaseType="ua:BaseObjectType">
<opc:Description>Stopper type node.</opc:Description>
<opc:Children>
<opc:Object SymbolicName="ProximitySensor" TypeDefinition="ProximitySensorType"
SupportsEvents="true">
<opc:BrowseName>Proximity Sensor</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="PneumaticActuator" TypeDefinition="PneumaticActuatorType"
SupportsEvents="true">
<opc:BrowseName>Pneumatic Actuator</opc:BrowseName>
</opc:Object>
</opc:Children>
</opc:ObjectType>

<opc:ObjectType SymbolicName="LiftType" BaseType="ua:BaseObjectType">
<opc:Description>Lift type node.</opc:Description>
<opc:Children>
<opc:Object SymbolicName="ProximitySensorUp" TypeDefinition="ProximitySensorType"
SupportsEvents="true">
<opc:BrowseName>Proximity Sensor Up</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="ProximitySensorDown" TypeDefinition="ProximitySensorType"
SupportsEvents="true">
<opc:BrowseName>Proximity Sensor Down</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="PneumaticActuator" TypeDefinition="PneumaticActuatorType"
SupportsEvents="true">
<opc:BrowseName>Pneumatic Actuator</opc:BrowseName>
</opc:Object>
</opc:Children>
</opc:ObjectType>
<opc:ObjectType SymbolicName="PositionerType" BaseType="ua:BaseObjectType">
<opc:Description>Positioner type node.</opc:Description>
<opc:Children>
<opc:Object SymbolicName="PneumaticActuator" TypeDefinition="PneumaticActuatorType"
SupportsEvents="true">
<opc:BrowseName>Pneumatic Actuator</opc:BrowseName>
</opc:Object>
</opc:Children>
</opc:ObjectType>

<opc:ObjectType SymbolicName="DivertType" BaseType="ua:BaseObjectType">
<opc:Description>Divert type node.</opc:Description>
<opc:Children>
<opc:Object SymbolicName="Stopper" TypeDefinition="StopperType" SupportsEvents="true">
<opc:BrowseName>Stopper</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="ReturnStopper" TypeDefinition="StopperType" SupportsEvents="true">
<opc:BrowseName>Return Stopper</opc:BrowseName>
</opc:Object>
```

```
<opc:Object SymbolicName="Arm" TypeDefinition="PneumaticActuatorType" SupportsEvents="true">
<opc:BrowseName>Arm</opc:BrowseName>
</opc:Object>
</opc:Children>
</opc:ObjectType>

<opc:ObjectType SymbolicName="CombineType" BaseType="ua:BaseObjectType">
<opc:Description>Combine type node.</opc:Description>
<opc:Children>
<opc:Object SymbolicName="Stopper" TypeDefinition="StopperType" SupportsEvents="true">
<opc:BrowseName>Stopper</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="PreStopper" TypeDefinition="StopperType" SupportsEvents="true">
<opc:BrowseName>Pre-Stopper</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="Arm" TypeDefinition="PneumaticActuatorType" SupportsEvents="true">
<opc:BrowseName>Arm</opc:BrowseName>
</opc:Object>
</opc:Children>
</opc:ObjectType>
```

Stations

```
<!--
Stations: the factory has two types of stations.
Tunnel Station (Equipped with lifts)  =  Stopper x 2 + Lift + Robot
- Basic Stations (Equipped with positioners)   =  Stopper x 2 + Positioner + Robot
-->
<opc:ObjectType SymbolicName="TunnelStationType" BaseType="ua:BaseObjectType">
<opc:Description>Tunnel station type node.</opc:Description>
<opc:Children>
<opc:Object SymbolicName="PreStopper" TypeDefinition="StopperType" SupportsEvents="true">
<opc:BrowseName>Pre-Stopper</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="Stopper" TypeDefinition="StopperType" SupportsEvents="true">
<opc:BrowseName>Stopper</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="Lift" TypeDefinition="LiftType" SupportsEvents="true">
<opc:BrowseName>Lift</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="Robot" TypeDefinition="RobotType" SupportsEvents="true">
<opc:BrowseName>Robot</opc:BrowseName>
</opc:Object>
</opc:Children>
</opc:ObjectType>

<opc:ObjectType SymbolicName="PositionerStationType" BaseType="ua:BaseObjectType">
<opc:Description>Positioner station type node.</opc:Description>
<opc:Children>
<opc:Object SymbolicName="PreStopper" TypeDefinition="StopperType" SupportsEvents="true">
<opc:BrowseName>Pre-Stopper</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="Stopper" TypeDefinition="StopperType" SupportsEvents="true">
<opc:BrowseName>Stopper</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="Positioner" TypeDefinition="PositionerType" SupportsEvents="true">
<opc:BrowseName>Positioner</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="Robot" TypeDefinition="RobotType" SupportsEvents="true">
<opc:BrowseName>Robot</opc:BrowseName>
</opc:Object>
</opc:Children>
</opc:ObjectType>
```

Factory

```
<!--
Stations: In total there are four stations.
Station 1: Tunnel Type
Station 2: Basic Type
Station 3: Basic Type
Station 4: Tunnel Type
-->
<opc:ObjectType SymbolicName="StationsConfiguration" BaseType="ua:FolderType">
<opc:Children>
<opc:Object SymbolicName="Station1" TypeDefinition="TunnelStationType" SupportsEvents="true">
<opc:BrowseName>Station S1</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="Station2" TypeDefinition="PositionerStationType" SupportsEvents="true">
<opc:BrowseName>Station S2</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="Station3" TypeDefinition="PositionerStationType" SupportsEvents="true">
<opc:BrowseName>Station S3</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="Station4" TypeDefinition="TunnelStationType" SupportsEvents="true">
<opc:BrowseName>Station S4</opc:BrowseName>
</opc:Object>
</opc:Children>
</opc:ObjectType>


<!--
Conveyor Control System: The factory works with two divert and two combine needles that allow
different configurations.
Two Divert needles: Divert12 and Divert34
Two Combine needles: Combine3 and Combine4
-->
<opc:ObjectType SymbolicName="ConveyorControlSystemConfiguration" BaseType="ua:FolderType">
<opc:Children>
<opc:Object SymbolicName="Divert12" TypeDefinition="DivertType" SupportsEvents="true">
<opc:BrowseName>Divert 12</opc:BrowseName>
<opc:Description>
It divides the HIGHWAY conveyor into 2 virtual conveyors: this one is to move pallets
between stations S1 and S2.
</opc:Description>
</opc:Object>
<opc:Object SymbolicName="Divert34" TypeDefinition="DivertType" SupportsEvents="true">
<opc:BrowseName>Divert 34</opc:BrowseName>
<opc:Description>
It divides the HIGHWAY conveyor into 2 virtual conveyors: this one is to move pallets
between stations S3 and S4.
</opc:Description>
</opc:Object>
<opc:Object SymbolicName="Combine2" TypeDefinition="CombineType" SupportsEvents="true">
<opc:BrowseName>Combine 2</opc:BrowseName>
<opc:Description>
It divert packages from the HIGHWAY to its ribbon (Station 2), or to feed packages from the
latter
to the HIGHWAY.
</opc:Description>
</opc:Object>
<opc:Object SymbolicName="Combine3" TypeDefinition="CombineType" SupportsEvents="true">
<opc:BrowseName>Combine 3</opc:BrowseName>
<opc:Description>
It divert packages from the HIGHWAY to its ribbon (Station 3), or to feed packages from the
latter
to the HIGHWAY.
</opc:Description>
</opc:Object>
<opc:Object SymbolicName="GStop" TypeDefinition="StopperType" SupportsEvents="true">
```

```
<opc:BrowseName>GStop</opc:BrowseName>
<opc:Description>A stopper for the entire assembly.</opc:Description>
</opc:Object>
</opc:Children>
</opc:ObjectType>

<!--
Conveyor belt system: the factory has three conveyor belts:
Highway
Satelite conveyor belt for station 2
Satelite conveyor belt for station 3
-->
<opc:ObjectType SymbolicName="ConveyorBeltsConfiguration" BaseType="ua:FolderType">
<opc:Children>
<opc:Object SymbolicName="Highway" TypeDefinition="ConveyorBeltType" SupportsEvents="true">
<opc:BrowseName>Highway</opc:BrowseName>
<opc:Description>Main conveyor of the factory.</opc:Description>
</opc:Object>
<opc:Object SymbolicName="Satelite2" TypeDefinition="ConveyorBeltType" SupportsEvents="true">
<opc:BrowseName>Satelite Conveyor 2</opc:BrowseName>
<opc:Description>Secondary belt for station 2.</opc:Description>
</opc:Object>
<opc:Object SymbolicName="Satelite3" TypeDefinition="ConveyorBeltType" SupportsEvents="true">
<opc:BrowseName>Satelite Conveyor 3</opc:BrowseName>
<opc:Description>Secondary belt for station 3.</opc:Description>
</opc:Object>
</opc:Children>
</opc:ObjectType>
<!--
Whole conveyor system: Conveyor Belts + Conveyor Control Systems
-->
<opc:ObjectType SymbolicName="ConveyorsConfiguration" BaseType="ua:FolderType">
<opc:Children>
<opc:Object SymbolicName="ConveyorBelts" TypeDefinition="ConveyorBeltsConfiguration"
SupportsEvents="true">
<opc:BrowseName>Conveyor Belts</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="ConveyorSystemControls"
TypeDefinition="ConveyorControlSystemConfiguration" SupportsEvents="true">
<opc:BrowseName>Conveyor System Controls</opc:BrowseName>
</opc:Object>
</opc:Children>
</opc:ObjectType>

<!--
Factory: it is composed by all four stations and the conveyor belt system
-->
<opc:ObjectType SymbolicName="FactoryConfiguration" BaseType="ua:FolderType">
<opc:Children>
<opc:Object SymbolicName="Stations" TypeDefinition="StationsConfiguration" SupportsEvents="true">
<opc:BrowseName>Stations</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="Conveyors" TypeDefinition="ConveyorsConfiguration"
SupportsEvents="true">
<opc:BrowseName>Conveyors</opc:BrowseName>
</opc:Object>
</opc:Children>
</opc:ObjectType>
```

M+O System

```
<!--
M+O System: it is composed by non intrusive sensors:
  - XY-MD02 Sensor to measure humidity and temperature    =   Humidity Sensor + Temperature Sensor
-->
<opc:ObjectType SymbolicName="XYMD02SensorType" BaseType="ua:BaseObjectType">
<opc:Description>XY-MD02 sensor that measures humidity and temperature.</opc:Description>
<opc:Children>
<opc:Object SymbolicName="TemperatureSensor" TypeDefinition="TemperatureSensorType"
SupportsEvents="true">
<opc:BrowseName>Temperature Sensor</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="HumiditySensor" TypeDefinition="HumiditySensorType" SupportsEvents="true">
<opc:BrowseName>Humidity Sensor</opc:BrowseName>
</opc:Object>
<opc:Property SymbolicName="CommunicationProtocol" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Communication Protocol</opc:BrowseName>
</opc:Property>
<opc:Property SymbolicName="InputVoltage" DataType="ua:String" ModellingRule="Mandatory"
AccessLevel="ReadWrite">
<opc:BrowseName>Input Voltage</opc:BrowseName>
</opc:Property>
</opc:Children>
</opc:ObjectType>

<opc:ObjectType SymbolicName="MOSystemConfiguration" BaseType="ua:FolderType">
<opc:Children>
<opc:Object SymbolicName="XYMD02Sensor1" TypeDefinition="XYMD02SensorType" SupportsEvents="true">
<opc:BrowseName>XY-MD02 Sensor #1</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="XYMD02Sensor2" TypeDefinition="XYMD02SensorType" SupportsEvents="true">
<opc:BrowseName>XY-MD02 Sensor #2</opc:BrowseName>
</opc:Object>
</opc:Children>
</opc:ObjectType>
```

OPC UA Server

```
<!--
OPC UA Server: ICAI Factory
-->
<opc:ObjectType SymbolicName="OPCUAServerType" BaseType="ua:BaseObjectType" SupportsEvents="true">
<opc:Description>Stations of the factory.</opc:Description>
<opc:Children>
<opc:Object SymbolicName="Factory" TypeDefinition="FactoryConfiguration" SupportsEvents="true">
<opc:BrowseName>Factory</opc:BrowseName>
</opc:Object>
<opc:Object SymbolicName="MOSystem" TypeDefinition="MOSystemConfiguration" SupportsEvents="true">
<opc:BrowseName>M+O System</opc:BrowseName>
</opc:Object>
</opc:Children>
</opc:ObjectType>

<opc:Object SymbolicName="OPCUAServer1" TypeDefinition="OPCUAServerType" SupportsEvents="true">
<opc:BrowseName>ICAI Factory</opc:BrowseName>
<opc:References>
<opc:Reference IsInverse="true">
<opc:ReferenceType>ua:Organizes</opc:ReferenceType>
<opc:TargetId>ua:ObjectsFolder</opc:TargetId>
</opc:Reference>
</opc:References>
</opc:Object>
```

ANNEX IV. OPC UA SERVER FILES

This annex comprises all scripts from the OPC UA server, except for the windows form scripts.

Program.cs

```
using Microsoft.AspNetCore.Hosting.Server;
using Opc.Ua;
using Opc.Ua.Configuration;
using System;
using System.IO;
using System.Windows.Forms;

namespace ICAIFactoryServer
{
    internal static class Program
    {
        // Configuration folder
        private static string configurationDirectory = @"Files\Configuration";

        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        private static void Main()
        {
            // Create default configuration.txt and update Configuration XML File
            LoadConfiguration.CreateDefaultConfig();
            LoadConfiguration.UpdateConfigurationFile();

            // Initialize application configuration
            ApplicationConfiguration.Initialize();

            ApplicationInstance app = new ApplicationInstance();
            app.ApplicationType = ApplicationType.Server;
            app.ConfigSectionName = "ICAI OPC UA Server";

            try
            {
                if (app.ProcessCommandLine())
                {
                    return;
                }

                if (!Environment.UserInteractive)
                {
                    app.StartAsService(new Server());
                    return;
                }

                // Load application configuration and check instance certificate
                string configFile = Path.Combine(configurationDirectory,
                        "OPCUAServer.Config.xml");
                app.LoadApplicationConfiguration(configFile, false).Wait();
                app.CheckApplicationInstanceCertificate(false, 0).Wait();

                // Start the OPC UA server and run the main form
                app.Start(new Server()).Wait();
                Application.Run(new Form_Main(app));
```

```
            }
            catch (Exception e)
            {
                Console.WriteLine(e.ToString());
            }
        }
    }
}
```

Server.cs

```
namespace ICAIFactoryServer
{
    internal class Server : StandardServer
    {
        private UserDatabase database;

        protected override MasterNodeManager
          CreateMasterNodeManager(IServerInternal server,
          Opc.Ua.ApplicationConfiguration configuration)
        {
            List<INodeManager> nodeManagers = new List<INodeManager>();
            nodeManagers.Add(new NodeManager(server, configuration));
            return new MasterNodeManager(server, configuration, null,
                nodeManagers.ToArray());
        }

        protected override ServerProperties LoadServerProperties()
        {
            var properties = new ServerProperties
            {
                ManufacturerName = "Gonzalo Carrasco",
                ProductName = "ICAI Factory OPC UA Server",
                ProductUri =
                        "http://opcfoundation.org/OPCServer/ICAIFactoryServer/v1.0",
                SoftwareVersion = Utils.GetAssemblySoftwareVersion(),
                BuildNumber = Utils.GetAssemblyBuildNumber(),
                BuildDate = Utils.GetAssemblyTimestamp()
            };

            return properties;
        }

        protected override void OnServerStarted(IServerInternal server)
        {
            database = UserDatabase.GetInstance();
            // request notifications when the user identity is changed. all valid
                users are accepted by default.
            server.SessionManager.ImpersonateUser += new
              ImpersonateEventHandler(SessionManager_ImpersonateUser);
        }

        /// <summary>
        /// Called when a client tries to change its user identity. This means
                its called when a
        /// user wants to connect to the server (not as anonymous) or wants to
                change of user.
        /// </summary>
        private void SessionManager_ImpersonateUser(Session session,
                ImpersonateEventArgs args)
        {
            UserNameIdentityToken userNameToken = args.NewIdentity as
            UserNameIdentityToken;
            if (userNameToken != null)
            {
                if (database.AuthenticateUser(userNameToken.UserName,
```

```
                    userNameToken.DecryptedPassword))
                {
                    database.GoodAccess(userNameToken.UserName);
                }
                else
                {
                    args.IdentityValidationError =
                        StatusCodes.BadUserAccessDenied;
                    database.BadAccess(userNameToken.UserName);
                }
            }
        }
    }
}
```

NodeManager.cs

```
namespace ICAIFactoryServer
{
    internal class NodeManager : CustomNodeManager2
    {
        private static OPCUAServerState opcUaServer;
        private static NodeIds nodeIds;

        public NodeManager(IServerInternal server,
          Opc.Ua.ApplicationConfiguration configuration) : base(server,
          configuration)
        {
            SystemContext.NodeIdFactory = this;

            string[] namespaceUrls = new string[2];
            namespaceUrls[0] = Namespaces.ICAIFactoryServer;
            namespaceUrls[1] = Namespaces.ICAIFactoryServer + "/Instance";
            SetNamespaces(namespaceUrls);
        }

        protected override NodeStateCollection LoadPredefinedNodes(ISystemContext
        context)
        {
            NodeStateCollection predefinedNodes = new NodeStateCollection();
            predefinedNodes.LoadFromBinaryResource(context,
            @"Files\Data\ICAIFactoryServer.PredefinedNodes.uanodes",
            typeof(NodeManager).GetTypeInfo().Assembly,
            true);

            return predefinedNodes;
        }

        public override void CreateAddressSpace(IDictionary<NodeId,
          IList<IReference>> externalReferences)
        {
            lock (Lock)
            {
                LoadPredefinedNodes(SystemContext, externalReferences);

                // Find the untyped Fabrica ICAI node that was created when the
                      model was loaded.
                BaseObjectState passiveNode =
                   (BaseObjectState)FindPredefinedNode(
                    new NodeId(ICAIFactoryServer.Objects.OPCUAServer1,
                         NamespaceIndexes[0]), typeof(BaseObjectState));

                // Convert the untyped node to a typed node that can be
                        manipulated within the server.
                opcUaServer = new OPCUAServerState(null);
```

```
            opcUaServer.Create(SystemContext, passiveNode);

            // THIS IS THE NEW CODE
            nodeIds = new NodeIds();
            nodeIds.UpdateNodesIds(opcUaServer);

            // Replace the untyped predefined nodes with their strongly typed
                versions.
            AddPredefinedNode(SystemContext, opcUaServer);

            // Set engineering units and range values for temperature and
                speed measurements
            SetMOSystemSensorProperties();
            SetConveyorSpeedProperties();
            SetRobotAxisProperties();
        }

