
*Semester Project Report - SS23*

---

# Development and implementation of Industry 4.0 Workpiece Transfer Unit


| | |
|---|---|
| Author: | Peeranut Noonurak |
| Matriculation No.: | 7023582 |
| Course of Studies: | M.Eng. in Industrial Informatics |
| First examiner: | Prof. Dr.-Ing. Armando Walter Colombo |
| Second examiner: | Martin Bär, M.Eng. |
| Third examiner: | Jeffrey Werman, M.Eng. |
| Submission date: | 7.11.2023 |

University of Applied Sciences Emden/Leer · Faculty of Technology ·
Department of Electrical Engineering and Computer Science
Master Degree in Industrial Informatics
Constantiaplatz 4 · 26723 Emden · http://www.hs-emden-leer.de

# Declaration of authorship

I hereby declare that I, the undersigned, am the sole author of this document, All sources consulted for this document have been listed: all quotations from and references to these sources have been properly cited and included in chapter notes and in the list of references. No version of this document, either in whole or any section of it, has been used to achieve an academic degree or any other examination. I understand that any false statements made in this declaration may be punishable by law.

07.11.2023

Date Signature

# Abstracts

The Industry 4.0 workpiece transfer unit project has realized several significant achievements in digital manufacturing. Through meticulous design and implementation, the project successfully integrated the unit into the digital factory landscape, adhering to the principles of Industry 4.0. Implementing the Asset Administration Shell (AAS) framework allowed seamless data exchange with other modules, enhancing production control. However, it is important to understand that this achievement is closely tied to the fact that the digital factory is always changing and still developing. New machines and technologies may regularly be added, which means that the workpiece transfer unit must be able to adjust and keep working smoothly. This ability to adapt to these ongoing changes is a fundamental part of Industry 4.0, emphasizing that being adaptable is not something that can be achieved once and then forgotten about; it is an ongoing process of evolution.

In anticipation of the future, several crucial tasks must be undertaken. As the digital factory continues its advancement, there is a necessity to enhance and increase the efficiency of the workpiece transfer unit. This necessitates a continuous process of updating and maintaining close collaboration with any other module of the digital factory. The objective is to ensure seamless integration with all other machinery and systems within the factory, thereby facilitating smoother production processes. Additionally, efforts should be directed towards identifying solutions for unsteady issues, such as the 'BadCommunicationError - Error in watchdog loop', which occasionally arises when the unit operates independently. This will ensure the unit's continued optimal performance, even in the face of unexpected occurrences.

Furthermore, the project has plans for future improvements in the unit's capabilities. One idea is to directly supply power from the UR5e robot controller's source, heightening reliability, especially when power interruptions are a concern. Moreover, the report underscores the potential for advanced file attachment management within the OPC UA server, which simplifies file handling, heightens security, and advances data integrity. These improvements align with the key principles of Industry 4.0, which focus on connectivity, data-driven decision-making, and continuously improving operations in the rapidly changing world of smart manufacturing. In summary, the Industry 4.0 workpiece transfer unit project is a significant achievement in digital manufacturing, with plans for future improvements and innovations to ensure its ongoing relevance and effectiveness in the ever-evolving digital factory environment.

Additionally, the most recent version of this document as well as resources that are essential to this project, such as program code and CAD files, can also be found on GitHub.

# Contents

# List of Figures

# List of Tables

# Acronyms

**AAS** Asset Administration Shell

**AASX** Asset Administration Shell Package Explorer

**C-Class** Communication capabilities class

**CD** Concept Description

**CP Notation** Communication and Presentation notation

**DICPS** Digitalization of Industrial Cyber-Physical Systems

**I40** Industry 4.0

**ICPS** Industrial Cyber-Physical Systems

**IDTA** Industrial Digital Twin Association

**IDTT** Industrial Data Transport Technologie

**IIC** Industrial Internet Consortium

**IIRA** Industrial Internet Reference Architecture

**LAN** Local Area Network

**OPC UA** Open Platform Communications Unified Architecture

**Opr** Submodel Element Operation

**P-Class** Presentation in information system class

**Prop** Submodel Element Property

**SoA** Service-oriented Architecture

**UR5e** Universal Robots version 5 of e-series

# 1. Introduction

## 1.1. Introduction and Motivation

The Digitalization of Industrial Cyber-Physical Systems (DICPS) is a transformative process that empowers organizations to leverage digital technologies for enhanced efficiency, productivity, and innovation within the Industrie 4.0 landscape. As part of the "Digitalization of ICPS" and "Industrial Cyber-Physical Systems" courses, students gain a comprehensive understanding of the underlying principles, technologies, and methodologies involved in this paradigm shift [Col01].

The Industrial Cyber-Physical Systems (ICPS) course specifically focuses on the latest advancements in industrial systems, honing in on Industry 4.0 solutions. This module equips students with a deep comprehension of Industry 4.0 frameworks, specifically emphasizing the digitalization and networking of systems using ICPS technologies. The course covers diverse facets related to ICPS, including understanding the outcomes of the 3rd Industrial Revolution, learning technologies and architectural patterns for implementing industrial cyber-physical systems, exposure to enterprise standard architectures, understanding the Life Cycle and Value Stream, and exploring standards and technologies for specifying and implementing different layers of the RAMI 4.0 vertical dimension [Col23].

Furthermore, the Industrial Data Transport Technologie (IDTT) course provided insights into implementing Industrial Cyber-Physical Systems (ICPS) by harnessing technological potential and fostering systematic innovation. Key aspects covered were horizontal and end-to-end digital integration and vertical integration in ICPS. Throughout the course, a focus was placed on crafting adaptable ICPS architectures utilizing widely used data transport technologies such as XML and OPC/OPC-UA. This facilitated the development of effective strategies for efficiently exchanging data within real-world industrial applications [Ber01].

This project, titled "Development and Implementation of Industry 4.0 Workpiece Transfer Unit," is a direct response to the transformative power of digitalization in industrial cyber-physical systems. As the Industrie 4.0 landscape continues to evolve, the role of digital twins, Asset Administration Shell (AAS) technology, and Service-oriented Architecture (SoA) become increasingly prominent. Building on the foundational knowledge gained from the "Digitalization of ICPS," "Industrial Cyber-Physical Systems," and "Industrial Data Transport Technologies" courses, this project focuses on the comprehensive digitalization of the Workpiece Transfer Unit within a digital factory environment. The Industry 4.0 Workpiece Transfer Unit encompasses diverse hardware components, including the Gripper, Base Tool, Robot Stand, and Universal Robot UR5e, all of which are interconnected through carefully designed software components, including the URP robot program and OPC UA server/client, to ensure seamless subsystem integration and communication.

## 1.2. Previous Summer Semester Project

In a previous summer semester project, a comprehensive workpiece transfer unit was designed, developed, and validated for integration into a digital manufacturing environment. The validation process involved testing the functionality and performance of subsystems, ensuring compliance with safety requirements and project specifications.

Although certain validations were deferred due to the ongoing construction of the digital factory, the workpiece transfer unit was designed with digitalization solutions in mind. The project showcased the unit's potential as a valuable component in advancing digital manufacturing by demonstrating pick-and-place operations in a simulated environment. The project laid the groundwork for future enhancements and refinements [NM23].

## 1.3. Aim and Structure of the Work

This extended project aims to digitalize the entire Workpiece Transfer Unit, encompassing the Gripper, Base Tool, Robot Stand, and Universal Robot UR5e. Doing so will achieve a comprehensive digital representation of the unit's functionalities, capabilities, and data.

The structure of this work comprises the following chapters:

1. **Introduction:** This chapter introduces the project's objectives, motivation, and background. It also provides an overview of the previous summer semester project and its contributions to the current endeavour.

2. **Background:** This chapter provides an overview of key concepts and frameworks relevant to the digitalization of the Workpiece Transfer Unit using the Asset Administration Shell (AAS). It delves into existing research and developments related to digitalization in industrial environments. It sets the foundation for the project's implementation approach, including Digital Factory in Technikum, CP Notation, RAMI 4.0, Asset Administration Shell (AAS), and Revolution Pi.

3. **Project Management:** This chapter provides a detailed insight into the project plan, timeline, and allocated resources for digitalizing the Workpiece Transfer Unit using the Asset Administration Shell (AAS). It explains the project's organizational structure and management approach, ensuring efficient execution and coordination.

4. **Detailed Design:** This chapter elaborates on the design decisions, technical considerations, and chosen architectural framework for integrating the Asset Administration Shell (AAS) into the Workpiece Transfer Unit. The chapter outlines the blueprint for AAS implementation, ensuring a comprehensive understanding of the solution's design principles.

5. **Implementation:** Describing the practical steps taken to transform the entire Workpiece Transfer Unit digitally, this chapter details the implementation of AAS for the Workpiece Transfer Unit, configuration of the Universal Robot UR5e programming, and deployment of OPC UA Server AASX on RevPi.

6. **Test & Validation:** This chapter discusses the validation procedure to verify the implemented Asset Administration Shell (AAS) functionality, performance, and interoperability within the digital factory environment. It also involves validating requirements to ensure the Workpiece Transfer Unit meets its criteria. The chapter highlights the testing methods employed, the outcomes achieved, and how they align with the project's requirements.

7. **Conclusion and Outlook:** Summarizing the project's achievements, emphasizing its contributions to digitalization in manufacturing, and outlining potential avenues for future enhancements and refinements.

Through the digitalization of the Workpiece Transfer Unit in the digital factory, this project aims to propel the digital transformation endeavors in the manufacturing environment. By building upon the achievements of the previous summer semester project, the current endeavor aspires to extend the boundaries of digitalization and unleash untapped potentials for elevating production management and processes within the expansive realm of Industrie 4.0.

## 1.4. Project's Key Requirements

This section outlines the pivotal requirements driving the development of the Industry 4.0 workpiece transfer unit. These requirements have been categorized into four key aspects: Functional Requirements, Non-Functional Requirements, Connectivity Requirements, and Documentation Requirements. Successful fulfilment of these requirements is fundamental to the overall success of this project.

**Functional Requirements** focus on the essential capabilities and functionalities of the unit, including:

- Seamless Module Integration and Quick Adaptation.
- Improved Interconnectivity and Data Exchange.
- Streamlined Production Management and Decision-making.
- Operate in All Six Directions.
- Standalone Operation.
- Virtual Representation.
- 3D Equipment Model Maintenance.

**Non-Functional Requirements** emphasize qualities critical to the unit's performance, including:

- CP Notation Positioning.
- RAMI 4.0 Conformance.

**Connectivity Requirements** pertain to the unit's communication protocols and interfaces, ensuring compatibility and interoperability with Industry 4.0 standards.

**Documentation Requirement** relates to the comprehensive project background that the report must provide, conveying essential concepts related to unit development and its context within the broader Industry 4.0 framework.

While this section offers a brief overview, subsequent chapters will delve into each requirement in detail, particularly in the project management section. This comprehensive exploration will provide a clear roadmap for the project's development, ensuring alignment with Industry 4.0 objectives.

# 2. Background

This chapter introduces key project components: Digital Factory, CP Notation, Reference Architecture Model Industrie 4.0, Industrie 4.0 Components, Digital Twin, Asset Administration Shell, and RevPi Core S. Understanding these is crucial for grasping the project's essence, as they underpin its core and applications. The chapter provides an overview, highlighting each component's significance and role. This exploration offers insights into project pillars, connections, and objectives.

## 2.1. Digital Factory in Technikum

The digital factory project, located within the Technikum at the Hochschule Emden/Leer University of Applied sciences, is an ongoing initiative that demonstrates technologies and facilitates the investigation and development of advanced industrial concepts. The digital factory seeks to create a digital manufacturing environment that embodies the principles of Industry 4.0 (I40). It is a dedicated space for students and researchers to acquire practical experience and knowledge of industrial processes [HS 22].

The core concept of the digital factory is the provision of various modules, each designed to perform specific manufacturing duties. These modules encompass various functionalities, including a laser cutting module, five other modules replicating various industrial machining and servicing processes, and a central workpiece transport module. Integrating these modules within the digital factory initiative demonstrates advanced manufacturing technologies' diverse capabilities and applications. The fundamental factory layout is depicted in the Figure 2.1.