        // Method handlers

    }

    private void SetMOSystemSensorProperties()
    {
        Opc.Ua.Range temperatureRange = new Opc.Ua.Range
        {
            Low = -5.0,
            High = 70.0
        };
        Opc.Ua.Range humidityRange = new Opc.Ua.Range
        {
            Low = 0.0,
            High = 100.0
        };

        opcUaServer.MOSystem.XYMD02Sensor1.TemperatureSensor.Measurement.EngineeringUnits.Value =
opcUaServer.MOSystem.XYMD02Sensor2.TemperatureSensor.Measurement.EngineeringUnits.Value =
new EUInformation("ºC", "Degrees Celsius", "http://www.opcfoundation.org/UA/units/un/cefact");
        opcUaServer.MOSystem.XYMD02Sensor1.TemperatureSensor.Measurement.EURange.Value =
opcUaServer.MOSystem.XYMD02Sensor2.TemperatureSensor.Measurement.EURange.Value =
            temperatureRange;
        opcUaServer.MOSystem.XYMD02Sensor1.HumiditySensor.Measurement.EngineeringUnits.Value =
opcUaServer.MOSystem.XYMD02Sensor2.HumiditySensor.Measurement.EngineeringUnits.Value =
new EUInformation("%rH", "Relative Humidity", "http://www.opcfoundation.org/UA/units/un/cefact");
        opcUaServer.MOSystem.XYMD02Sensor1.HumiditySensor.Measurement.EURange.Value =
opcUaServer.MOSystem.XYMD02Sensor2.HumiditySensor.Measurement.EURange.Value =
            humidityRange;

opcUaServer.MOSystem.XYMD02Sensor1.InputVoltage.Value =
opcUaServer.MOSystem.XYMD02Sensor2.InputVoltage.Value = "DC: 5-30V";

opcUaServer.MOSystem.XYMD02Sensor1.CommunicationProtocol.Value =
opcUaServer.MOSystem.XYMD02Sensor2.CommunicationProtocol.Value = "Modbus RS-485";
    }

    private void SetConveyorSpeedProperties()
    {
        Opc.Ua.Range speedRange = new Opc.Ua.Range
        {
            Low = 0.0,
            High = 10.0
        };
        string speedUnits = "m/s";
        string speedDescription = "Meters per second";

        opcUaServer.Factory.Conveyors.ConveyorBelts.Highway.Speed.EngineeringUnits.Value =
new EUInformation(speedUnits, speedDescription, "http://www.opcfoundation.org/UA/units/un/cefact");
        opcUaServer.Factory.Conveyors.ConveyorBelts.Highway.Speed.EURange.Value = speedRange;
```

```
            opcUaServer.Factory.Conveyors.ConveyorBelts.Satelite2.Speed.EngineeringUnits.Value =
new EUInformation(speedUnits, speedDescription, "http://www.opcfoundation.org/UA/units/un/cefact");
            opcUaServer.Factory.Conveyors.ConveyorBelts.Satelite2.Speed.EURange.Value = speedRange;
            opcUaServer.Factory.Conveyors.ConveyorBelts.Satelite3.Speed.EngineeringUnits.Value =
new EUInformation(speedUnits, speedDescription, "http://www.opcfoundation.org/UA/units/un/cefact");
            opcUaServer.Factory.Conveyors.ConveyorBelts.Satelite3.Speed.EURange.Value = speedRange;
        }
        private void SetRobotAxisProperties()
        {
            int numAxis = 6;
            opcUaServer.Factory.Stations.Station1.Robot.NumAxis.Value = numAxis;
            opcUaServer.Factory.Stations.Station2.Robot.NumAxis.Value = numAxis;
            opcUaServer.Factory.Stations.Station3.Robot.NumAxis.Value = numAxis;
            opcUaServer.Factory.Stations.Station4.Robot.NumAxis.Value = numAxis;
        }
    }
}
```

NodeIds.cs

```
namespace ICAIFactoryServer
{
    internal class NodeIds
    {
        public void UpdateNodesIds(OPCUAServerState serverState)
        {
            /// <summary>
            /// STATIONS
            /// </summary>
            //Station 1
serverState.Factory.Stations.Station1.PreStopper.ProximitySensor.ObjectDetected.NodeId =
new NodeId("\"I_S1_PSTOP\"",
serverState.Factory.Stations.Station1.PreStopper.ProximitySensor.ObjectDetected.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station1.Stopper.ProximitySensor.ObjectDetected.NodeId =
new NodeId("\"I_S1_STOP\"",
serverState.Factory.Stations.Station1.Stopper.ProximitySensor.ObjectDetected.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station1.Lift.ProximitySensorDown.ObjectDetected.NodeId =
new NodeId("\"I_S1_LIFT_DOWN\"",
serverState.Factory.Stations.Station1.Lift.ProximitySensorDown.ObjectDetected.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station1.Lift.ProximitySensorUp.ObjectDetected.NodeId =
new NodeId("\"I_S1_LIFT_UP\"",
serverState.Factory.Stations.Station1.Lift.ProximitySensorUp.ObjectDetected.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station1.PreStopper.PneumaticActuator.Status.NodeId =
new NodeId("\"O_S1_PSTOP_ON\"",
serverState.Factory.Stations.Station1.PreStopper.PneumaticActuator.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station1.Stopper.PneumaticActuator.Status.NodeId =
                new NodeId("\"O_S1_STOP_ON\"",
serverState.Factory.Stations.Station1.Stopper.PneumaticActuator.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station1.Lift.PneumaticActuator.Status.NodeId =
                new NodeId("\"O_S1_LIFT_UP\"",
serverState.Factory.Stations.Station1.Lift.PneumaticActuator.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station1.Robot.Status.NodeId =
                new NodeId("\"O_S1_ROBOT_ON\"",
serverState.Factory.Stations.Station1.Robot.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station1.Robot.Gripper.NodeId =
                new NodeId("\"O_S1_GRIPPER_ON\"",
serverState.Factory.Stations.Station1.Robot.Gripper.NodeId.NamespaceIndex);
            //Station 2
serverState.Factory.Stations.Station2.PreStopper.ProximitySensor.ObjectDetected.NodeId =
new NodeId("\"I_S2_PSTOP\"",
serverState.Factory.Stations.Station2.PreStopper.ProximitySensor.ObjectDetected.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station2.Stopper.ProximitySensor.ObjectDetected.NodeId =
new NodeId("\"I_S2_STOP\"",
serverState.Factory.Stations.Station2.Stopper.ProximitySensor.ObjectDetected.NodeId.NamespaceIndex);
```

```
            serverState.Factory.Stations.Station2.PreStopper.PneumaticActuator.Status.NodeId =
new NodeId("\"O_S2_PSTOP_ON\"",
serverState.Factory.Stations.Station2.PreStopper.PneumaticActuator.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station2.Stopper.PneumaticActuator.Status.NodeId =
                new NodeId("\"O_S2_STOP_ON\"",
serverState.Factory.Stations.Station2.Stopper.PneumaticActuator.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station2.Positioner.PneumaticActuator.Status.NodeId =
new NodeId("\"O_S2_LOCK_UP\"",
serverState.Factory.Stations.Station2.Positioner.PneumaticActuator.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station2.Robot.Status.NodeId =
                new NodeId("\"O_S2_ROBOT_ON\"",
serverState.Factory.Stations.Station2.Robot.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station2.Robot.Gripper.NodeId =
                new NodeId("\"O_S2_GRIPPER_ON\"",
serverState.Factory.Stations.Station2.Robot.Gripper.NodeId.NamespaceIndex);
            //Station 3
serverState.Factory.Stations.Station3.PreStopper.ProximitySensor.ObjectDetected.NodeId =
new NodeId("\"I_S3_PSTOP\"",
serverState.Factory.Stations.Station3.PreStopper.ProximitySensor.ObjectDetected.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station3.Stopper.ProximitySensor.ObjectDetected.NodeId =
new NodeId("\"I_S3_STOP\"",
serverState.Factory.Stations.Station3.Stopper.ProximitySensor.ObjectDetected.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station3.PreStopper.PneumaticActuator.Status.NodeId =
new NodeId("\"O_S3_PSTOP_ON\"",
serverState.Factory.Stations.Station3.PreStopper.PneumaticActuator.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station3.Stopper.PneumaticActuator.Status.NodeId =
                new NodeId("\"O_S3_STOP_ON\"",
serverState.Factory.Stations.Station3.Stopper.PneumaticActuator.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station3.Positioner.PneumaticActuator.Status.NodeId =
new NodeId("\"O_S3_LOCK_UP\"",
serverState.Factory.Stations.Station3.Positioner.PneumaticActuator.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station3.Robot.Status.NodeId =
                new NodeId("\"O_S3_ROBOT_ON\"",
serverState.Factory.Stations.Station3.Robot.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station3.Robot.Gripper.NodeId =
                new NodeId("\"O_S3_GRIPPER_ON\"",
serverState.Factory.Stations.Station3.Robot.Gripper.NodeId.NamespaceIndex);
            //Station 4
serverState.Factory.Stations.Station4.PreStopper.ProximitySensor.ObjectDetected.NodeId =
new NodeId("\"I_S4_PSTOP\"",
serverState.Factory.Stations.Station4.PreStopper.ProximitySensor.ObjectDetected.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station4.Stopper.ProximitySensor.ObjectDetected.NodeId =
new NodeId("\"I_S4_STOP\"",
serverState.Factory.Stations.Station4.Stopper.ProximitySensor.ObjectDetected.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station4.Lift.ProximitySensorDown.ObjectDetected.NodeId =
new NodeId("\"I_S4_LIFT_DOWN\"",
serverState.Factory.Stations.Station4.Lift.ProximitySensorDown.ObjectDetected.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station4.Lift.ProximitySensorUp.ObjectDetected.NodeId =
new NodeId("\"I_S4_LIFT_UP\"",
serverState.Factory.Stations.Station4.Lift.ProximitySensorUp.ObjectDetected.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station4.PreStopper.PneumaticActuator.Status.NodeId =
new NodeId("\"O_S4_PSTOP_ON\"",
serverState.Factory.Stations.Station4.PreStopper.PneumaticActuator.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station4.Stopper.PneumaticActuator.Status.NodeId =
                new NodeId("\"O_S4_STOP_ON\"",
serverState.Factory.Stations.Station4.Stopper.PneumaticActuator.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station4.Lift.PneumaticActuator.Status.NodeId =
                new NodeId("\"O_S4_LIFT_UP\"",
serverState.Factory.Stations.Station4.Lift.PneumaticActuator.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station4.Robot.Status.NodeId =
                new NodeId("\"O_S4_ROBOT_ON\"",
serverState.Factory.Stations.Station4.Robot.Status.NodeId.NamespaceIndex);
            serverState.Factory.Stations.Station4.Robot.Gripper.NodeId =
                new NodeId("\"O_S4_GRIPPER_ON\"",
serverState.Factory.Stations.Station4.Robot.Gripper.NodeId.NamespaceIndex);

            /// <summary>
```

```
            /// CONVEYORS
            /// </summary>
            // Divert 12

serverState.Factory.Conveyors.ConveyorSystemControls.Divert12.Stopper.ProximitySensor.ObjectDetected.No
deId =
                new NodeId("\"I_D12_STOP\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Divert12.Stopper.ProximitySensor.ObjectDetected.No
deId.NamespaceIndex);

serverState.Factory.Conveyors.ConveyorSystemControls.Divert12.ReturnStopper.ProximitySensor.ObjectDetec
ted.NodeId =
                new NodeId("\"I_D12_RSTOP\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Divert12.ReturnStopper.ProximitySensor.ObjectDetec
ted.NodeId.NamespaceIndex);

serverState.Factory.Conveyors.ConveyorSystemControls.Divert12.Stopper.PneumaticActuator.Status.NodeId =
                new NodeId("\"O_D12_STOP_ON\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Divert12.Stopper.PneumaticActuator.Status.NodeId.N
amespaceIndex);

serverState.Factory.Conveyors.ConveyorSystemControls.Divert12.ReturnStopper.PneumaticActuator.Status.No
deId =
                new NodeId("\"O_D12_RSTOP_ON\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Divert12.ReturnStopper.PneumaticActuator.Status.No
deId.NamespaceIndex);
            serverState.Factory.Conveyors.ConveyorSystemControls.Divert12.Arm.Status.NodeId =
                new NodeId("\"O_D12_ARM_ST\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Divert12.Arm.Status.NodeId.NamespaceIndex);
            // Divert 34
serverState.Factory.Conveyors.ConveyorSystemControls.Divert34.Stopper.ProximitySensor.ObjectDetected.No
deId =
                new NodeId("\"I_D34_STOP\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Divert34.Stopper.ProximitySensor.ObjectDetected.No
deId.NamespaceIndex);

serverState.Factory.Conveyors.ConveyorSystemControls.Divert34.ReturnStopper.ProximitySensor.ObjectDetec
ted.NodeId =
                new NodeId("\"I_D34_RSTOP\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Divert34.ReturnStopper.ProximitySensor.ObjectDetec
ted.NodeId.NamespaceIndex);

serverState.Factory.Conveyors.ConveyorSystemControls.Divert34.Stopper.PneumaticActuator.Status.NodeId =
                new NodeId("\"O_D34_STOP_ON\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Divert34.Stopper.PneumaticActuator.Status.NodeId.N
amespaceIndex);

serverState.Factory.Conveyors.ConveyorSystemControls.Divert34.ReturnStopper.PneumaticActuator.Status.No
deId =
                new NodeId("\"O_D34_RSTOP_ON\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Divert34.ReturnStopper.PneumaticActuator.Status.No
deId.NamespaceIndex);
            serverState.Factory.Conveyors.ConveyorSystemControls.Divert34.Arm.Status.NodeId =
                new NodeId("\"O_D34_ARM_ST\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Divert34.Arm.Status.NodeId.NamespaceIndex);
            //Combine 2

serverState.Factory.Conveyors.ConveyorSystemControls.Combine2.HighwayStopper.ProximitySensor.ObjectDete
cted.NodeId =
                new NodeId("\"I_C2_STOPH\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Combine2.HighwayStopper.ProximitySensor.ObjectDete
cted.NodeId.NamespaceIndex);

serverState.Factory.Conveyors.ConveyorSystemControls.Combine2.SatelliteStopper.ProximitySensor.ObjectDe
tected.NodeId =
```

```
                new NodeId("\"I_C2_STOPS\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Combine2.SatelliteStopper.ProximitySensor.ObjectDe
tected.NodeId.NamespaceIndex);

serverState.Factory.Conveyors.ConveyorSystemControls.Combine2.HighwayStopper.PneumaticActuator.Status.N
odeId =
                new NodeId("\"O_C2_STOPH_ON\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Combine2.HighwayStopper.PneumaticActuator.Status.N
odeId.NamespaceIndex);

serverState.Factory.Conveyors.ConveyorSystemControls.Combine2.SatelliteStopper.PneumaticActuator.Status
.NodeId =
                new NodeId("\"O_C2_STOPS_ON\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Combine2.SatelliteStopper.PneumaticActuator.Status
.NodeId.NamespaceIndex);
            serverState.Factory.Conveyors.ConveyorSystemControls.Combine2.Arm.Status.NodeId =
                new NodeId("\"O_C2_ARM_SP\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Combine2.Arm.Status.NodeId.NamespaceIndex);
            //Combine 3
serverState.Factory.Conveyors.ConveyorSystemControls.Combine3.HighwayStopper.ProximitySensor.ObjectDete
cted.NodeId =
                new NodeId("\"I_C3_STOPH\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Combine3.HighwayStopper.ProximitySensor.ObjectDete
cted.NodeId.NamespaceIndex);

serverState.Factory.Conveyors.ConveyorSystemControls.Combine3.SatelliteStopper.ProximitySensor.ObjectDe
tected.NodeId =
                new NodeId("\"I_C3_STOPS\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Combine3.SatelliteStopper.ProximitySensor.ObjectDe
tected.NodeId.NamespaceIndex);

serverState.Factory.Conveyors.ConveyorSystemControls.Combine3.HighwayStopper.PneumaticActuator.Status.N
odeId =
                new NodeId("\"O_C3_STOPH_ON\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Combine3.HighwayStopper.PneumaticActuator.Status.N
odeId.NamespaceIndex);

serverState.Factory.Conveyors.ConveyorSystemControls.Combine3.SatelliteStopper.PneumaticActuator.Status
.NodeId =
                new NodeId("\"O_C3_STOPS_ON\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Combine3.SatelliteStopper.PneumaticActuator.Status
.NodeId.NamespaceIndex);
            serverState.Factory.Conveyors.ConveyorSystemControls.Combine3.Arm.Status.NodeId =
                new NodeId("\"O_C3_ARM_SP\"",
serverState.Factory.Conveyors.ConveyorSystemControls.Combine3.Arm.Status.NodeId.NamespaceIndex);
            //GStop
serverState.Factory.Conveyors.ConveyorSystemControls.GStop.PneumaticActuator.Status.NodeId =
                new NodeId("\"O_GSTOP_ON\"",
serverState.Factory.Conveyors.ConveyorSystemControls.GStop.PneumaticActuator.Status.NodeId.NamespaceInd
ex);
            //Highway
            serverState.Factory.Conveyors.ConveyorBelts.Highway.Status.NodeId =
                new NodeId("\"HIGHWAY_ON\"",
serverState.Factory.Conveyors.ConveyorBelts.Highway.Status.NodeId.NamespaceIndex);
            serverState.Factory.Conveyors.ConveyorBelts.Highway.Speed.NodeId =
                new NodeId("\"HIGHWAY_SPEED\"",
serverState.Factory.Conveyors.ConveyorBelts.Highway.Speed.NodeId.NamespaceIndex);
            //Satellite 2
            serverState.Factory.Conveyors.ConveyorBelts.Satelite2.Status.NodeId =
                new NodeId("\"SATELLITE2_ON\"",
serverState.Factory.Conveyors.ConveyorBelts.Satelite2.Status.NodeId.NamespaceIndex);
            serverState.Factory.Conveyors.ConveyorBelts.Satelite2.Speed.NodeId =
                new NodeId("\"SATELLITE2_SPEED\"",
serverState.Factory.Conveyors.ConveyorBelts.Satelite2.Speed.NodeId.NamespaceIndex);
            //Satellite 3
            serverState.Factory.Conveyors.ConveyorBelts.Satelite3.Status.NodeId =
```

```
                new NodeId("\"SATELLITE3_ON\"",
serverState.Factory.Conveyors.ConveyorBelts.Satelite3.Status.NodeId.NamespaceIndex);
            serverState.Factory.Conveyors.ConveyorBelts.Satelite3.Speed.NodeId =
                new NodeId("\"SATELLITE3_SPEED\"",
serverState.Factory.Conveyors.ConveyorBelts.Satelite3.Speed.NodeId.NamespaceIndex);
            /// <summary>
            /// M+O System
            /// </summary>
            // XY_MD02 Sensor #1
            serverState.MOSystem.XYMD02Sensor1.HumiditySensor.Measurement.NodeId =
                new NodeId("\"SENSOR1_HUM\"",
serverState.MOSystem.XYMD02Sensor1.HumiditySensor.Measurement.NodeId.NamespaceIndex);
            serverState.MOSystem.XYMD02Sensor1.TemperatureSensor.Measurement.NodeId =
                new NodeId("\"SENSOR1_TEMP\"",
serverState.MOSystem.XYMD02Sensor1.TemperatureSensor.Measurement.NodeId.NamespaceIndex);
            // XY_MD02 Sensor #2
            serverState.MOSystem.XYMD02Sensor2.HumiditySensor.Measurement.NodeId =
                new NodeId("\"SENSOR2_HUM\"",
serverState.MOSystem.XYMD02Sensor1.HumiditySensor.Measurement.NodeId.NamespaceIndex);
            serverState.MOSystem.XYMD02Sensor2.TemperatureSensor.Measurement.NodeId =
                new NodeId("\"SENSOR2_TEMP\"",
serverState.MOSystem.XYMD02Sensor1.TemperatureSensor.Measurement.NodeId.NamespaceIndex);
        }
    }
}
```

LoadConfiguration.cs