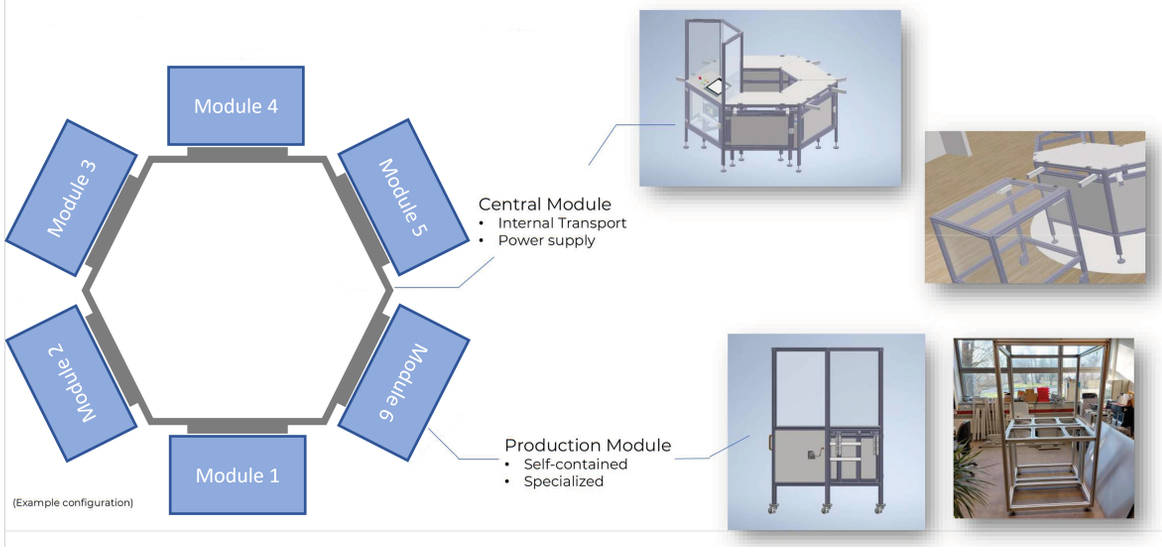



Figure 2.1.: Concept of digital factory at HS Emden Leer, based on [HS 22]

The design of the digital factory prioritises modularity, enabling the seamless integration and reconfiguration of modules. This modularity facilitates flexibility and adaptability, enabling the digital factory to respond effectively to changing production requirements. Physical and digital connections allow for the exchange or replacement of modules, facilitating a streamlined and interconnected manufacturing ecosystem.

## 2.2. CP Notation

Classification of Presentation and Communication (CP) notation is a fundamental concept within the DIN SPEC 91345:2016-04 framework. Within an information system, it facilitates the categorization of assets based on their presentation and communication capabilities. This classification scheme specifies how information system objects are administered and presented, regardless of their communication capabilities. [PAB+16].

In this context, "object" refers to any digitally represented asset within the system. The CP Notation provides a method for classifying assets based on how they are presented in the information system and their communication capabilities, with these two characteristics being independent. The purpose of the CP notation is to facilitate the management of objects within an information system. Passive communication capabilities (such as signal interfaces or RFID) can improve this administration, but they are not required.

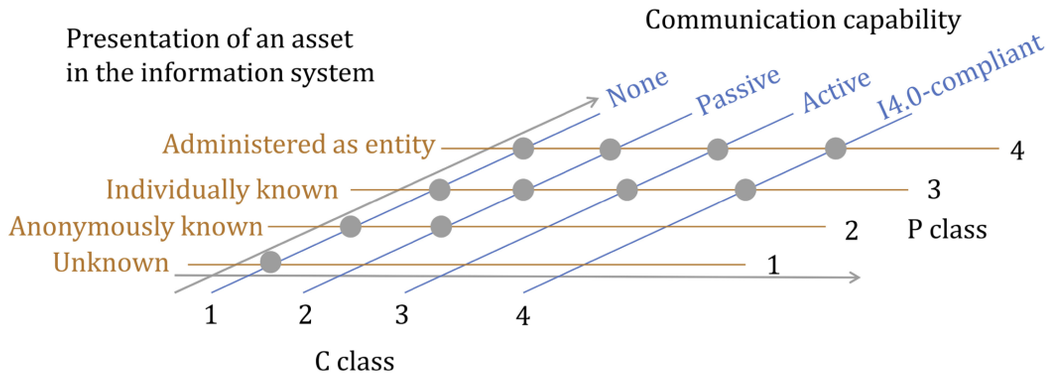



Figure 2.2.: Asset Presentation and Communication Capability, based on [PAB+16]

Figure 2.2 provides an illustrative representation of the classification system. The administration class of each object, essentially its presentation within the information system, can be chosen per design requirements.

As a result, the CP Notation classification system employs the "CP" acronym to represent "Communication" and "Presentation," followed by a number that represents an object's classification. This system is comparable to extensively employed notation systems such as IP protection classes. As shown in Figure 2.3, selecting the CP class for an asset depends on factors such as communication capabilities and presentation needs within the information system.

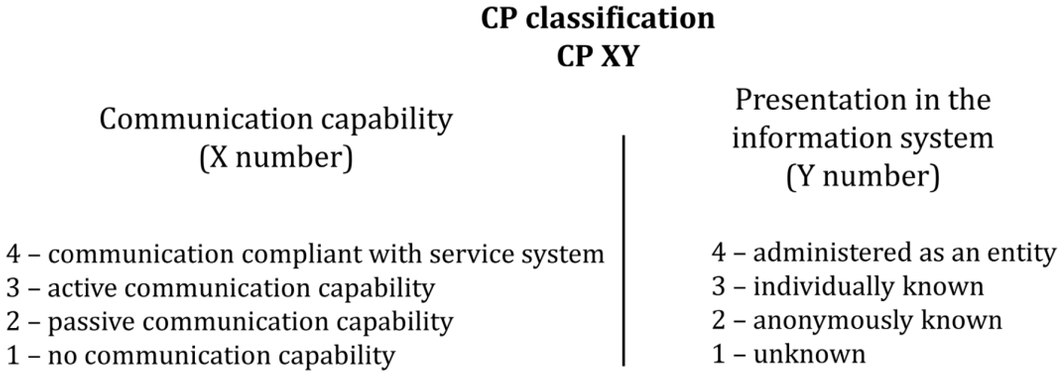



Figure 2.3.: CP Notation System for Communication and Presentation Classification, based on [PAB+16]

For instance, an individually identifiable asset capable of active communication could be denoted as CP33, which represents a traditional field device with a field bus connection. On the other hand, an asset like a security container, which is monitored throughout its lifecycle but lacks communication capability, can be classified as CP14.