```
namespace ICAIFactoryServer
{
    public class LoadConfiguration
    {
        // Folders users will interact with
        static string configurationDirectory = @"Files\Configuration";
        static string newConfigPath = @"Files\Configuration\new_config.txt";
        static string currentConfigPath =
          @"Files\Configuration\current_config.txt";

        /// <summary>
        /// Creates the default configuration file with the saffest settings.
        /// </summary>
        public static void CreateDefaultConfig()
        {
            // Default configuration values
            string serverPort = "26543";
            string securityPolicy = "Basic256Sha256";
            string securityMode = "SignAndEncrypt_3";
            string userSetting = "UserName_1";
            string certificatePolicy = "false";

            // Create the content for the configuration file
            string content = $"Server Port: {serverPort}{Environment.NewLine}" +
                             $"Security Policies:
                                  {securityPolicy}{Environment.NewLine}" +
                             $"Security Mode:
                                    {securityMode}{Environment.NewLine}" +
                             $"User Settings: {userSetting}{Environment.NewLine}"
                             +
                             $"Certificate Policy: {certificatePolicy}";

            // Write the content to the currentConfigPath and newConfigPath files
            File.WriteAllText(currentConfigPath, content);
            File.WriteAllText(newConfigPath, content);
        }
```

```
/// <summary>
/// Updates the configuration file based on the new configuration
    settings.
/// </summary>
public static void UpdateConfigurationFile()
{
    // Read the content of the new_Config.txt file
    string newConfigContent = File.ReadAllText(newConfigPath);
    File.WriteAllText(currentConfigPath, newConfigContent);

    // Split the content by line
    string[] lines = newConfigContent.Split(new[] { Environment.NewLine
        }, StringSplitOptions.RemoveEmptyEntries);

    // Create variables to store the policies, modes, user settings,
          server's port and certificate settings
    List<string> securityPolicies = new List<string>();
    List<string> securityModes = new List<string>();
    List<string> userSettings = new List<string>();
    string portNumber = "";
    string certificatePolicy = "";

    // Process each line to extract the policies, modes, user settings,
          server's port and certificate settings
    foreach (string line in lines)
    {
        if (line.StartsWith("Security Policies:"))
        {
            string policies = line.Replace("Security Policies:",
                 "").Trim();
            securityPolicies = policies.Split(',').Select(p =>
                 p.Trim()).ToList();
        }
        else if (line.StartsWith("Security Mode:"))
        {
            string modes = line.Replace("Security Mode:", "").Trim();
            securityModes = modes.Split(',').Select(m =>
                 m.Trim()).ToList();
        }
        else if (line.StartsWith("User Settings:"))
        {
            string user = line.Replace("User Settings:", "").Trim();
            userSettings = user.Split(',').Select(m =>
                 m.Trim()).ToList();
        }
        else if (line.StartsWith("Server Port:"))
        {
            string port = line.Replace("Server Port:", "").Trim();
            portNumber = port;
            if (string.IsNullOrEmpty(portNumber))
            {
                portNumber = "26543";
            }
        }
        else if (line.StartsWith("Certificate Policy:"))
        {
            // Extract certificate policy
            string policy = line.Replace("Certificate Policy:",
                 "").Trim();
            certificatePolicy = policy;
            if (string.IsNullOrEmpty(policy))
            {
                certificatePolicy = "false";
            }
        }
    }
```

```
        // Read the content of OPCUAServer.ConfigBase.xml
        string configContent = File.ReadAllText(configurationDirectory +
          @"\OPCUAServer.ConfigBase.xml");

        // Update the SecurityPolicies section in the configContent
        int startIndex = configContent.IndexOf("<SecurityPolicies>");
        int endIndex = configContent.IndexOf("</SecurityPolicies>") +
            "</SecurityPolicies>".Length;
        configContent = configContent.Remove(startIndex, endIndex -
            startIndex);
        string content = GenerateSecurityPoliciesContent(securityPolicies,
            securityModes);
        string modifiedContent = configContent.Insert(startIndex, content);
        modifiedContent = modifiedContent.Replace("\r\n", "\n").Replace("\n",
            Environment.NewLine);

        // Update the UserTokenPolicies section in the modifiedContent
        startIndex = modifiedContent.IndexOf("<UserTokenPolicies>");
        endIndex = modifiedContent.IndexOf("</UserTokenPolicies>") +
            "</UserTokenPolicies>".Length;
        configContent = modifiedContent.Remove(startIndex, endIndex -
            startIndex);
        content = GenerateUserSettingsContent(userSettings);
        modifiedContent = configContent.Insert(startIndex, content);
        modifiedContent = modifiedContent.Replace("\r\n", "\n").Replace("\n",
            Environment.NewLine);

        // Update the BaseAddresses section in the modifiedContent
        startIndex = modifiedContent.IndexOf("<BaseAddresses>");
        endIndex = modifiedContent.IndexOf("</BaseAddresses>") +
            "</BaseAddresses>".Length;
        configContent = modifiedContent.Remove(startIndex, endIndex -
            startIndex);
        content = GeneratePortAddressContent(portNumber);
        modifiedContent = configContent.Insert(startIndex, content);
        modifiedContent = modifiedContent.Replace("\r\n", "\n").Replace("\n",
            Environment.NewLine);


        // Update the AutoAcceptUntrustedCertificates section in the
            modifiedContent
        startIndex =
            modifiedContent.IndexOf("<AutoAcceptUntrustedCertificates>");
        endIndex =
            modifiedContent.IndexOf("</AutoAcceptUntrustedCertificates>") +
            "</AutoAcceptUntrustedCertificates>".Length;
        configContent = modifiedContent.Remove(startIndex, endIndex -
            startIndex);
        content = GenerateCertificatePolicyContent(certificatePolicy);
        modifiedContent = configContent.Insert(startIndex, content);
        modifiedContent = modifiedContent.Replace("\r\n", "\n").Replace("\n",
            Environment.NewLine);

        // Write the modifiedContent to OPCUAServer.Config.xml
        File.WriteAllText(configurationDirectory +
            @"\OPCUAServer.Config.xml", modifiedContent);

    }

    private static string GenerateSecurityPoliciesContent(List<string>
      policies, List<string> modes)
    {
        StringBuilder stringBuilder = new StringBuilder();
        stringBuilder.AppendLine("<SecurityPolicies>");
        int count = Math.Max(policies.Count, modes.Count);

        for (int i = 0; i < modes.Count; i++)
```

```
        {
            if (modes[i] == "None_1")
            {
                stringBuilder.AppendLine("      <ServerSecurityPolicy>");
                stringBuilder.AppendLine("
                     <SecurityMode>None_1</SecurityMode>");
                stringBuilder.AppendLine("
                    <SecurityPolicyUri>http://opcfoundation.org/UA/SecurityPolicy#
                    None</SecurityPolicyUri>");
                stringBuilder.AppendLine("
                     <SecurityLevel>0</SecurityLevel>");
                stringBuilder.AppendLine("      </ServerSecurityPolicy>");
            }
            else
            {
                for (int j = 0; j < policies.Count; j++)
                {
                    stringBuilder.AppendLine("      <ServerSecurityPolicy>");
                    stringBuilder.AppendLine($"
                         <SecurityMode>{modes[i]}</SecurityMode>");
                    stringBuilder.AppendLine($"
                            <SecurityPolicyUri>{GetPolicyUri(policies[j])}</Security
                            PolicyUri>");
                    stringBuilder.AppendLine("
                         <SecurityLevel>0</SecurityLevel>");
                    stringBuilder.AppendLine("
                         </ServerSecurityPolicy>");
                }
            }
        }
        stringBuilder.AppendLine("    </SecurityPolicies>");
        return stringBuilder.ToString();
    }

    private static string GetPolicyUri(string policy)
    {
        switch (policy)
        {
            case "None":
                return "http://opcfoundation.org/UA/SecurityPolicy#None";
            case "Basic128Rsa15":
                return
                    "http://opcfoundation.org/UA/SecurityPolicy#Basic128Rsa15";
            case "Basic256":
                return "http://opcfoundation.org/UA/SecurityPolicy#Basic256";
            case "Basic256Sha256":
                return
                     "http://opcfoundation.org/UA/SecurityPolicy#Basic256Sha256";
            default:
                throw new ArgumentException($"Unknown security policy:
                     {policy}");
        }
    }
    private static string GenerateUserSettingsContent(List<string>
      userTokens)
    {
        StringBuilder stringBuilder = new StringBuilder();
        stringBuilder.AppendLine("<UserTokenPolicies>");

        for (int i = 0; i < userTokens.Count; i++)
        {
            stringBuilder.AppendLine("      <ua:UserTokenPolicy>");
            stringBuilder.AppendLine($"
               <ua:TokenType>{userTokens[i]}</ua:TokenType>");
            stringBuilder.AppendLine($"
            <ua:SecurityPolicyUri>{GetUserPolicyUri(userTokens[i])}</ua:SecurityPolicyUri>");
            stringBuilder.AppendLine("      </ua:UserTokenPolicy>");
```

```
            }
            stringBuilder.AppendLine("    </UserTokenPolicies>");
            return stringBuilder.ToString();
        }
        private static string GetUserPolicyUri(string userToken)
        {
            switch (userToken)
            {
                case "Anonymous_0":
                    return "http://opcfoundation.org/UA/SecurityPolicy#None";
                case "UserName_1":
                    return "http://opcfoundation.org/UA/SecurityPolicy#Basic256";
                case "Certificate_2":
                    return "http://opcfoundation.org/UA/SecurityPolicy#Basic256";
                default:
                    throw new ArgumentException($"Unknown security policy:
                         {userToken}");
            }
        }

        private static string GeneratePortAddressContent(string portNumber)
        {
            StringBuilder stringBuilder = new StringBuilder();
            stringBuilder.AppendLine("<BaseAddresses>");
            stringBuilder.AppendLine($"
                <ua:String>opc.tcp://localhost:{portNumber}/ICAIFactoryServer</ua:St
                ring>");
            stringBuilder.AppendLine("    </BaseAddresses>");
            return stringBuilder.ToString();
        }
        private static string GenerateCertificatePolicyContent(string
         certificatePolicy)
        {
            StringBuilder stringBuilder = new StringBuilder();
                stringBuilder.AppendLine($"<AutoAcceptUntrustedCertificates>{
                certificatePolicy}</AutoAcceptUntrustedCertificates>");
            return stringBuilder.ToString();
        }
    }
}
```

UserDatabase.cs

```
namespace ICAIFactoryServer
{
    internal class UserDatabase
    {
        private static UserDatabase instance;
        private readonly Dictionary<string, string> users = new
          Dictionary<string, string>();
        private static ConnectionLog log;

        // Private constructor to prevent external instantiation
        private UserDatabase()
        {
            // Initialization code
            log = ConnectionLog.GetInstance();
            log.ClearLogs();
        }

        // Public method to access the single instance
        public static UserDatabase GetInstance()
        {
            if (instance == null)
            {
```

```
            instance = new UserDatabase();
        }
        return instance;
    }

    /// <summary>
    /// Adds a new user account with a username and password.
    /// </summary>
    /// <param name="username">The username of the user.</param>
    /// <param name="password">The password of the user.</param>
    public void AddUser(string username, string password)
    {
        users[username] = password;
    }

    /// <summary>
    /// Removes a user with the specified username from the database.
    /// </summary>
    /// <param name="username">The username of the user to remove.</param>
    public void RemoveUser(string username)
    {
        users.Remove(username);
    }

    /// <summary>
    /// Authenticates a user with the specified username and password.
    /// </summary>
    /// <param name="username">The username of the user to
      authenticate.</param>
    /// <param name="password">The password of the user.</param>
    /// <returns>True if the authentication is successful, false
      otherwise.</returns>
    public bool AuthenticateUser(string username, string password)
    {
        if (users.TryGetValue(username, out string storedPassword))
        {
            // Check if both passwords are empty
            if (string.IsNullOrEmpty(storedPassword) &&
                    string.IsNullOrEmpty(password))
            {
                return true;
            }

            return password == storedPassword;
        }

        return false;
    }

    /// <summary>
    /// Logs a successful login attempt by a user.
    /// </summary>
    /// <param name="username">The username of the user who logged in
      successfully.</param>
    public void GoodAccess(string username)
    {
        string logMessage = $"Successful login attempt by user '{username}'
            at {DateTime.Now}.";

        if (log.IsMonitoringAllowed())
        {
            log.WriteLog(logMessage);
        }
    }

    /// <summary>
    /// Checks if a username already exists in the user database.
```

```
        /// </summary>
        /// <param name="username">The username to check.</param>
        /// <returns>True if the username exists, false otherwise.</returns>
        public bool CheckUserName(string username)
        {
            return users.ContainsKey(username);
        }

        /// <summary>
        /// Retrieves all users from the user database.
        /// </summary>
        /// <returns>A dictionary containing all the users.</returns>
        public Dictionary<string, string> GetAllUsers()
        {
            // Use the singleton instance's users dictionary
            return users;
        }

        /// <summary>
        /// Logs a failed login attempt by a user.
        /// </summary>
        /// <param name="username">The username of the user who made the failed
            attempt.</param>
        public void BadAccess(string username)
        {
            string logMessage;

            if (CheckUserName(username))
            {
                logMessage = $"Failed login attempt by user '{username}' at
                   {DateTime.Now}: incorrect password.";
            }
            else
            {
                logMessage = $"Failed login attempt for non-existent user
                   '{username}' at {DateTime.Now}.";
            }

            if (log.IsMonitoringAllowed())
            {
                log.WriteLog(logMessage);
            }
        }
    }
}
```

ConnectionLog.cs

```
namespace ICAIFactoryServer
{
    internal class ConnectionLog
    {
        private static ConnectionLog instance;
        private bool monitoringAllowed = true;

        // Specify the path to the log.txt file
        private static string logFilePath = @"C:\ProgramData\OPC
          Foundation\Log\ICAIFactoryServer_connectionLog.txt";
        private string logMessage = $"Connection Log: ICAI Factory Server started
          at {DateTime.Now}." +
        "\n--------------------------------------------------------------------------";

        // Private constructor to prevent external instantiation
        private ConnectionLog()
        {
            // Initialization code: Creates a new log file
        }

        // Public method to access the single instance
        public static ConnectionLog GetInstance()
        {
            if (instance == null)
            {
                instance = new ConnectionLog();
            }
            return instance;
        }

        /// <summary>
        /// Clears the existing logs and writes the initial log message.
        /// </summary>
        public void ClearLogs()
        {
            try
            {
                EnsureDirectoryExists();
                File.Delete(logFilePath);
            }
            catch { }
            WriteLog(logMessage);
        }

        /// <summary>
        /// Starts monitoring server login activity.
        /// </summary>
        public void StartMonitoring()
        {
            monitoringAllowed = true;
            Console.WriteLine("Monitoring server login activity... Press Enter to
                stop.\n");
        }

        /// <summary>
        /// Stops monitoring server login activity.
        /// </summary>
        public void StopMonitoring()
        {
            monitoringAllowed = false;
            Console.WriteLine("Server login monitoring has ended.\n");
        }

        /// <summary>
```

```
        /// Checks if monitoring server login activity is allowed.
        /// </summary>
        /// <returns>True if monitoring is allowed, false otherwise.</returns>
        public bool IsMonitoringAllowed()
        {
            return monitoringAllowed;
        }

        /// <summary>
        /// Writes the log message to the log file.
        /// </summary>
        /// <param name="logMessage">The log message to be written.</param>
        public void WriteLog(string logMessage)
        {
            // Write the log message to the log file
            using (StreamWriter writer = new StreamWriter(logFilePath, true))
            {
                writer.WriteLine(logMessage);
            }
        }

        /// <summary>
        /// Ensures that the directory for the log file exists. If it doesn't
              exist, it creates the directory.
        /// </summary>
        private void EnsureDirectoryExists()
        {
            string directoryPath = Path.GetDirectoryName(logFilePath);
            if (!Directory.Exists(directoryPath))
            {
                Directory.CreateDirectory(directoryPath);
            }
        }
    }
}
```

ANNEX V. IOT2050

Specifications

Attribute	Value
CPU Technology	ARM TI AM6548 HS
System Memory Capacity	2GB
System Memory Type	DDR4
Number of COM Ports	1
Series	IOT2050 Advanced
Communication Protocol	Ethernet
Media Interfaces	USB 2.0
COM Port Type	1x RS 232 / 422 / 485
Extension	Arduino, mPCIe
On Board I/O	x20 Digital I/O, x6 Analog Inputs
Operating Systems	SIMATIC Industrial OS (preinstalled)
Embedded Features	Battery Buffered RTC, LEDs, Watchdog
Operating Temperature	0 → +50°C

Figure 91. IOT2050 gateway specifications (SIEMENS, 2021).

GATEWAY BOOT UP

The initial step while installing the gateway involves flashing the device with a suitable image, as it does not come pre-installed with any software. This image, obtained from the Siemens Support website, is then written onto the device's storage medium to initialise its operating system and associated software.

The provided image, specifically designed by Siemens, includes a Debian operating system along with all the necessary drivers tailored for the IOT2050 device. Moreover, it encompasses

practical IoT software components like Node-RED and Mosquitto's MQTT broker, which streamline the setup process and offer convenient plug-and-play functionalities. The image file is available for download on the Siemens Support website.

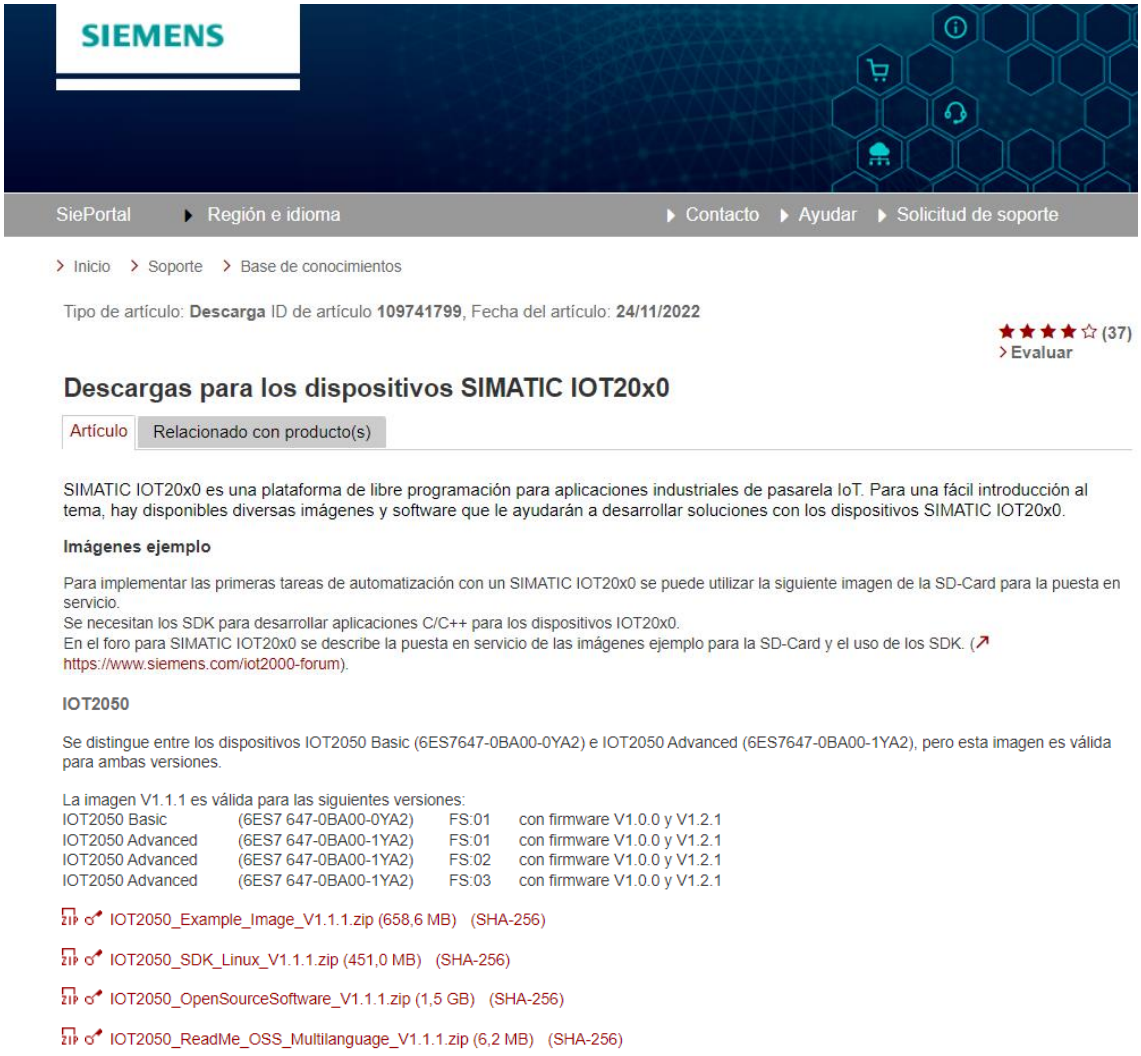


Figure 92. IOT2050 image file from the Siemens Support website.

Once the image file has been obtained, the configuration of the IOT2050 device and the execution of the initial boot-up process begins. To achieve this, the image file needs to be flashed onto a micro-USB using software such as Win32DiskImager. This software provides the capability to write the image file onto the micro-USB, which will be used for configuring

the IOT2050 device. It is essential to format the micro-USB and ensure that it has enough storage capacity to accommodate the image file before proceeding with the flashing process.

To flash the image onto the micro-USB, open Win32DiskImager and select the downloaded image file from the Siemens Support website. Choose the micro-USB as the destination device and click the 'write' button to initiate the flashing process.

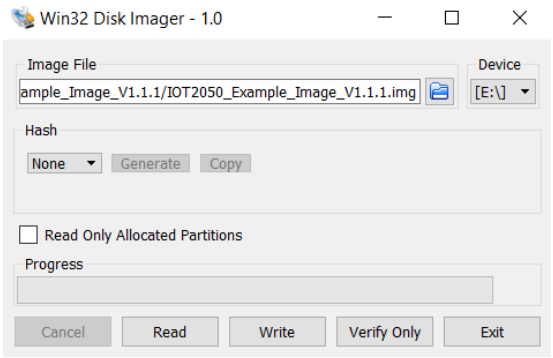


Figure 93. Win32DiskManager flashing process.

Once the flashing process is completed, the micro-USB can be inserted into the IOT2050 device, and the initial boot up process can be performed.

The IOT2050 Gateway features readily accessible UART pins that serve the purpose of serial communication and facilitate the debugging and visualisation of the device's booting process. For this purpose, a UART to USB adapter and jumper wires can be utilised to establish a connection with the RX, TX, and GND pins of the UART interface.

Siemens' Instruction Sheet provides a table that contains the necessary information to identify each pin.

UART Debug

X14

Pin	Assignment
1	GND
2	RTS_N
3	n. c.
4	RxD
5	TxD
6	CTS_N

Figure 94. UART pins of the IOT2050 device (SIEMENS, 2021).

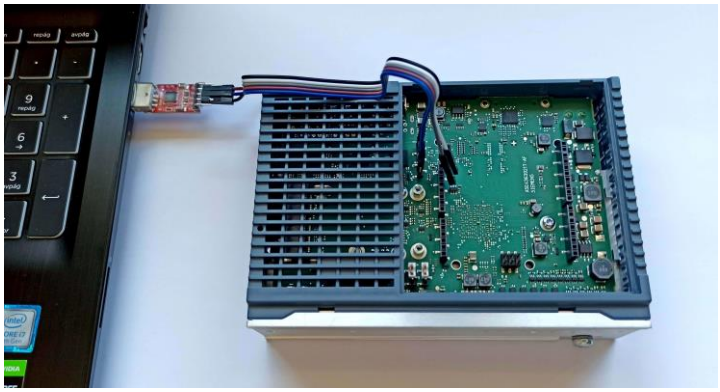


Figure 95. UART to USB adapter set-up.

After the adapter has been connected, a terminal program such as PuTTY[15] can be employed to communicate with the IOT2050 device. This enables the monitoring of the booting process and allows for the detection of any errors or issues that may arise during configuration.

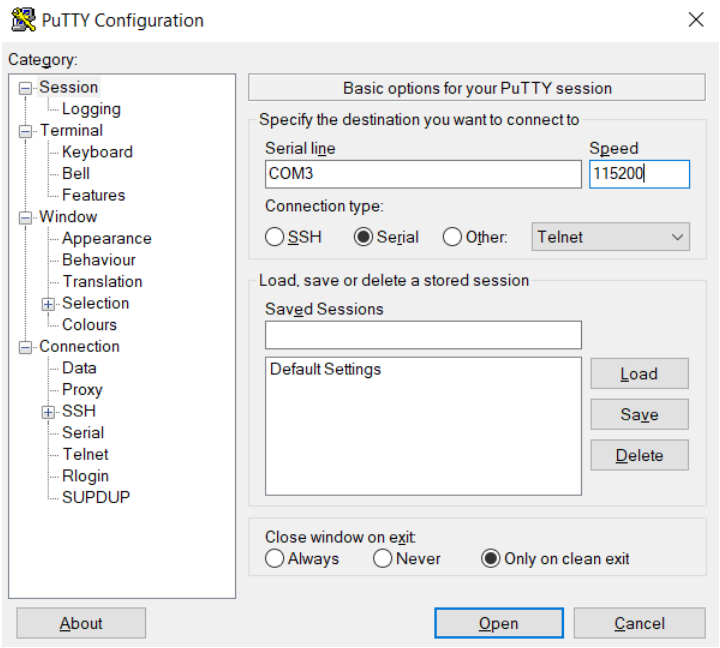


Figure 96. Putty configuration to connect to the UART port.

[15] The baud rate for the IOT2050 is 155200.

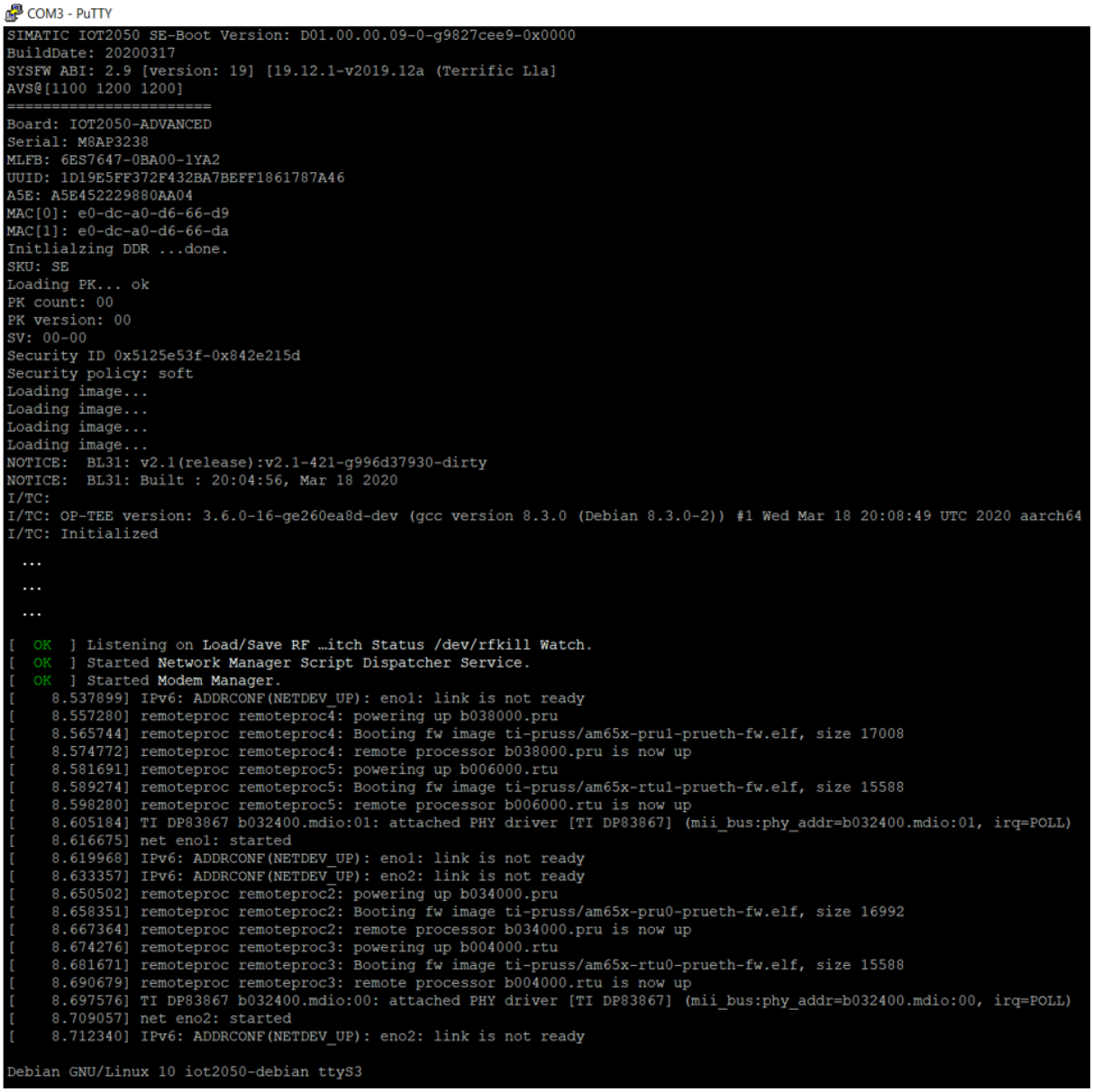


Figure 97. Boot-up process monitored through Putty.

Upon successful booting of the IOT2050 device and its arrival at the login prompt, it is mandatory to change the default password for security reasons during the first setup.