In Industrie 4.0, an asset is considered a CP24, CP34, or CP44 component based on the classification system illustrated in Figure 2.3. The CP Notation system ensures that assets are organized coherently, allowing for efficient administration, presentation, and communication within an information system.

In summary, CP Notation is a crucial component of the industrial cyber-physical systems (ICPS). It provides a structured approach to categorizing and administering assets based on communication and presentation capabilities within the digital ecosystem. This classification system enhances the interoperability, modularity, and standardized representation of assets in an Industrie 4.0 context.

## 2.3. Reference Architecture Model Industrie 4.0 (RAMI 4.0)

The Reference Architecture Model Industrie 4.0 (RAMI 4.0) is a structured representation of the fundamental elements of an asset within a three-axis level model, as illustrated in Figure 2.4. This model provides a method for decomposing complex interdependencies into more manageable components by combining all three dimensions at each stage of an asset's lifecycle to encompass all relevant aspects [Sch16].

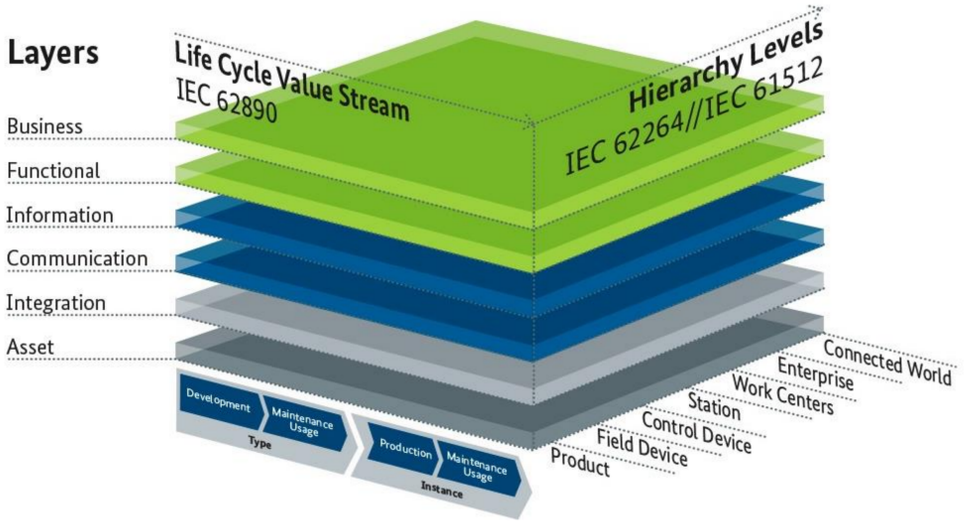



Figure 2.4.: RAMI 4.0 , based on [Sch16]

The three axes comprising RAMI 4.0 are:

- **The Architecture axis ("Layers"):** This axis consists of six layers that portray the information pertinent to an asset's role.
- **The "Life Cycle & Value Stream" axis:** This axis captures an asset's life cycle and its value-added process, encompassing its production, utilization, and eventual disposal.
- **The "Hierarchy Levels" axis:** This axis assigns functional models to specific levels, ensuring a coherent representation across industries.

The primary objective of RAMI 4.0 is to describe assets and asset combinations precisely. It is important to note that RAMI 4.0's description is logical, and actual implementations may vary. For instance, a machine's MES function may be logically described in one layer but implemented in another layer, showcasing the model's flexibility.

### 2.3.1. Architecture Axis ("Layers")

The vertical architecture axis utilizes six layers to characterize an asset's characteristics, system structures, functions, and function-specific data. These layers define the structural characteristics of an asset or asset combination and contribute to the logical representation of assets and their relationships: Business, Functional, Information, Communication, Integration, and Asset.

#### Business Layer

The "Business" layer captures the commercial perspective of an asset, encompassing organizational conditions, monetary aspects, function integrity, business models, legal conditions, and more. It orchestrates services on the "Functional" layer and connects different business processes.

#### Functional Layer

The "Functional" layer defines the logical functions of an asset concerning its role within the Industrie 4.0 system. This layer handles digital function descriptions, horizontal integration, runtime environments for services and business processes, and for applications and technical functionality.

#### Information Layer

The "Information" layer centers on the data used, generated, or altered by an asset's technical functionality. It involves runtime environments for event processing, rule execution, formal model and rule descriptions, data persistence, integration of data, acquiring new data, and more.

#### Communication Layer

The "Communication" layer outlines Industrie 4.0-compliant access to asset information and functions by other assets. It defines how data is used, distributed, and accessed within the system.

#### Integration Layer

The "Integration" layer bridges the gap between the physical and information worlds. It encompasses the infrastructure for function implementation, technical process control, event generation, and human-machine interfaces.

#### Asset Layer

The "Asset" layer represents the physical existence of assets. It serves as the interface between humans and the information world, and the connection of the assets to the “Integration” layer.

### 2.3.2. Life Cycle & Value Stream Axis

The "Life Cycle & Value Stream" axis delves into an asset's depiction at specific times during its lifespan, from production to disposal.

### 2.3.3. Hierarchy Axis

The "Hierarchy" axis aligns with factory reference architecture models, incorporating levels like "enterprise," "work centers," and "station." Industrie 4.0-specific terms like "connected world," "field device," and "product" have been integrated to address various industries' requirements.

This comprehensive model offers a structured framework for asset description, enhancing interoperability, modularity, and standardized representation within the Industrie 4.0 landscape.