```
Debian GNU/Linux 10 iot2050-debian ttyS3

iot2050-debian login: root
Password:
You are required to change your password immediately (administrator enforced)
Changing password for root.
Current password:
New password:
Retype new password:
Linux iot2050-debian 4.19.94 #1 SMP PREEMPT Wed Sep 1 11:08:36 UTC 2021 aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@iot2050-debian:~# whoami
root
```

Figure 98. New password for the root account of the IOT2050 device.

GATEWAY CONFIGURATION

Upon changing the default password, the UART bootup tool can be disconnected, and a connection to port 192.168.200.1 via SSH can be established. To do so, an Ethernet cable is connected to port P1 and a terminal program such as PuTTY is employed to establish the connection.

With the successful establishment of the SSH connection, the IOT2050 device is remotely configured and managed. This capability affords the user several key functionalities, including the ability to configure network settings, install software packages, and monitor system performance, among others.

Typing the command `iot2050setup`, a setup page is launched displaying all available configuration options for the IOT2050 device.

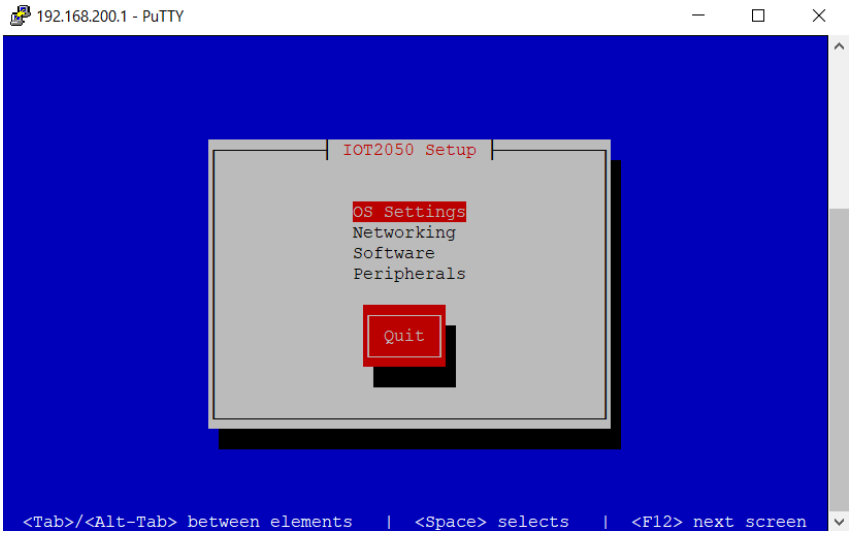


Figure 99. IOT2050 set-up.

Upon accessing the settings menu, various options are presented to configure the system. These options encompass changing usernames and passwords, modifying network settings, managing software packages, and configuring peripherals.

Regarding this project, the most pertinent settings are the last two, namely auto-start configurations and peripheral configurations. By default, the auto-start settings are

preconfigured to launch a Mosquito broker for MQTT communication and Node-RED for flow development. These settings are left as they are for future configuration of the OPC UA clients.

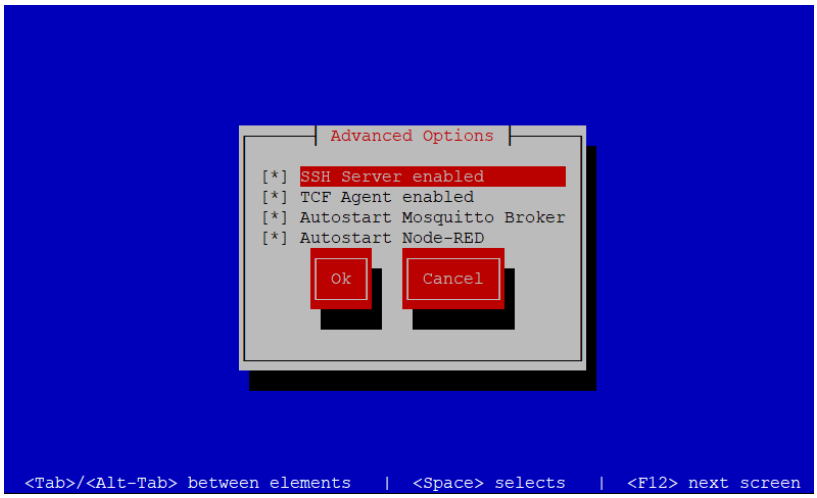


Figure 100. Auto-start settings for the IOT2050 device.

On the other hand, the peripheral menu enables the configuration of the serial port and the Arduino I/O pins of the IoT gateway. The serial port is of particular importance as it will be used to establish a connection with the sensors from the M+O System. To accommodate the use of RS-485 Modbus sensors, configuring the serial port to RS-485 is necessary, although it is also possible to set it up for RS-232 and RS-422 communication.

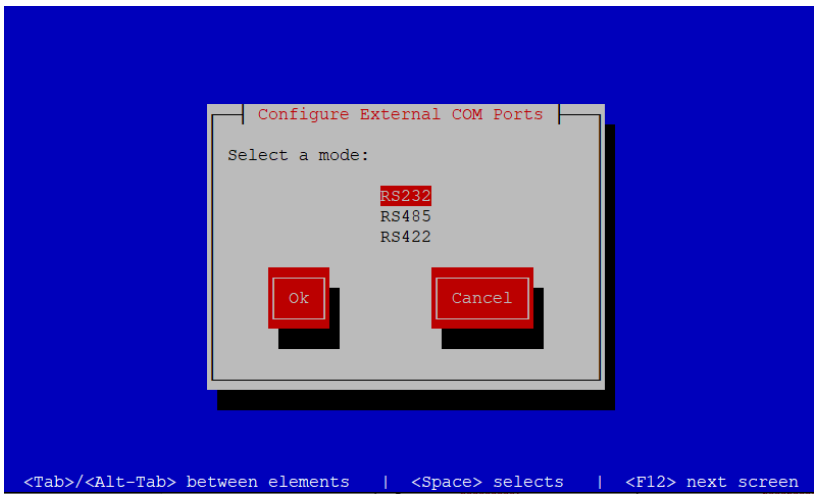


Figure 101. Peripherals settings for the IOT2050 serial port.

ANNEX VI. NIST STUDY

Function	Category	Subcategory	Informative References	Significance for ICAI's factory
	Asset Management (ID.AM): The data, personnel, devices, systems, and facilities that enable the organization to achieve business purposes are identified and managed consistent with their relative importance to organizational objectives and the organization's risk strategy.	**ID.AM-1:** Physical devices and systems within the organization are inventoried	· **CIS CSC** 1 · **COBIT 5** BAI09.01, BAI09.02 · **ISA 62443-2-1:2009** 4.2.3.4 · **ISA 62443-3-3:2013** SR 7.8 · **ISO/IEC 27001:2013** A.8.1.1, A.8.1.2 · **NIST SP 800-53 Rev. 4** CM-8, PM-5	VERY IMPORTANT
		ID.AM-2: Software platforms and applications within the organization are inventoried	· **CIS CSC** 2 · **COBIT 5** BAI09.01, BAI09.02, BAI09.05 · **ISA 62443-2-1:2009** 4.2.3.4 · **ISA 62443-3-3:2013** SR 7.8 · **ISO/IEC 27001:2013** A.8.1.1, A.8.1.2, A.12.5.1 · **NIST SP 800-53 Rev. 4** CM-8, PM-5	VERY IMPORTANT
		ID.AM-3: Organizational communication and data flows are mapped	· **CIS CSC** 12 · **COBIT 5** DSS05.02 · **ISA 62443-2-1:2009** 4.2.3.4 · **ISO/IEC 27001:2013** A.13.2.1, A.13.2.2 · **NIST SP 800-53 Rev. 4** AC-4, CA-3, CA-9, PL-8	VERY IMPORTANT
		ID.AM-4: External information systems are catalogued	· **CIS CSC** 12 · **COBIT 5** APO02.02, APO10.04, DSS01.02 · **ISO/IEC 27001:2013** A.11.2.6 · **NIST SP 800-53 Rev. 4** AC-20, SA-9	IMPORTANT
		ID.AM-5: Resources (e.g., hardware, devices, data, time, personnel, and software) are prioritized based on their classification, criticality, and business value	· **CIS CSC** 13, 14 · **COBIT 5** APO03.03, APO03.04, APO12.01, BAI04.02, BAI09.02 · **ISA 62443-2-1:2009** 4.2.3.6 · **ISO/IEC 27001:2013** A.8.2.1 · **NIST SP 800-53 Rev. 4** CP-2, RA-2, SA-14, SC-6	VERY IMPORTANT
		ID.AM-6: Cybersecurity roles and responsibilities for the entire workforce and third-party stakeholders (e.g., suppliers, customers, partners) are established	· **CIS CSC** 17, 19 · **COBIT 5** APO01.02, APO07.06, APO13.01, DSS06.03 · **ISA 62443-2-1:2009** 4.3.2.3.3 · **ISO/IEC 27001:2013** A.6.1.1 · **NIST SP 800-53 Rev. 4** CP-2, PS-7, PM-11	VERY IMPORTANT
	Business Environment (ID.BE): The organization's mission, objectives, stakeholders, and activities are understood and prioritized; this information is used to inform cybersecurity roles, responsibilities, and risk management decisions.	**ID.BE-1:** The organization's role in the supply chain is identified and communicated	· **COBIT 5** APO08.01, APO08.04, APO08.05, APO10.03, APO10.04, APO10.05 · **ISO/IEC 27001:2013** A.15.1.1, A.15.1.2, A.15.1.3, A.15.2.1, A.15.2.2 · **NIST SP 800-53 Rev. 4** CP-2, SA-12	IMPORTANT
		ID.BE-2: The organization's place in critical infrastructure and its industry sector is identified and communicated	· **COBIT 5** APO02.06, APO03.01 · **ISO/IEC 27001:2013** Clause 4.1 · **NIST SP 800-53 Rev. 4** PM-8	IMPORTANT
		ID.BE-3: Priorities for organizational mission, objectives, and activities are established and communicated	· **COBIT 5** APO02.01, APO02.06, APO03.01 · **ISA 62443-2-1:2009** 4.2.2.1, 4.2.3.6 · **NIST SP 800-53 Rev. 4** PM-11, SA-14	MODERATE
		ID.BE-4: Dependencies and critical functions for delivery of critical services are established	· **COBIT 5** APO10.01, BAI04.02, BAI09.02 · **ISO/IEC 27001:2013** A.11.2.2, A.11.2.3, A.12.1.3 · **NIST SP 800-53 Rev. 4** CP-8, PE-9, PE-11, PM-8, SA-14	MODERATE
		ID.BE-5: Resilience requirements to support delivery of critical services are established for all operating states (e.g. under duress/attack, during recovery, normal operations)	· **COBIT 5** BAI03.02, DSS04.02 · **ISO/IEC 27001:2013** A.11.1.4, A.17.1.1, A.17.1.2, A.17.2.1 · **NIST SP 800-53 Rev. 4** CP-2, CP-11, SA-13, SA-14	IMPORTANT
		ID.GV-1: Organizational cybersecurity policy is established and communicated	· **CIS CSC** 19 · **COBIT 5** APO01.03, APO13.01, EDM01.01, EDM01.02 · **ISA 62443-2-1:2009** 4.3.2.6 · **ISO/IEC 27001:2013** A.5.1.1 · **NIST SP 800-53 Rev. 4** -1 controls from all security control families	IMPORTANT

IDENTIFY (ID)	**Governance (ID.GV):** The policies, procedures, and processes to manage and monitor the organization's regulatory, legal, risk, environmental, and operational requirements are understood and inform the management of cybersecurity risk.	**ID.GV-2:** Cybersecurity roles and responsibilities are coordinated and aligned with internal roles and external partners	· **COBIT 5** APO01.02, APO10.03, APO13.02, DSS05.04 · **ISA 62443-2-1:2009** 4.3.2.3.3 · **ISO/IEC 27001:2013** A.6.1.1, A.7.2.1, A.15.1.1 · **NIST SP 800-53 Rev. 4** PS-7, PM-1, PM-2	IMPORTANT
		ID.GV-3: Legal and regulatory requirements regarding cybersecurity, including privacy and civil liberties obligations, are understood and managed	· **CIS CSC** 19 · **COBIT 5** BAI02.01, MEA03.01, MEA03.04 · **ISA 62443-2-1:2009** 4.4.3.7 · **ISO/IEC 27001:2013** A.18.1.1, A.18.1.2, A.18.1.3, A.18.1.4, A.18.1.5 · **NIST SP 800-53 Rev. 4** -1 controls from all security control families	MODERATE
		ID.GV-4: Governance and risk management processes address cybersecurity risks	· **COBIT 5** EDM03.02, APO12.02, APO12.05, DSS04.02 · **ISA 62443-2-1:2009** 4.2.3.1, 4.2.3.3, 4.2.3.8, 4.2.3.9, 4.2.3.11, 4.3.2.4.3, 4.3.2.6.3 · **ISO/IEC 27001:2013** Clause 6 · **NIST SP 800-53 Rev. 4** SA-2, PM-3, PM-7, PM-9, PM-10, PM-11	IMPORTANT
	Risk Assessment (ID.RA): The organization understands the cybersecurity risk to organizational operations (including mission, functions, image, or reputation), organizational assets, and individuals.	**ID.RA-1:** Asset vulnerabilities are identified and documented	· **CIS CSC** 4 · **COBIT 5** APO12.01, APO12.02, APO12.03, APO12.04, DSS05.01, DSS05.02 · **ISA 62443-2-1:2009** 4.2.3, 4.2.3.7, 4.2.3.9, 4.2.3.12 · **ISO/IEC 27001:2013** A.12.6.1, A.18.2.3 · **NIST SP 800-53 Rev. 4** CA-2, CA-7, CA-8, RA-3, RA-5, SA-5, SA-11, SI-2, SI-4, SI-5	VERY IMPORTANT
		ID.RA-2: Cyber threat intelligence is received from information sharing forums and sources	· **CIS CSC** 4 · **COBIT 5** BAI08.01 · **ISA 62443-2-1:2009** 4.2.3, 4.2.3.9, 4.2.3.12 · **ISO/IEC 27001:2013** A.6.1.4 · **NIST SP 800-53 Rev. 4** SI-5, PM-15, PM-16	IMPORTANT
		ID.RA-3: Threats, both internal and external, are identified and documented	· **CIS CSC** 4 · **COBIT 5** APO12.01, APO12.02, APO12.03, APO12.04 · **ISA 62443-2-1:2009** 4.2.3, 4.2.3.9, 4.2.3.12 · **ISO/IEC 27001:2013** Clause 6.1.2 · **NIST SP 800-53 Rev. 4** RA-3, SI-5, PM-12, PM-16	VERY IMPORTANT
		ID.RA-4: Potential business impacts and likelihoods are identified	· **CIS CSC** 4 · **COBIT 5** DSS04.02 · **ISA 62443-2-1:2009** 4.2.3, 4.2.3.9, 4.2.3.12 · **ISO/IEC 27001:2013** A.16.1.6, Clause 6.1.2 · **NIST SP 800-53 Rev. 4** RA-2, RA-3, SA-14, PM-9, PM-11	VERY IMPORTANT
		ID.RA-5: Threats, vulnerabilities, likelihoods, and impacts are used to determine risk	· **CIS CSC** 4 · **COBIT 5** APO12.02 · **ISO/IEC 27001:2013** A.12.6.1 · **NIST SP 800-53 Rev. 4** RA-2, RA-3, PM-16	VERY IMPORTANT
		ID.RA-6: Risk responses are identified and prioritized	· **CIS CSC** 4 · **COBIT 5** APO12.05, APO13.02 · **ISO/IEC 27001:2013** Clause 6.1.3 · **NIST SP 800-53 Rev. 4** PM-4, PM-9	VERY IMPORTANT
	Risk Management Strategy (ID.RM): The organization's priorities, constraints, risk tolerances, and assumptions are established and used to support operational risk decisions.	**ID.RM-1:** Risk management processes are established, managed, and agreed to by organizational stakeholders	· **CIS CSC** 4 · **COBIT 5** APO12.04, APO12.05, APO13.02, BAI02.03, BAI04.02 · **ISA 62443-2-1:2009** 4.3.4.2 · **ISO/IEC 27001:2013** Clause 6.1.3, Clause 8.3, Clause 9.3 · **NIST SP 800-53 Rev. 4** PM-9	IMPORTANT
		ID.RM-2: Organizational risk tolerance is determined and clearly expressed	· **COBIT 5** APO12.06 · **ISA 62443-2-1:2009** 4.3.2.6.5 · **ISO/IEC 27001:2013** Clause 6.1.3, Clause 8.3 · **NIST SP 800-53 Rev. 4** PM-9	IMPORTANT

		ID.RM-3: The organization's determination of risk tolerance is informed by its role in critical infrastructure and sector specific risk analysis	· **ISO/IEC 27001:2013** Clause 6.1.3, Clause 8.3 · **NIST SP 800-53 Rev. 4** SA-14, PM-8, PM-9, PM-11	MODERATE
	Supply Chain Risk Management (ID.SC): The organization's priorities, constraints, risk tolerances, and assumptions are established and used to support risk decisions associated with managing supply chain risk. The organization has established and implemented the processes to identify, assess and manage supply chain risks.	**ID.SC-1:** Cyber supply chain risk management processes are identified, established, assessed, managed, and agreed to by organizational stakeholders	· **CIS CSC** 4 · **COBIT 5** APO10.01, APO10.04, APO12.04, APO12.05, APO13.02, BAI01.03, BAI02.03, BAI04.02 · **ISA 62443-2-1:2009** 4.3.4.2 · **ISO/IEC 27001:2013** A.15.1.1, A.15.1.2, A.15.1.3, A.15.2.1, A.15.2.2 · **NIST SP 800-53 Rev. 4** SA-9, SA-12, PM-9	IMPORTANT
		ID.SC-2: Suppliers and third party partners of information systems, components, and services are identified, prioritized, and assessed using a cyber supply chain risk assessment process	· **COBIT 5** APO10.01, APO10.02, APO10.04, APO10.05, APO12.01, APO12.02, APO12.03, APO12.04, APO12.05, APO12.06, APO13.02, BAI02.03 · **ISA 62443-2-1:2009** 4.2.3.1, 4.2.3.2, 4.2.3.3, 4.2.3.4, 4.2.3.6, 4.2.3.8, 4.2.3.9, 4.2.3.10, 4.2.3.12, 4.2.3.13, 4.2.3.14 · **ISO/IEC 27001:2013** A.15.2.1, A.15.2.2 · **NIST SP 800-53 Rev. 4** RA-2, RA-3, SA-12, SA-14, SA-15, PM-9	MODERATE
		ID.SC-3: Contracts with suppliers and third-party partners are used to implement appropriate measures designed to meet the objectives of an organization's cybersecurity program and Cyber Supply Chain Risk Management Plan.	· **COBIT 5** APO10.01, APO10.02, APO10.03, APO10.04, APO10.05 · **ISA 62443-2-1:2009** 4.3.2.6.4, 4.3.2.6.7 · **ISO/IEC 27001:2013** A.15.1.1, A.15.1.2, A.15.1.3 · **NIST SP 800-53 Rev. 4** SA-9, SA-11, SA-12, PM-9	IMPORTANT
		ID.SC-4: Suppliers and third-party partners are routinely assessed using audits, test results, or other forms of evaluations to confirm they are meeting their contractual obligations.	· **COBIT 5** APO10.01, APO10.03, APO10.04, APO10.05, MEA01.01, MEA01.02, MEA01.03, MEA01.04, MEA01.05 · **ISA 62443-2-1:2009** 4.3.2.6.7 · **ISA 62443-3-3:2013** SR 6.1 · **ISO/IEC 27001:2013** A.15.2.1, A.15.2.2 · **NIST SP 800-53 Rev. 4** AU-2, AU-6, AU-12, AU-16, PS-7, SA-9, SA-12	MODERATE
		ID.SC-5: Response and recovery planning and testing are conducted with suppliers and third-party providers	· **CIS CSC** 19, 20 · **COBIT 5** DSS04.04 · **ISA 62443-2-1:2009** 4.3.2.5.7, 4.3.4.5.11 · **ISA 62443-3-3:2013** SR 2.8, SR 3.3, SR.6.1, SR 7.3, SR 7.4 · **ISO/IEC 27001:2013** A.17.1.3 · **NIST SP 800-53 Rev. 4** CP-2, CP-4, IR-3, IR-4, IR-6, IR-8, IR-9	MODERATE
		PR.AC-1: Identities and credentials are issued, managed, verified, revoked, and audited for authorized devices, users and processes	· **CIS CSC** 1, 5, 15, 16 · **COBIT 5** DSS05.04, DSS06.03 · **ISA 62443-2-1:2009** 4.3.3.5.1 · **ISA 62443-3-3:2013** SR 1.1, SR 1.2, SR 1.3, SR 1.4, SR 1.5, SR 1.7, SR 1.8, SR 1.9 · **ISO/IEC 27001:2013** A.9.2.1, A.9.2.2, A.9.2.3, A.9.2.4, A.9.2.6, A.9.3.1, A.9.4.2, A.9.4.3 · **NIST SP 800-53 Rev. 4** AC-1, AC-2, IA-1, IA-2, IA-3, IA-4, IA-5, IA-6, IA-7, IA-8, IA-9, IA-10, IA-11	VERY IMPORTANT
		PR.AC-2: Physical access to assets is managed and protected	· **COBIT 5** DSS01.04, DSS05.05 · **ISA 62443-2-1:2009** 4.3.3.3.2, 4.3.3.3.8 · **ISO/IEC 27001:2013** A.11.1.1, A.11.1.2, A.11.1.3, A.11.1.4, A.11.1.5, A.11.1.6, A.11.2.1, A.11.2.3, A.11.2.5, A.11.2.6, A.11.2.7, A.11.2.8 · **NIST SP 800-53 Rev. 4** PE-2, PE-3, PE-4, PE-5, PE-6, PE-8	VERY IMPORTANT
		PR.AC-3: Remote access is managed	· **CIS CSC** 12 · **COBIT 5** APO13.01, DSS01.04, DSS05.03 · **ISA 62443-2-1:2009** 4.3.3.6.6 · **ISA 62443-3-3:2013** SR 1.13, SR 2.6	VERY IMPORTANT

			· **ISO/IEC 27001:2013** A.6.2.1, A.6.2.2, A.11.2.6, A.13.1.1, A.13.2.1 · **NIST SP 800-53 Rev. 4** AC-1, AC-17, AC-19, AC-20, SC-15	
	Identity Management, Authentication and Access Control (PR.AC): Access to physical and logical assets and associated facilities is limited to authorized users, processes, and devices, and is managed consistent with the assessed risk of unauthorized access to authorized activities and transactions.	**PR.AC-4:** Access permissions and authorizations are managed, incorporating the principles of least privilege and separation of duties	· **CIS CSC** 3, 5, 12, 14, 15, 16, 18 · **COBIT 5** DSS05.04 · **ISA 62443-2-1:2009** 4.3.3.7.3 · **ISA 62443-3-3:2013** SR 2.1 · **ISO/IEC 27001:2013** A.6.1.2, A.9.1.2, A.9.2.3, A.9.4.1, A.9.4.4, A.9.4.5 · **NIST SP 800-53 Rev. 4** AC-1, AC-2, AC-3, AC-5, AC-6, AC-14, AC-16, AC-24	VERY IMPORTANT
		PR.AC-5: Network integrity is protected (e.g., network segregation, network segmentation)	· **CIS CSC** 9, 14, 15, 18 · **COBIT 5** DSS01.05, DSS05.02 · **ISA 62443-2-1:2009** 4.3.3.4 · **ISA 62443-3-3:2013** SR 3.1, SR 3.8 · **ISO/IEC 27001:2013** A.13.1.1, A.13.1.3, A.13.2.1, A.14.1.2, A.14.1.3 · **NIST SP 800-53 Rev. 4** AC-4, AC-10, SC-7	IMPORTANT
		PR.AC-6: Identities are proofed and bound to credentials and asserted in interactions	· **CIS CSC**, 16 · **COBIT 5** DSS05.04, DSS05.05, DSS05.07, DSS06.03 · **ISA 62443-2-1:2009** 4.3.3.2.2, 4.3.3.5.2, 4.3.3.7.2, 4.3.3.7.4 · **ISA 62443-3-3:2013** SR 1.1, SR 1.2, SR 1.4, SR 1.5, SR 1.9, SR 2.1 · **ISO/IEC 27001:2013**, A.7.1.1, A.9.2.1 · **NIST SP 800-53 Rev. 4** AC-1, AC-2, AC-3, AC-16, AC-19, AC-24, IA-1, IA-2, IA-4, IA-5, IA-8, PE-2, PS-3	IMPORTANT
		PR.AC-7: Users, devices, and other assets are authenticated (e.g., single-factor, multi-factor) commensurate with the risk of the transaction (e.g., individuals' security and privacy risks and other organizational risks)	· **CIS CSC** 1, 12, 15, 16 · **COBIT 5** DSS05.04, DSS05.10, DSS06.10 · **ISA 62443-2-1:2009** 4.3.3.6.1, 4.3.3.6.2, 4.3.3.6.3, 4.3.3.6.4, 4.3.3.6.5, 4.3.3.6.6, 4.3.3.6.7, 4.3.3.6.8, 4.3.3.6.9 · **ISA 62443-3-3:2013** SR 1.1, SR 1.2, SR 1.5, SR 1.7, SR 1.8, SR 1.9, SR 1.10 · **ISO/IEC 27001:2013** A.9.2.1, A.9.2.4, A.9.3.1, A.9.4.2, A.9.4.3, A.18.1.4 · **NIST SP 800-53 Rev. 4** AC-7, AC-8, AC-9, AC-11, AC-12, AC-14, IA-1, IA-2, IA-3, IA-4, IA-5, IA-8, IA-9, IA-10, IA-11	VERY IMPORTANT
	Awareness and Training (PR.AT): The organization's personnel and partners are provided cybersecurity awareness education and are trained to perform their cybersecurity-related duties and responsibilities consistent with related policies, procedures, and agreements.	**PR.AT-1:** All users are informed and trained	· **CIS CSC** 17, 18 · **COBIT 5** APO07.03, BAI05.07 · **ISA 62443-2-1:2009** 4.3.2.4.2 · **ISO/IEC 27001:2013** A.7.2.2, A.12.2.1 · **NIST SP 800-53 Rev. 4** AT-2, PM-13	VERY IMPORTANT
		PR.AT-2: Privileged users understand their roles and responsibilities	· **CIS CSC** 5, 17, 18 · **COBIT 5** APO07.02, DSS05.04, DSS06.03 · **ISA 62443-2-1:2009** 4.3.2.4.2, 4.3.2.4.3 · **ISO/IEC 27001:2013** A.6.1.1, A.7.2.2 · **NIST SP 800-53 Rev. 4** AT-3, PM-13	VERY IMPORTANT
		PR.AT-3: Third-party stakeholders (e.g., suppliers, customers, partners) understand their roles and responsibilities	· **CIS CSC** 17 · **COBIT 5** APO07.03, APO07.06, APO10.04, APO10.05 · **ISA 62443-2-1:2009** 4.3.2.4.2 · **ISO/IEC 27001:2013** A.6.1.1, A.7.2.1, A.7.2.2 · **NIST SP 800-53 Rev. 4** PS-7, SA-9, SA-16	MODERATE
		PR.AT-4: Senior executives understand their roles and responsibilities	· **CIS CSC** 17, 19 · **COBIT 5** EDM01.01, APO01.02, APO07.03 · **ISA 62443-2-1:2009** 4.3.2.4.2 · **ISO/IEC 27001:2013** A.6.1.1, A.7.2.2	MODERATE

PROTECT (PR)		**PR.AT-5:** Physical and cybersecurity personnel understand their roles and responsibilities	· **CIS CSC** 17 · **COBIT 5** APO07.03 · **ISA 62443-2-1:2009** 4.3.2.4.2 · **ISO/IEC 27001:2013** A.6.1.1, A.7.2.2 · **NIST SP 800-53 Rev. 4** AT-3, IR-2, PM-13	VERY IMPORTANT
	Data Security (PR.DS): Information and records (data) are managed consistent with the organization's risk strategy to protect the confidentiality, integrity, and availability of information.	**PR.DS-1:** Data-at-rest is protected	· **CIS CSC** 13, 14 · **COBIT 5** APO01.06, BAI02.01, BAI06.01, DSS04.07, DSS05.03, DSS06.06 · **ISA 62443-3-3:2013** SR 3.4, SR 4.1 · **ISO/IEC 27001:2013** A.8.2.3 · **NIST SP 800-53 Rev. 4** MP-8, SC-12, SC-28	VERY IMPORTANT
		PR.DS-2: Data-in-transit is protected	· **CIS CSC** 13, 14 · **COBIT 5** APO01.06, DSS05.02, DSS06.06 · **ISA 62443-3-3:2013** SR 3.1, SR 3.8, SR 4.1, SR 4.2 · **ISO/IEC 27001:2013** A.8.2.3, A.13.1.1, A.13.2.1, A.13.2.3, A.14.1.2, A.14.1.3 · **NIST SP 800-53 Rev. 4** SC-8, SC-11, SC-12	VERY IMPORTANT
		PR.DS-3: Assets are formally managed throughout removal, transfers, and disposition	· **CIS CSC** 1 · **COBIT 5** BAI09.03 · **ISA 62443-2-1:2009** 4.3.3.3.9, 4.3.4.4.1 · **ISA 62443-3-3:2013** SR 4.2 · **ISO/IEC 27001:2013** A.8.2.3, A.8.3.1, A.8.3.2, A.8.3.3, A.11.2.5, A.11.2.7 · **NIST SP 800-53 Rev. 4** CM-8, MP-6, PE-16	IMPORTANT
		PR.DS-4: Adequate capacity to ensure availability is maintained	· **CIS CSC** 1, 2, 13 · **COBIT 5** APO13.01, BAI04.04 · **ISA 62443-3-3:2013** SR 7.1, SR 7.2 · **ISO/IEC 27001:2013** A.12.1.3, A.17.2.1 · **NIST SP 800-53 Rev. 4** AU-4, CP-2, SC-5	IMPORTANT
		PR.DS-5: Protections against data leaks are implemented	· **CIS CSC** 13 · **COBIT 5** APO01.06, DSS05.04, DSS05.07, DSS06.02 · **ISA 62443-3-3:2013** SR 5.2 · **ISO/IEC 27001:2013** A.6.1.2, A.7.1.1, A.7.1.2, A.7.3.1, A.8.2.2, A.8.2.3, A.9.1.1, A.9.1.2, A.9.2.3, A.9.4.1, A.9.4.4, A.9.4.5, A.10.1.1, A.11.1.4, A.11.1.5, A.11.2.1, A.13.1.1, A.13.1.3, A.13.2.1, A.13.2.3, A.13.2.4, A.14.1.2, A.14.1.3 · **NIST SP 800-53 Rev. 4** AC-4, AC-5, AC-6, PE-19, PS-3, PS-6, SC-7, SC-8, SC-13, SC-31, SI-4	VERY IMPORTANT
		PR.DS-6: Integrity checking mechanisms are used to verify software, firmware, and information integrity	· **CIS CSC** 2, 3 · **COBIT 5** APO01.06, BAI06.01, DSS06.02 · **ISA 62443-3-3:2013** SR 3.1, SR 3.3, SR 3.4, SR 3.8 · **ISO/IEC 27001:2013** A.12.2.1, A.12.5.1, A.14.1.2, A.14.1.3, A.14.2.4 · **NIST SP 800-53 Rev. 4** SC-16, SI-7	VERY IMPORTANT
		PR.DS-7: The development and testing environment(s) are separate from the production environment	· **CIS CSC** 18, 20 · **COBIT 5** BAI03.08, BAI07.04 · **ISO/IEC 27001:2013** A.12.1.4 · **NIST SP 800-53 Rev. 4** CM-2	VERY IMPORTANT
		PR.DS-8: Integrity checking mechanisms are used to verify hardware integrity	· **COBIT 5** BAI03.05 · **ISA 62443-2-1:2009** 4.3.4.4.4 · **ISO/IEC 27001:2013** A.11.2.4 · **NIST SP 800-53 Rev. 4** SA-10, SI-7	VERY IMPORTANT
			· **CIS CSC** 3, 9, 11 · **COBIT 5** BAI10.01, BAI10.02, BAI10.03, BAI10.05	

	Information Protection Processes and Procedures (PR.IP): Security policies (that address purpose, scope, roles, responsibilities, management commitment, and coordination among organizational entities), processes, and procedures are maintained and used to manage protection of information systems and assets.	**PR.IP-1:** A baseline configuration of information technology/industrial control systems is created and maintained incorporating security principles (e.g. concept of least functionality)	· **ISA 62443-2-1:2009** 4.3.4.3.2, 4.3.4.3.3 · **ISA 62443-3-3:2013** SR 7.6 · **ISO/IEC 27001:2013** A.12.1.2, A.12.5.1, A.12.6.2, A.14.2.2, A.14.2.3, A.14.2.4 · **NIST SP 800-53 Rev. 4** CM-2, CM-3, CM-4, CM-5, CM-6, CM-7, CM-9, SA-10	VERY IMPORTANT
		PR.IP-2: A System Development Life Cycle to manage systems is implemented	· **CIS CSC** 18 · **COBIT 5** APO13.01, BAI03.01, BAI03.02, BAI03.03 · **ISA 62443-2-1:2009** 4.3.4.3.3 · **ISO/IEC 27001:2013** A.6.1.5, A.14.1.1, A.14.2.1, A.14.2.5 · **NIST SP 800-53 Rev. 4** PL-8, SA-3, SA-4, SA-8, SA-10, SA-11, SA-12, SA-15, SA-17, SI-12, SI-13, SI-14, SI-16, SI-17	IMPORTANT
		PR.IP-3: Configuration change control processes are in place	· **CIS CSC** 3, 11 · **COBIT 5** BAI01.06, BAI06.01 · **ISA 62443-2-1:2009** 4.3.4.3.2, 4.3.4.3.3 · **ISA 62443-3-3:2013** SR 7.6 · **ISO/IEC 27001:2013** A.12.1.2, A.12.5.1, A.12.6.2, A.14.2.2, A.14.2.3, A.14.2.4 · **NIST SP 800-53 Rev. 4** CM-3, CM-4, SA-10	IMPORTANT