## 2.4. Industrie 4.0 Components

This section covers Industrie 4.0 components and their primary functions and importance within the framework.

### 2.4.1. Overview

Industrie 4.0 components, denoted as I4.0 components, are pivotal in modern industrial systems. They are identifiable participants with the ability to communicate seamlessly within an I4.0 framework. These components consist of an administration shell and a corresponding asset, interconnected digitally within an I4.0 system. This connection adheres to specific classifications, such as CP24, CP34, or CP44, and offers services with defined quality of service (QoS) properties.

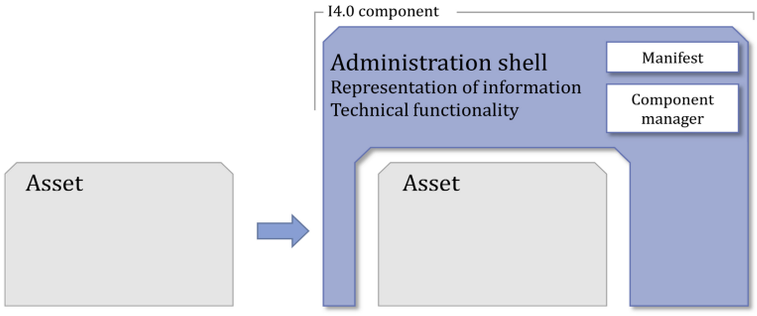



Figure 2.5.: An I4.0 component connects the asset and the administration shell, based on [PAB+16]

Figure 2.5 visually represents the relationship between an asset and its function as an I4.0 component. It specifies that while an asset may not inherently possess I4.0 characteristics, it becomes an I4.0 component when equipped with a "administration shell" and at least passive communication capability. This transformation encapsulates the essence of I4.0 components, highlighting their crucial function as interfaces between the physical and digital worlds.

The adaptability of I4.0 components to assets with various communication capabilities and varieties demonstrates their flexibility. Components of I4.0 provide graduated protection for services and data specific to their applications. These components are utilized across the industrial spectrum, from production systems to individual devices. The administration shell is the key interface, storing vital information regarding the asset's attributes and functionalities, structured according to RAMI4.0.

As a result, Industrie 4.0 components are the fundamental framework for modern industrial operations. Their ability to integrate physical assets with digital connectivity increases productivity, information flow, and safety. This integration, demonstrated by the administration shell and its connection to assets, enables the ability of Industrie 4.0 components to transform current industrial landscapes.

### 2.4.2. Properties and Functionality

I4.0 components are distinguished by their unique properties. They exhibit active or passive I4.0-compliant communication capabilities and can be identified as entities. Their primary function is to represent assets with information, and they may encompass a wide spectrum of technical capabilities. These components are tailored to their intended purposes under the administration shell's security capabilities, ensuring appropriate security levels.

### 2.4.3. Identifiability and State

Unique identifiability characterizes I4.0 components, allowing them to be found within the network. Their dynamic state throughout their lifecycle is continuously queryable, facilitating local and global administration tasks. Distinguishing between asset types and instances enables establishing relationships between component producers and users.

### 2.4.4. Communication and Security

For communication, I4.0 components rely on service-oriented architecture with standardized semantics. Services provide self-information, while profiles prescribe technology and service specifics. They support multiple communication protocols, which ensures adaptability and compatibility. Security is the most significant concern, with information availability, confidentiality, and integrity at its core. Data shared across organizational boundaries takes anonymization and identity management into consideration.

### 2.4.5. System Composition and Functionality

I4.0 systems consist of I4.0 components along with lower-level components. These systems serve specific purposes, exhibit standardized properties, and support standardized services and states. I4.0 components can form nested structures, enhancing the system's versatility. Encapsulability ensures core functionality remains unaffected by external disruptions, as shown in Figure 2.6 and Figure 2.7, respectively.

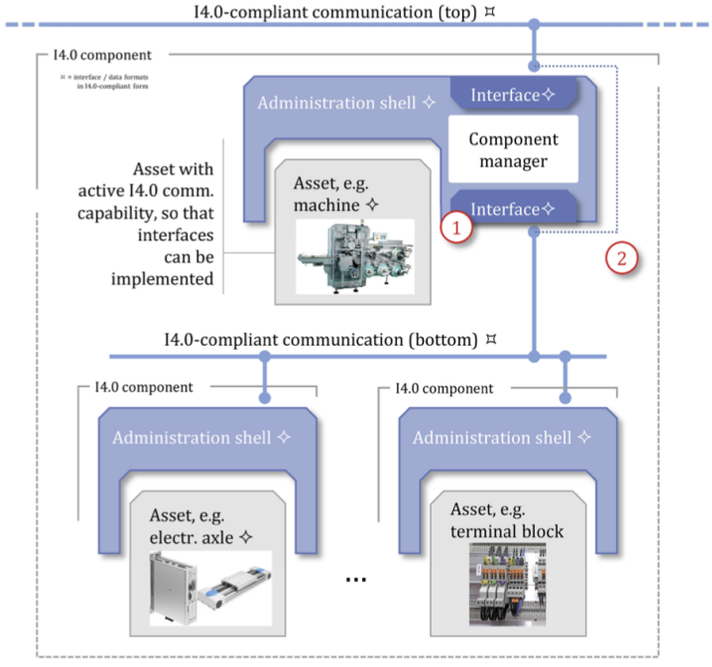



Figure 2.6.: Nestability of I4.0 components, based on [PAB+16]

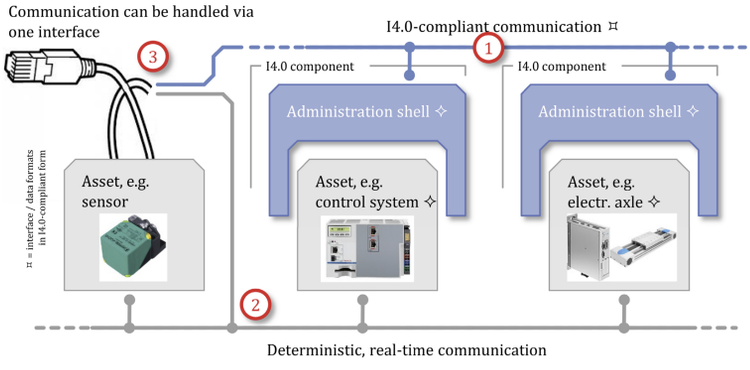



Figure 2.7.: Encapsulability of I4.0-compliant real-time communication, based on [PAB+16]

In conclusion, Industrie 4.0 components are the fundamental foundations of the I4.0 framework, enabling seamless communication, efficient administration, and secure interactions in dynamic industrial environments.