		PR.IP-4: Backups of information are conducted, maintained, and tested	· **CIS CSC** 10 · **COBIT 5** APO13.01, DSS01.01, DSS04.07 · **ISA 62443-2-1:2009** 4.3.4.3.9 · **ISA 62443-3-3:2013** SR 7.3, SR 7.4 · **ISO/IEC 27001:2013** A.12.3.1, A.17.1.2, A.17.1.3, A.18.1.3 · **NIST SP 800-53 Rev. 4** CP-4, CP-6, CP-9	VERY IMPORTANT
		PR.IP-5: Policy and regulations regarding the physical operating environment for organizational assets are met	· **COBIT 5** DSS01.04, DSS05.05 · **ISA 62443-2-1:2009** 4.3.3.3.1 4.3.3.3.2, 4.3.3.3.3, 4.3.3.3.5, 4.3.3.3.6 · **ISO/IEC 27001:2013** A.11.1.4, A.11.2.1, A.11.2.2, A.11.2.3 · **NIST SP 800-53 Rev. 4** PE-10, PE-12, PE-13, PE-14, PE-15, PE-18	MODERATE
		PR.IP-6: Data is destroyed according to policy	· **COBIT 5** BAI09.03, DSS05.06 · **ISA 62443-2-1:2009** 4.3.4.4.4 · **ISA 62443-3-3:2013** SR 4.2 · **ISO/IEC 27001:2013** A.8.2.3, A.8.3.1, A.8.3.2, A.11.2.7 · **NIST SP 800-53 Rev. 4** MP-6	VERY IMPORTANT
		PR.IP-7: Protection processes are improved	· **COBIT 5** APO11.06, APO12.06, DSS04.05 · **ISA 62443-2-1:2009** 4.4.3.1, 4.4.3.2, 4.4.3.3, 4.4.3.4, 4.4.3.5, 4.4.3.6, 4.4.3.7, 4.4.3.8 · **ISO/IEC 27001:2013** A.16.1.6, Clause 9, Clause 10 · **NIST SP 800-53 Rev. 4** CA-2, CA-7, CP-2, IR-8, PL-2, PM-6	VERY IMPORTANT
		PR.IP-8: Effectiveness of protection technologies is shared	· **COBIT 5** BAI08.04, DSS03.04 · **ISO/IEC 27001:2013** A.16.1.6 · **NIST SP 800-53 Rev. 4** AC-21, CA-7, SI-4	IMPORTANT
		PR.IP-9: Response plans (Incident Response and Business Continuity) and recovery plans (Incident Recovery and Disaster Recovery) are in place and managed	· **CIS CSC** 19 · **COBIT 5** APO12.06, DSS04.03 · **ISA 62443-2-1:2009** 4.3.2.5.3, 4.3.4.5.1 · **ISO/IEC 27001:2013** A.16.1.1, A.17.1.1, A.17.1.2, A.17.1.3 · **NIST SP 800-53 Rev. 4** CP-2, CP-7, CP-12, CP-13, IR-7, IR-8, IR-9, PE-17	VERY IMPORTANT
		PR.IP-10: Response and recovery plans are tested	· **CIS CSC** 19, 20 · **COBIT 5** DSS04.04 · **ISA 62443-2-1:2009** 4.3.2.5.7, 4.3.4.5.11 · **ISA 62443-3-3:2013** SR 3.3 · **ISO/IEC 27001:2013** A.17.1.3	VERY IMPORTANT

		PR.IP-11: Cybersecurity is included in human resources practices (e.g., deprovisioning, personnel screening)	· **CIS CSC** 5, 16 · **COBIT 5** APO07.01, APO07.02, APO07.03, APO07.04, APO07.05 · **ISA 62443-2-1:2009** 4.3.3.2.1, 4.3.3.2.2, 4.3.3.2.3 · **ISO/IEC 27001:2013** A.7.1.1, A.7.1.2, A.7.2.1, A.7.2.2, A.7.2.3, A.7.3.1, A.8.1.4 · **NIST SP 800-53 Rev. 4** PS-1, PS-2, PS-3, PS-4, PS-5, PS-6, PS-7, PS-8, SA-21	MODERATE
		PR.IP-12: A vulnerability management plan is developed and implemented	· **CIS CSC** 4, 18, 20 · **COBIT 5** BAI03.10, DSS05.01, DSS05.02 · **ISO/IEC 27001:2013** A.12.6.1, A.14.2.3, A.16.1.3, A.18.2.2, A.18.2.3 · **NIST SP 800-53 Rev. 4** RA-3, RA-5, SI-2	VERY IMPORTANT
	Maintenance (PR.MA): Maintenance and repairs of industrial control and information system components are performed consistent with policies and procedures.	**PR.MA-1:** Maintenance and repair of organizational assets are performed and logged, with approved and controlled tools	· **COBIT 5** BAI03.10, BAI09.02, BAI09.03, DSS01.05 · **ISA 62443-2-1:2009** 4.3.3.3.7 · **ISO/IEC 27001:2013** A.11.1.2, A.11.2.4, A.11.2.5, A.11.2.6 · **NIST SP 800-53 Rev. 4** MA-2, MA-3, MA-5, MA-6	VERY IMPORTANT
		PR.MA-2: Remote maintenance of organizational assets is approved, logged, and performed in a manner that prevents unauthorized access	· **CIS CSC** 3, 5 · **COBIT 5** DSS05.04 · **ISA 62443-2-1:2009** 4.3.3.6.5, 4.3.3.6.6, 4.3.3.6.7, 4.3.3.6.8 · **ISO/IEC 27001:2013** A.11.2.4, A.15.1.1, A.15.2.1 · **NIST SP 800-53 Rev. 4** MA-4	VERY IMPORTANT
	Protective Technology (PR.PT): Technical security solutions are managed to ensure the security and resilience of systems and assets, consistent with related policies, procedures, and agreements.	**PR.PT-1:** Audit/log records are determined, documented, implemented, and reviewed in accordance with policy	· **CIS CSC** 1, 3, 5, 6, 14, 15, 16 · **COBIT 5** APO11.04, BAI03.05, DSS05.04, DSS05.07, MEA02.01 · **ISA 62443-2-1:2009** 4.3.3.3.9, 4.3.3.5.8, 4.3.4.4.7, 4.4.2.1, 4.4.2.2, 4.4.2.4 · **ISA 62443-3-3:2013** SR 2.8, SR 2.9, SR 2.10, SR 2.11, SR 2.12 · **ISO/IEC 27001:2013** A.12.4.1, A.12.4.2, A.12.4.3, A.12.4.4, A.12.7.1 · **NIST SP 800-53 Rev. 4** AU Family	VERY IMPORTANT
		PR.PT-2: Removable media is protected and its use restricted according to policy	· **CIS CSC** 8, 13 · **COBIT 5** APO13.01, DSS05.02, DSS05.06 · **ISA 62443-3-3:2013** SR 2.3 · **ISO/IEC 27001:2013** A.8.2.1, A.8.2.2, A.8.2.3, A.8.3.1, A.8.3.3, A.11.2.9 · **NIST SP 800-53 Rev. 4** MP-2, MP-3, MP-4, MP-5, MP-7, MP-8	VERY IMPORTANT
		PR.PT-3: The principle of least functionality is incorporated by configuring systems to provide only essential capabilities	· **CIS CSC** 3, 11, 14 · **COBIT 5** DSS05.02, DSS05.05, DSS06.06 · **ISA 62443-2-1:2009** 4.3.3.5.1, 4.3.3.5.2, 4.3.3.5.3, 4.3.3.5.4, 4.3.3.5.5, 4.3.3.5.6, 4.3.3.5.7, 4.3.3.5.8, 4.3.3.6.1, 4.3.3.6.2, 4.3.3.6.3, 4.3.3.6.4, 4.3.3.6.5, 4.3.3.6.6, 4.3.3.6.7, 4.3.3.6.8, 4.3.3.6.9, 4.3.3.7.1, 4.3.3.7.2, 4.3.3.7.3, 4.3.3.7.4 · **ISA 62443-3-3:2013** SR 1.1, SR 1.2, SR 1.3, SR 1.4, SR 1.5, SR 1.6, SR 1.7, SR 1.8, SR 1.9, SR 1.10, SR 1.11, SR 1.12, SR 1.13, SR 2.1, SR 2.2, SR 2.3, SR 2.4, SR 2.5, SR 2.6, SR 2.7 · **ISO/IEC 27001:2013** A.9.1.2 · **NIST SP 800-53 Rev. 4** AC-3, CM-7	MODERATE
		PR.PT-4: Communications and control networks are protected	· **CIS CSC** 8, 12, 15 · **COBIT 5** DSS05.02, APO13.01 · **ISA 62443-3-3:2013** SR 3.1, SR 3.5, SR 3.8, SR 4.1, SR 4.3, SR 5.1, SR 5.2, SR 5.3, SR 7.1, SR 7.6 · **ISO/IEC 27001:2013** A.13.1.1, A.13.2.1, A.14.1.3	VERY IMPORTANT

			· **NIST SP 800-53 Rev. 4** AC-4, AC-17, AC-18, CP-8, SC-7, SC-19, SC-20, SC-21, SC-22, SC-23, SC-24, SC-25, SC-29, SC-32, SC-36, SC-37, SC-38, SC-39, SC-40, SC-41, SC-43	
		PR.PT-5: Mechanisms (e.g., failsafe, load balancing, hot swap) are implemented to achieve resilience requirements in normal and adverse situations	· **COBIT 5** BAI04.01, BAI04.02, BAI04.03, BAI04.04, BAI04.05, DSS01.05 · **ISA 62443-2-1:2009** 4.3.2.5.2 · **ISA 62443-3-3:2013** SR 7.1, SR 7.2 · **ISO/IEC 27001:2013** A.17.1.2, A.17.2.1 · **NIST SP 800-53 Rev. 4** CP-7, CP-8, CP-11, CP-13, PL-8, SA-14, SC-6	VERY IMPORTANT
DETECT (DE)	**Anomalies and Events (DE.AE):** Anomalous activity is detected and the potential impact of events is understood.	**DE.AE-1:** A baseline of network operations and expected data flows for users and systems is established and managed	· **CIS CSC** 1, 4, 6, 12, 13, 15, 16 · **COBIT 5** DSS03.01 · **ISA 62443-2-1:2009** 4.4.3.3 · **ISO/IEC 27001:2013** A.12.1.1, A.12.1.2, A.13.1.1, A.13.1.2 · **NIST SP 800-53 Rev. 4** AC-4, CA-3, CM-2, SI-4	VERY IMPORTANT
		DE.AE-2: Detected events are analyzed to understand attack targets and methods	· **CIS CSC** 3, 6, 13, 15 · **COBIT 5** DSS05.07 · **ISA 62443-2-1:2009** 4.3.4.5.6, 4.3.4.5.7, 4.3.4.5.8 · **ISA 62443-3-3:2013** SR 2.8, SR 2.9, SR 2.10, SR 2.11, SR 2.12, SR 3.9, SR 6.1, SR 6.2 · **ISO/IEC 27001:2013** A.12.4.1, A.16.1.1, A.16.1.4 · **NIST SP 800-53 Rev. 4** AU-6, CA-7, IR-4, SI-4	VERY IMPORTANT
		DE.AE-3: Event data are collected and correlated from multiple sources and sensors	· **CIS CSC** 1, 3, 4, 5, 6, 7, 8, 11, 12, 13, 14, 15, 16 · **COBIT 5** BAI08.02 · **ISA 62443-3-3:2013** SR 6.1 · **ISO/IEC 27001:2013** A.12.4.1, A.16.1.7 · **NIST SP 800-53 Rev. 4** AU-6, CA-7, IR-4, IR-5, IR-8, SI-4	VERY IMPORTANT
		DE.AE-4: Impact of events is determined	· **CIS CSC** 4, 6 · **COBIT 5** APO12.06, DSS03.01 · **ISO/IEC 27001:2013** A.16.1.4 · **NIST SP 800-53 Rev. 4** CP-2, IR-4, RA-3, SI-4	VERY IMPORTANT
		DE.AE-5: Incident alert thresholds are established	· **CIS CSC** 6, 19 · **COBIT 5** APO12.06, DSS03.01 · **ISA 62443-2-1:2009** 4.2.3.10 · **ISO/IEC 27001:2013** A.16.1.4 · **NIST SP 800-53 Rev. 4** IR-4, IR-5, IR-8	VERY IMPORTANT
	Security Continuous Monitoring (DE.CM): The information system and assets are monitored to identify cybersecurity events and verify the effectiveness of protective measures.	**DE.CM-1:** The network is monitored to detect potential cybersecurity events	· **CIS CSC** 1, 7, 8, 12, 13, 15, 16 · **COBIT 5** DSS01.03, DSS03.05, DSS05.07 · **ISA 62443-3-3:2013** SR 6.2 · **NIST SP 800-53 Rev. 4** AC-2, AU-12, CA-7, CM-3, SC-5, SC-7, SI-4	VERY IMPORTANT
		DE.CM-2: The physical environment is monitored to detect potential cybersecurity events	· **COBIT 5** DSS01.04, DSS01.05 · **ISA 62443-2-1:2009** 4.3.3.3.8 · **ISO/IEC 27001:2013** A.11.1.1, A.11.1.2 · **NIST SP 800-53 Rev. 4** CA-7, PE-3, PE-6, PE-20	VERY IMPORTANT
		DE.CM-3: Personnel activity is monitored to detect potential cybersecurity events	· **CIS CSC** 5, 7, 14, 16 · **COBIT 5** DSS05.07 · **ISA 62443-3-3:2013** SR 6.2 · **ISO/IEC 27001:2013** A.12.4.1, A.12.4.3 · **NIST SP 800-53 Rev. 4** AC-2, AU-12, AU-13, CA-7, CM-10, CM-11	VERY IMPORTANT
		DE.CM-4: Malicious code is detected	· **CIS CSC** 4, 7, 8, 12 · **COBIT 5** DSS05.01 · **ISA 62443-2-1:2009** 4.3.4.3.8 · **ISA 62443-3-3:2013** SR 3.2 · **ISO/IEC 27001:2013** A.12.2.1 · **NIST SP 800-53 Rev. 4** SI-3, SI-8	VERY IMPORTANT
		DE.CM-5: Unauthorized mobile code is	· **CIS CSC** 7, 8 · **COBIT 5** DSS05.01 · **ISA 62443-3-3:2013** SR 2.4	

		detected	· **ISO/IEC 27001:2013** A.12.5.1, A.12.6.2 · **NIST SP 800-53 Rev. 4** SC-18, SI-4, SC-44	VERY IMPORTANT
		DE.CM-6: External service provider activity is monitored to detect potential cybersecurity events	· **COBIT 5** APO07.06, APO10.05 · **ISO/IEC 27001:2013** A.14.2.7, A.15.2.1 · **NIST SP 800-53 Rev. 4** CA-7, PS-7, SA-4, SA-9, SI-4	IMPORTANT
		DE.CM-7: Monitoring for unauthorized personnel, connections, devices, and software is performed	· **CIS CSC** 1, 2, 3, 5, 9, 12, 13, 15, 16 · **COBIT 5** DSS05.02, DSS05.05 · **ISO/IEC 27001:2013** A.12.4.1, A.14.2.7, A.15.2.1 · **NIST SP 800-53 Rev. 4** AU-12, CA-7, CM-3, CM-8, PE-3, PE-6, PE-20, SI-4	VERY IMPORTANT
		DE.CM-8: Vulnerability scans are performed	· **CIS CSC** 4, 20 · **COBIT 5** BAI03.10, DSS05.01 · **ISA 62443-2-1:2009** 4.2.3.1, 4.2.3.7 · **ISO/IEC 27001:2013** A.12.6.1 · **NIST SP 800-53 Rev. 4** RA-5	VERY IMPORTANT
	Detection Processes (DE.DP): Detection processes and procedures are maintained and tested to ensure awareness of anomalous events.	**DE.DP-1:** Roles and responsibilities for detection are well defined to ensure accountability	· **CIS CSC** 19 · **COBIT 5** APO01.02, DSS05.01, DSS06.03 · **ISA 62443-2-1:2009** 4.4.3.1 · **ISO/IEC 27001:2013** A.6.1.1, A.7.2.2 · **NIST SP 800-53 Rev. 4** CA-2, CA-7, PM-14	IMPORTANT
		DE.DP-2: Detection activities comply with all applicable requirements	· **COBIT 5** DSS06.01, MEA03.03, MEA03.04 · **ISA 62443-2-1:2009** 4.4.3.2 · **ISO/IEC 27001:2013** A.18.1.4, A.18.2.2, A.18.2.3 · **NIST SP 800-53 Rev. 4** AC-25, CA-2, CA-7, SA-18, SI-4, PM-14	IMPORTANT
		DE.DP-3: Detection processes are tested	· **COBIT 5** APO13.02, DSS05.02 · **ISA 62443-2-1:2009** 4.4.3.2 · **ISA 62443-3-3:2013** SR 3.3 · **ISO/IEC 27001:2013** A.14.2.8 · **NIST SP 800-53 Rev. 4** CA-2, CA-7, PE-3, SI-3, SI-4, PM-14	VERY IMPORTANT
		DE.DP-4: Event detection information is communicated	· **CIS CSC** 19 · **COBIT 5** APO08.04, APO12.06, DSS02.05 · **ISA 62443-2-1:2009** 4.3.4.5.9 · **ISA 62443-3-3:2013** SR 6.1 · **ISO/IEC 27001:2013** A.16.1.2, A.16.1.3 · **NIST SP 800-53 Rev. 4** AU-6, CA-2, CA-7, RA-5, SI-4	VERY IMPORTANT
		DE.DP-5: Detection processes are continuously improved	· **COBIT 5** APO11.06, APO12.06, DSS04.05 · **ISA 62443-2-1:2009** 4.4.3.4 · **ISO/IEC 27001:2013** A.16.1.6 · **NIST SP 800-53 Rev. 4**, CA-2, CA-7, PL-2, RA-5, SI-4, PM-14	VERY IMPORTANT