### 2.4.6. Administration Shell of I4.0 Components

The administration shell is a fundamental concept in I4.0 that transforms assets into active I4.0 components, offering a virtual representation within the digital framework. In embedded systems, the administration shell can reside within the asset itself or be distributed across IT systems. It captures life cycle data, converting it into accessible information, and this information is made available through views. The component manager, equipped with an I4.0-compliant service-oriented API, handles services and maintenance.

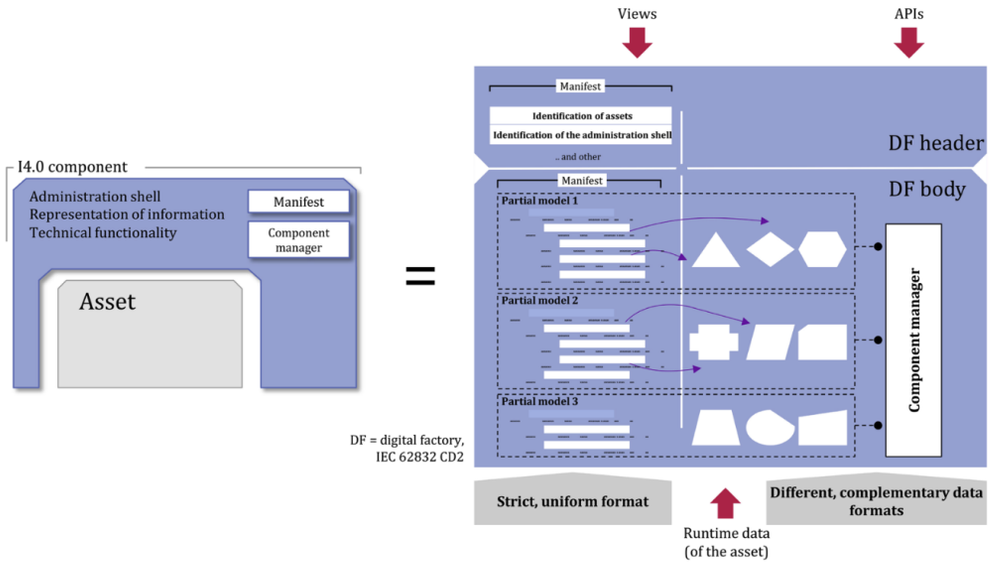
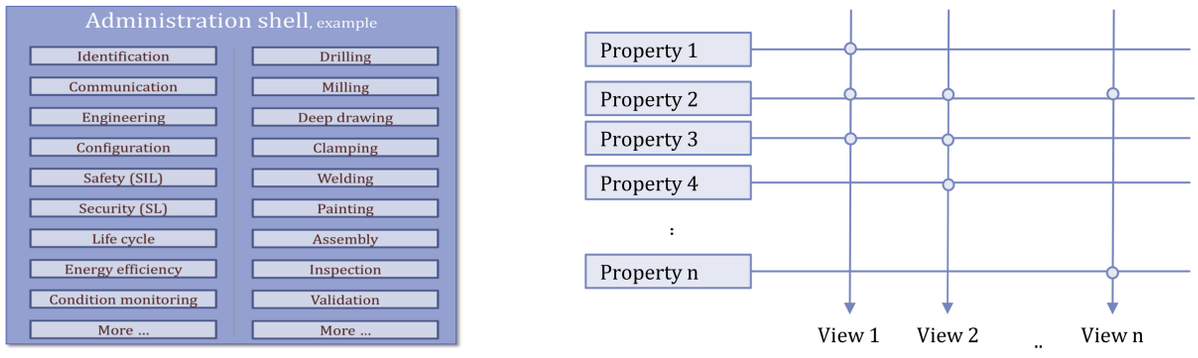



Figure 2.8.: Diagram of an I4.0 administration shell, based on [PAB+16]

Figure 2.8 depicts the administration shell's structure, including the DF header and body. The header identifies the asset within the I4.0 system, while the body stores the asset's actual data. This structure is divided into partial models tailored for specific purposes, and views within these models provide varied perspectives through properties, data, and functions.

Properties within the administration shell are hierarchically organized and can reference other properties and I4.0-compliant entities, contributing to a network of linked information. They must ensure information security, considering availability, integrity, and confidentiality, as shown in Figure 2.9.




(a) Examples of domain specific models

(b) Diagram of how views are created

Figure 2.9.: Partial Models and Views, based on [PAB+16]

Managed by the component manager and manifest, the administration shell ensures the organization, administration, and maintenance of information. Identifiers link assets and administration shells unambiguously. The administration shell can characterize both types and instances of assets, allowing for the universal linking of information. It can also reference other administration shells or non-I4.0-compliant

Fundamental requirements for the administration shell encompass structured header and body components, accessibility through service-oriented architecture, and the inclusion of crucial elements like the manifest and component manager. Additionally, it supports nesting, categorizes types and instances, and accommodates references, extra properties, and a minimum set of defined properties for reliability and consistency.

## 2.5. Digital Twin

This section explores the digital twin concept, including its relationships within systems and its implementation in various use cases.

### 2.5.1. The Concept and History

Since its initial introduction in 2003 by Grieves in his product lifecycle management course, the concept of a digital twin has experienced substantial development. It signifies a digital representation of a physical entity's attributes, behaviours, and interconnections. Initially conceptualized with three core components—the physical product, the virtual product, and their interconnections—the term has evolved with multiple definitions. In this evolution, the Industrial Internet Consortium (IIC) has comprehensively characterized a digital twin representing an entity with attributes and models, facilitating interaction through service interfaces. This comprehensive concept supports entity-related data's communication, storage, analysis, and prediction.

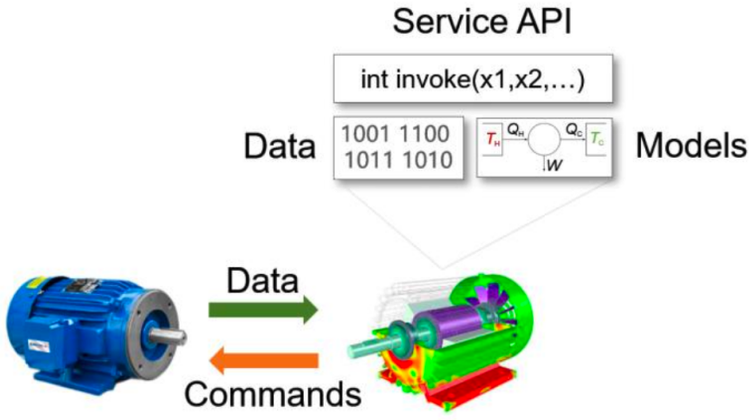



Figure 2.10.: Digital Twin – Data, Models and Service Interfaces, based on [BML+20]

Figure 2.10 illustrates the concept of a digital twin. It comprises three core aspects: data, models, and service interfaces. The data aspect encompasses vast amounts of information, including design specifications, manufacturing processes, usage data, and transactional records. The challenge lies in collecting, processing, and managing this data for effective monitoring and analysis. The models' aspect involves computational, geometrical, and visualization models that mirror real-world entities' characteristics. These models facilitate simulations and predictive analytics. The service interface aspect facilitates interaction between software applications and the digital twin, enabling connectivity and interoperability.

### 2.5.2. Relationships Among Digital Twins in Systems

The relationships among digital twins can be hierarchical, associational, or peer-to-peer. Hierarchical relationships involve assembling component digital twins into more complex entities, while associational relationships represent interactions between digital twins, mirroring real-world associations. Peer-to-peer relationships are observed among similar equipment performing similar functions.

### 2.5.3. Use Cases

Digital twin concepts find application across various domains. In smart manufacturing, digital twins streamline data collection and enable advanced applications such as predictive maintenance and energy efficiency optimization. Automotive applications encompass performance assessment and maintenance improvement. Supply chains benefit from comprehensive risk assessment while building automation leverages consolidated data for efficient property management. The concept of digital twins extends to smart cities, aiding in critical infrastructure analysis, visualization, and planning. One example can be seen in Figure 2.11.

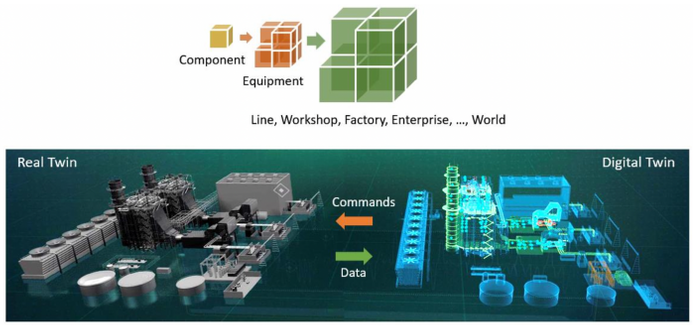



Figure 2.11.: Digital Factory Representation from Twins, based on [BML+20]

In essence, the digital twin concept bridges the physical and digital realms, offering a powerful tool for monitoring, prediction, and optimization across diverse industries.

## 2.6. Asset Administration Shell (AAS)

checked As described in The Asset Administration Shell (AAS) is a central concept introduced by Plattform Industrie 4.0 to drive interoperability, a critical facet of open ecosystems and cooperation. Initially proposed as part of the RAMI 4.0 in 2015, AAS represents the digital twin of an object, transforming it into an Industrie 4.0 Component. Collaboration between Plattform Industrie 4.0 and Industrial Internet Consortium (IIC) led to the alignment of the two reference architectures, with AAS being unique to RAMI 4.0

In 2017, the Plattform Industrie 4.0 and IIC started a collaboration to map and align the two reference architectures, Plattform Industrie 4.0's RAMI4.0 and IIC's Industrial Internet Reference Architecture (IIRA). The concept of the I4.0 component and, thus, of its AAS was identified as unique to RAMI 4.0. Since then, it has been observed that the concepts of digital twin and AAS are evolving in the same direction. This is why today, the AAS is described as implementing a digital twin for Industry 4.0.

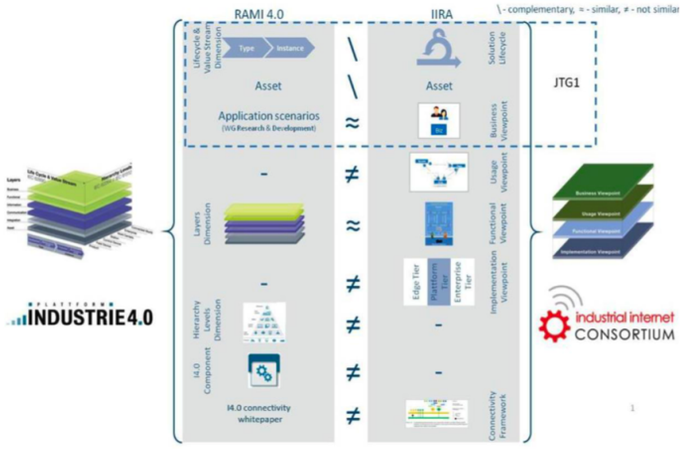



Figure 2.12.: Comparison References Architectures of Plattform Industrie 4.0 and IIC, based on [BML+20]

AAS specifications, introduced in 2018 and subsequently revised, facilitate interoperability across asset value streams. This aids the substitution of assets without altering their digital twin's information model and API. The IEC Workgroup WG24, focusing on Asset Administration Shell for Industrial Applications, underscores the growing significance of AAS. The AAS employs a technology-neutral information model, offering various formats for interoperability, such as XML, JSON, RDF, OPC UA, and AutomationML. AASX, a package format for offline information exchange, is also defined in Figure 2.13.