	Response Planning (RS.RP): Response processes and procedures are executed and maintained, to ensure response to detected cybersecurity incidents.	**RS.RP-1:** Response plan is executed during or after an incident	· **CIS CSC** 19 · **COBIT 5** APO12.06, BAI01.10 · **ISA 62443-2-1:2009** 4.3.4.5.1 · **ISO/IEC 27001:2013** A.16.1.5 · **NIST SP 800-53 Rev. 4** CP-2, CP-10, IR-4, IR-8	VERY IMPORTANT
		RS.CO-1: Personnel know their roles and order of operations when a response is needed	· **CIS CSC** 19 · **COBIT 5** EDM03.02, APO01.02, APO12.03 · **ISA 62443-2-1:2009** 4.3.4.5.2, 4.3.4.5.3, 4.3.4.5.4 · **ISO/IEC 27001:2013** A.6.1.1, A.7.2.2, A.16.1.1 · **NIST SP 800-53 Rev. 4** CP-2, CP-3, IR-3, IR-8	IMPORTANT
		RS.CO-2: Incidents are reported consistent with established criteria	· **CIS CSC** 19 · **COBIT 5** DSS01.03 · **ISA 62443-2-1:2009** 4.3.4.5.5 · **ISO/IEC 27001:2013** A.6.1.3, A.16.1.2	VERY IMPORTANT

	Communications (RS.CO): Response activities are coordinated with internal and external stakeholders (e.g. external support from law enforcement agencies).		· **NIST SP 800-53 Rev. 4** AU-6, IR-6, IR-8	
		RS.CO-3: Information is shared consistent with response plans	· **CIS CSC** 19 · **COBIT 5** DSS03.04 · **ISA 62443-2-1:2009** 4.3.4.5.2 · **ISO/IEC 27001:2013** A.16.1.2, Clause 7.4, Clause 16.1.2 · **NIST SP 800-53 Rev. 4** CA-2, CA-7, CP-2, IR-4, IR-8, PE-6, RA-5, SI-4	IMPORTANT
		RS.CO-4: Coordination with stakeholders occurs consistent with response plans	· **CIS CSC** 19 · **COBIT 5** DSS03.04 · **ISA 62443-2-1:2009** 4.3.4.5.5 · **ISO/IEC 27001:2013** Clause 7.4 · **NIST SP 800-53 Rev. 4** CP-2, IR-4, IR-8	IMPORTANT
		RS.CO-5: Voluntary information sharing occurs with external stakeholders to achieve broader cybersecurity situational awareness	· **CIS CSC** 19 · **COBIT 5** BAI08.04 · **ISO/IEC 27001:2013** A.6.1.4 · **NIST SP 800-53 Rev. 4** SI-5, PM-15	MODERATE
RESPOND (RS)	**Analysis (RS.AN):** Analysis is conducted to ensure effective response and support recovery activities.	**RS.AN-1:** Notifications from detection systems are investigated	· **CIS CSC** 4, 6, 8, 19 · **COBIT 5** DSS02.04, DSS02.07 · **ISA 62443-2-1:2009** 4.3.4.5.6, 4.3.4.5.7, 4.3.4.5.8 · **ISA 62443-3-3:2013** SR 6.1 · **ISO/IEC 27001:2013** A.12.4.1, A.12.4.3, A.16.1.5 · **NIST SP 800-53 Rev. 4** AU-6, CA-7, IR-4, IR-5, PE-6, SI-4	VERY IMPORTANT
		RS.AN-2: The impact of the incident is understood	· **COBIT 5** DSS02.02 · **ISA 62443-2-1:2009** 4.3.4.5.6, 4.3.4.5.7, 4.3.4.5.8 · **ISO/IEC 27001:2013** A.16.1.4, A.16.1.6 · **NIST SP 800-53 Rev. 4** CP-2, IR-4	IMPORTANT
		RS.AN-3: Forensics are performed	· **COBIT 5** APO12.06, DSS03.02, DSS05.07 · **ISA 62443-3-3:2013** SR 2.8, SR 2.9, SR 2.10, SR 2.11, SR 2.12, SR 3.9, SR 6.1 · **ISO/IEC 27001:2013** A.16.1.7 · **NIST SP 800-53 Rev. 4** AU-7, IR-4	VERY IMPORTANT
		RS.AN-4: Incidents are categorized consistent with response plans	· **CIS CSC** 19 · **COBIT 5** DSS02.02 · **ISA 62443-2-1:2009** 4.3.4.5.6 · **ISO/IEC 27001:2013** A.16.1.4 · **NIST SP 800-53 Rev. 4** CP-2, IR-4, IR-5, IR-8	IMPORTANT
		RS.AN-5: Processes are established to receive, analyze and respond to vulnerabilities disclosed to the organization from internal and external sources (e.g. internal testing, security bulletins, or security researchers)	· **CIS CSC** 4, 19 · **COBIT 5** EDM03.02, DSS05.07 · **NIST SP 800-53 Rev. 4** SI-5, PM-15	IMPORTANT
	Mitigation (RS.MI): Activities are performed to prevent expansion of an event, mitigate its effects, and resolve the incident.	**RS.MI-1:** Incidents are contained	· **CIS CSC** 19 · **COBIT 5** APO12.06 · **ISA 62443-2-1:2009** 4.3.4.5.6 · **ISA 62443-3-3:2013** SR 5.1, SR 5.2, SR 5.4 · **ISO/IEC 27001:2013** A.12.2.1, A.16.1.5 · **NIST SP 800-53 Rev. 4** IR-4	VERY IMPORTANT
		RS.MI-2: Incidents are mitigated	· **CIS CSC** 4, 19 · **COBIT 5** APO12.06 · **ISA 62443-2-1:2009** 4.3.4.5.6, 4.3.4.5.10 · **ISO/IEC 27001:2013** A.12.2.1, A.16.1.5 · **NIST SP 800-53 Rev. 4** IR-4	VERY IMPORTANT
		RS.MI-3: Newly identified vulnerabilities are mitigated or documented as accepted risks	· **CIS CSC** 4 · **COBIT 5** APO12.06 · **ISO/IEC 27001:2013** A.12.6.1 · **NIST SP 800-53 Rev. 4** CA-7, RA-3, RA-5	VERY IMPORTANT
	Improvements (RS.IM): Organizational response activities are improved by incorporating lessons learned from current and previous	**RS.IM-1:** Response plans incorporate lessons learned	· **COBIT 5** BAI01.13 · **ISA 62443-2-1:2009** 4.3.4.5.10, 4.4.3.4 · **ISO/IEC 27001:2013** A.16.1.6, Clause 10 · **NIST SP 800-53 Rev. 4** CP-2, IR-4, IR-8	VERY IMPORTANT

	detection/response activities.	**RS.IM-2:** Response strategies are updated	· **COBIT 5** BAI01.13, DSS04.08 · **ISO/IEC 27001:2013** A.16.1.6, Clause 10 · **NIST SP 800-53 Rev. 4** CP-2, IR-4, IR-8	VERY IMPORTANT
RECOVER (RC)	**Recovery Planning (RC.RP):** Recovery processes and procedures are executed and maintained to ensure restoration of systems or assets affected by cybersecurity incidents.	**RC.RP-1:** Recovery plan is executed during or after a cybersecurity incident	· **CIS CSC** 10 · **COBIT 5** APO12.06, DSS02.05, DSS03.04 · **ISO/IEC 27001:2013** A.16.1.5 · **NIST SP 800-53 Rev. 4** CP-10, IR-4, IR-8	VERY IMPORTANT
	Improvements (RC.IM): Recovery planning and processes are improved by incorporating lessons learned into future activities.	**RC.IM-1:** Recovery plans incorporate lessons learned	· **COBIT 5** APO12.06, BAI05.07, DSS04.08 · **ISA 62443-2-1:2009** 4.4.3.4 · **ISO/IEC 27001:2013** A.16.1.6, Clause 10 · **NIST SP 800-53 Rev. 4** CP-2, IR-4, IR-8	VERY IMPORTANT
		RC.IM-2: Recovery strategies are updated	· **COBIT 5** APO12.06, BAI07.08 · **ISO/IEC 27001:2013** A.16.1.6, Clause 10 · **NIST SP 800-53 Rev. 4** CP-2, IR-4, IR-8	VERY IMPORTANT
	Communications (RC.CO): Restoration activities are coordinated with internal and external parties (e.g. coordinating centers, Internet Service Providers, owners of attacking systems, victims, other CSIRTs, and vendors).	**RC.CO-1:** Public relations are managed	· **COBIT 5** EDM03.02 · **ISO/IEC 27001:2013** A.6.1.4, Clause 7.4	IMPORTANT
		RC.CO-2: Reputation is repaired after an incident	· **COBIT 5** MEA03.02 · **ISO/IEC 27001:2013** Clause 7.4	IMPORTANT
		RC.CO-3: Recovery activities are communicated to internal and external stakeholders as well as executive and management teams	· **COBIT 5** APO12.06 · **ISO/IEC 27001:2013** Clause 7.4 · **NIST SP 800-53 Rev. 4** CP-2, IR-4	IMPORTANT

ANNEX VII. NODE-RED DASHBOARD PROJECT

Subscription Management Flow

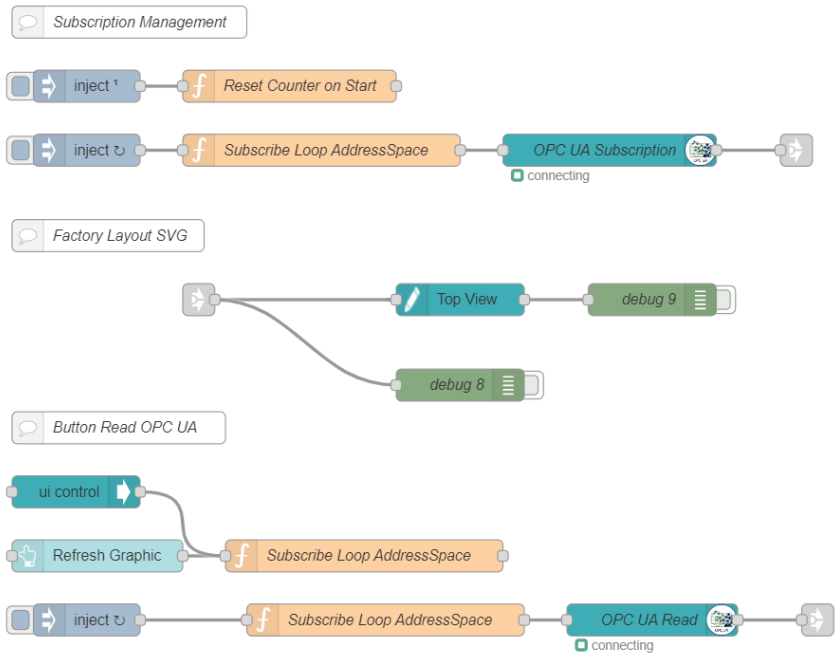
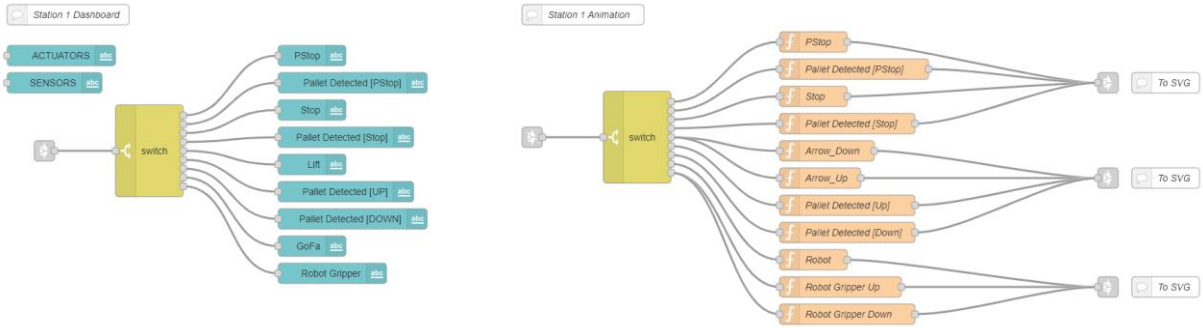


Figure 102. Node-RED dashboard: Subscription management flow (Original work).

Stations Flow



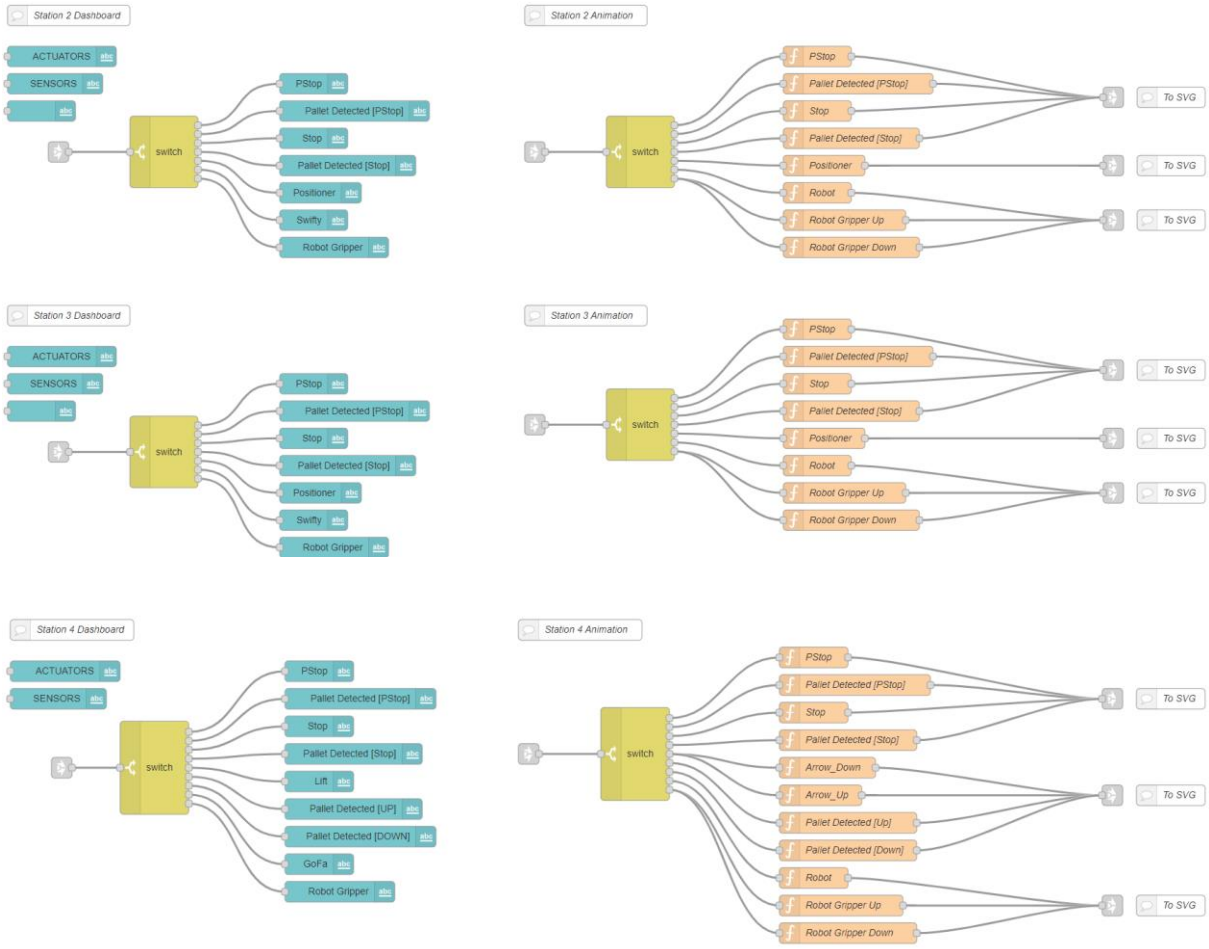


Figure 103. Node-RED dashboard: Stations flow (Original work).

Conveyor Belts Flow

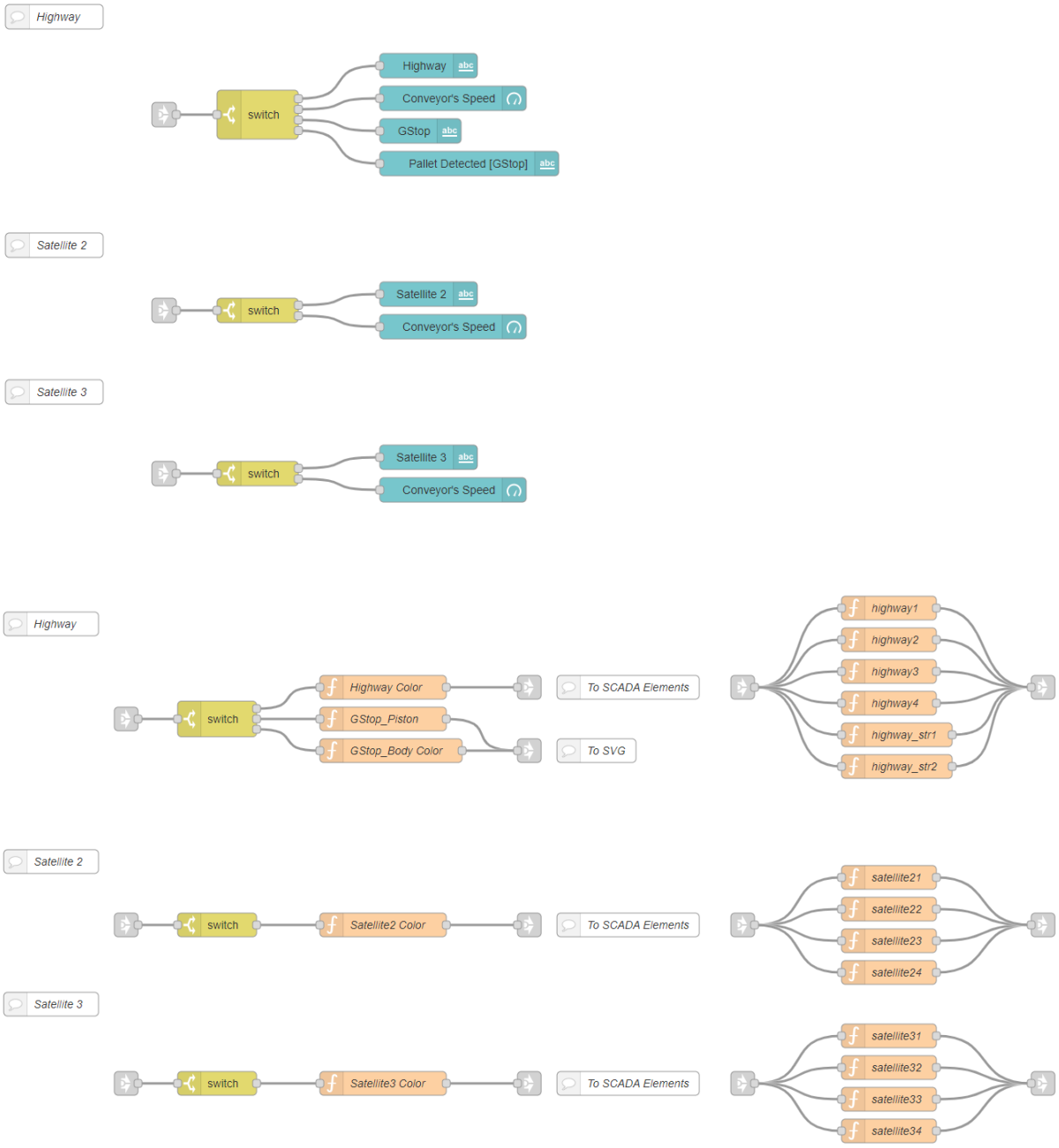


Figure 104. Node-RED dashboard: Conveyor belts flow (Original work).

Diverts & Combines Flow

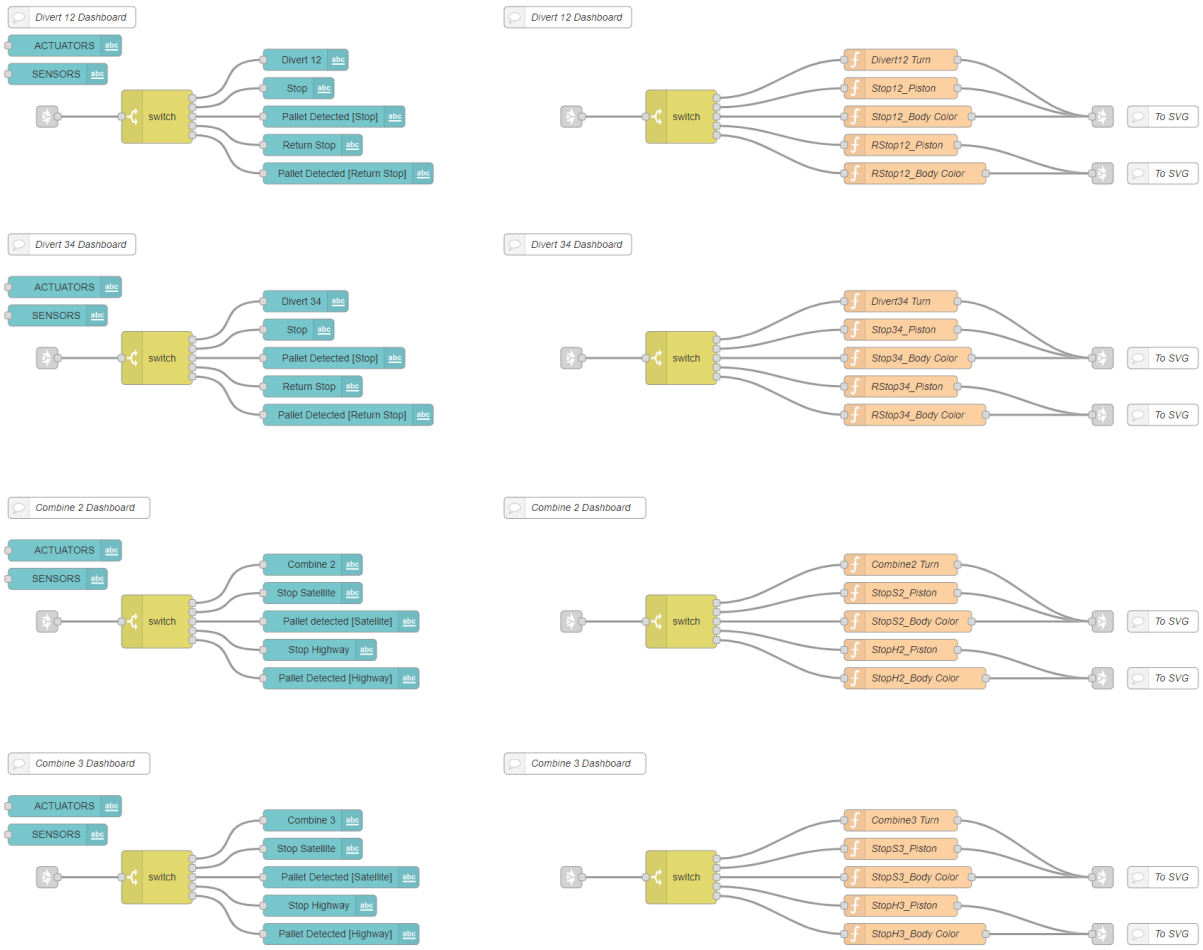


Figure 105. Node-RED dashboard: Diverts and combines flow (Original work).

Assets Flow

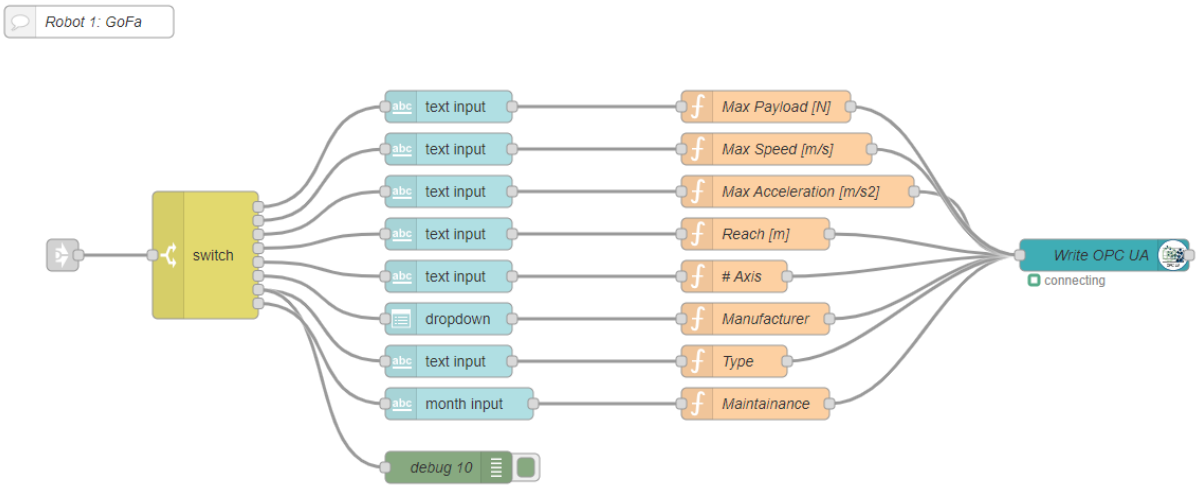


Figure 106. Node-RED dashboard: Assets flow (Original work).

M+O System Flow

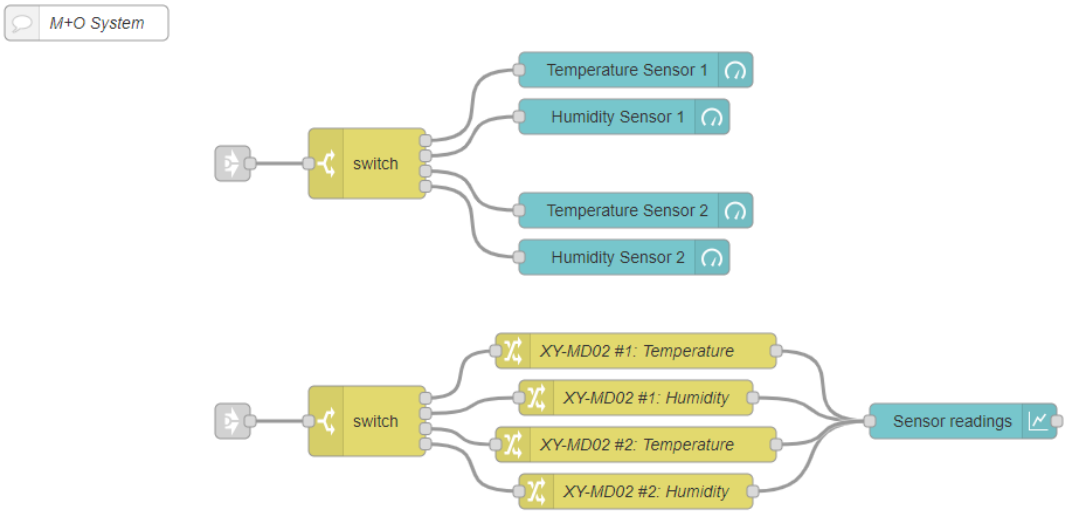


Figure 107. Node-RED dashboard: M+O System flow (Original work).

ANNEX VIII. SYSTEM'S COMMISSIONING

1. Connect all peripheral devices to the IOT2050:

- ✓ Connect the gateway to the factory's LAN network using an Ethernet cable. Use the connector installed in the box, which is connected to the Ethernet port with a static IP address (P1). The gateway static IP address is: 192.168.56.170.
- ✓ Connect the USB to RS-485 converter to the flexible USB connector and attach the sensors. Be careful to connect Line A and Line B to the correct pins on the RS-485 converter. To power up the sensors, connect them to a power supply of 12V.
- ✓ Connect the RS-232 converter to the other flexible USB connector and to the data diode following the diagram on the diode's case. Power up the diode using the same 24V power supply used by the IoT gateway.

NOTE: It is important to connect each serial converter to its respective USB connector. If this is not the case, the node-RED project will not work. This can be solved by modifying the Node-RED program accordingly or by simply swapping connections.

2. Bootup the IOT gateway:

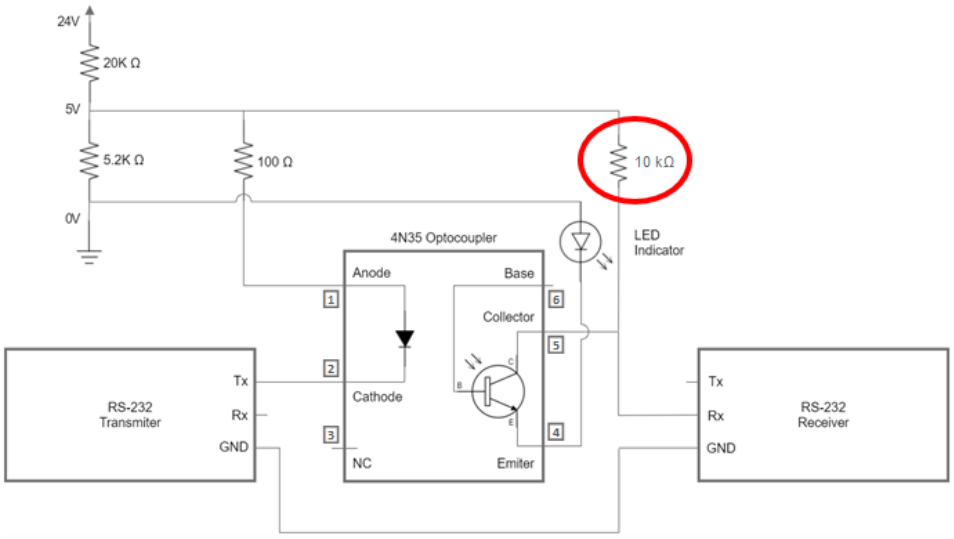
- ✓ Connect the power cable to the electrical box and turn on the device using the power switch. Wait a few minutes for the device to fully start.
- ✓ Access the Node-RED workspace from any computer in the lab by entering the following URL: http://192.168.56.170:1880.
- ✓ Verify that the Node-RED flows are working correctly. There are three flows in total: OPC UA client subscription, M+O Modbus master, and RS-232 transmitter.
- ✓ **Optional:** Use PuTTY to establish an SSH connection (username: root; password: edgeicai).

NOTE: The RS-232 transmitter has two versions, one with encryption and another without. The program that uses encryption does not work properly and should be modified due to limitations in the RS-232 transmitter node.

3. Start the lab computer that hosts the rest of the NOA components:

- ✓ Run the server application and create two user accounts: one for the OPC UA client that writes data to the server (username: diode; password: diode), and one for the OPC UA client used in the dashboard application (username: dash; password: dash). These usernames and passwords can be changed, but they also need to be updated in the respective Node-RED projects.
- ✓ Connect the USB to RS-232 converter, used to receive information from the diode, to the lab computer. Check the "device administrator" to determine the COM port used.
- ✓ Start the Node-RED project that receives data from the diode and writes it to the server. The command to start this project is: `node-red projects/rs232_Receiver`. Once started, access it from the browser using the following URL: http://127.0.0.1:1880. If there are any issues, verify if the COM port in the project corresponds to that shown in the "device administrator".
- ✓ Run the Node-RED project of the SCADA dashboard. This project must be started on a different port than that from the previous one: `node-red projects/dashboard -p 1881`. Access it through the browser using the URL: http://127.0.0.1:1881. The dashboard can be opened from: http://127.0.0.1:1881/ui.

DATA DIODE NOTES: The data diode has a known issue because the resistor, indicated in the provided schematic, is missing.



As a result, it may not function properly with all USB to RS-232 converters connected in the receiving end of the diode. The only converter found to work reliably, even with the missing resistor, is the Serial Port CH340C.

Type C to TTL Serial Port CH340C Module CH340 USB Bus Conversion Chip ISP Communicate Connector for STM32 Serial Port Download

★★★★★ 5.0 6 Reviews 31 Sold

0,97€ ~~1,22€~~ 20% off

Another issue to consider is related to the USB to RS-232 converter used for transmissions, as only one model had its drivers pre-installed in the IOT2050 Gateway. Although this specific device is a USB to RS-485 converter, users can remove the four coloured jumpers and access the RS-232 pins that allow it to connect to the diode.