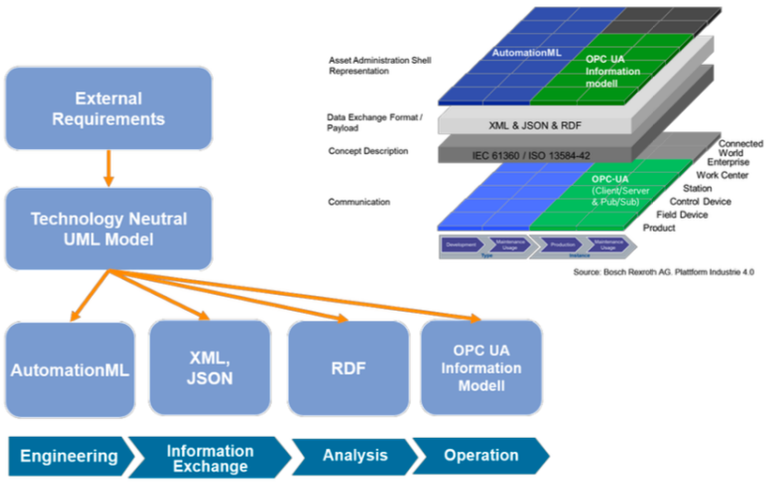



Figure 2.13.: Serializations and Mappings for AAS, based on [BML+20]

Figure 2.14 depicts comprehensive AAS comprises metadata about the asset it represents and submodels that cater to specific aspects related to the asset and relevant use cases. The information model of AAS consists of elements like assets, submodels, and concept descriptions, each globally and locally identifiable. Submodels encompass submodel elements, including properties, operations, and collections, all semantically defined. The AAS ensures security through access permission rules, guaranteeing access to designated objects to authenticated subjects.

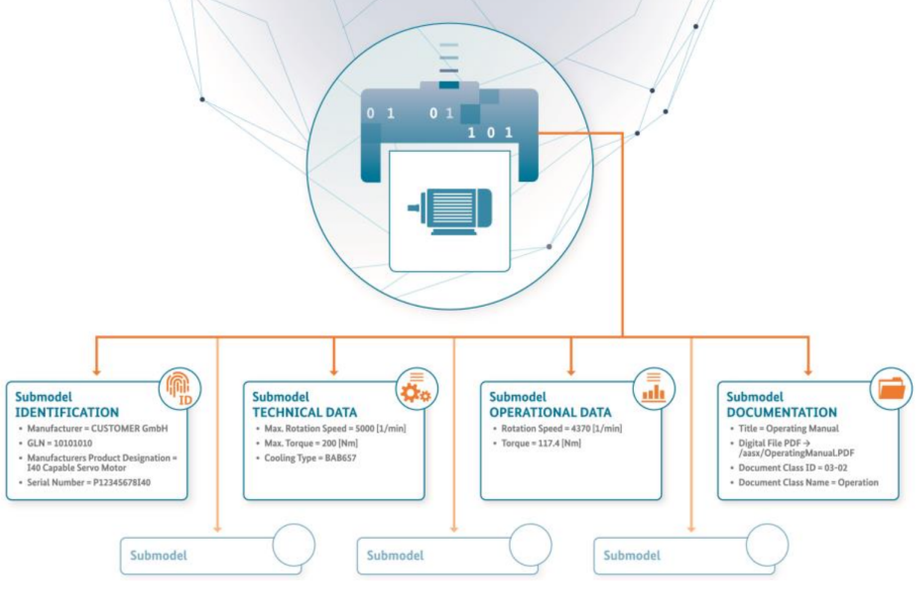



Figure 2.14.: The Asset Administration Shells with its Submodels, based on [BML+20]

In summary, the Asset Administration Shell is pivotal in driving interoperability within Industry 4.0, offering a standardized framework for digital twin implementation and facilitating efficient information exchange across assets and applications.

## 2.7. RevPi Core S

checked The RevPi Core S is a compact and powerful industrial computing module designed for automation and IoT applications. It is a product from Kunbus, a German company specializing in industrial communication and automation solutions. "RevPi" originates from "Revolution Pi," representing a platform tailored for industrial use cases.


Figure 2.15.: RevPi Core S, based on [Pi20]

### 2.7.1. Overview

At its core, Revolution Pi embodies the fusion of physical systems and digital intelligence. It encapsulates the principles of Industry 4.0, bringing together automation, real-time data analysis, and connectivity. The name "Revolution Pi" itself alludes to a paradigm shift in industrial practices, where the traditional boundaries between human-operated systems and intelligent machines are transcended.

### 2.7.2. Integration of IoT and Industrial Automation

Revolution Pi exemplifies the seamless integration of IoT technologies with industrial automation. It leverages IoT's ability to gather and analyze vast data in real-time. Connecting various devices and sensors enables industries to monitor processes, predict maintenance needs, and optimize operations for enhanced efficiency.

### 2.7.3. Applications

The applications of Revolution Pi are vast and diverse. It finds utility across numerous sectors, including manufacturing, energy, transportation, and agriculture. In manufacturing, it facilitates predictive maintenance, reducing downtime and increasing productivity. In energy management, it aids in monitoring and optimizing energy consumption. Moreover, Revolution Pi can contribute to smart traffic management and logistics in transportation.

### 2.7.4. Future Implications

The adoption of Revolution Pi signals a transformative era in industrial practices. As industries continue to embrace digitization, the concept promises to enable advanced data-driven decision-making, increased automation, and the creation of truly interconnected smart systems.

The RevPi Core S, as a component of the Revolution Pi platform, embodies these principles by offering a compact and versatile solution for industrial computing needs, as shown in Figure 2.15.

### 2.7.5. Features

The RevPi Core S often incorporates the following features:

- **Compact Design**: The RevPi Core S is intentionally designed to be small yet robust, ensuring its suitability for constrained spaces.
- **I/O Expansion**: The module is equipped with a range of I/O ports and interfaces, enabling seamless connectivity and control over various industrial devices and sensors.
- **Industrial Communication**: Expect the RevPi Core S to support industrial communication protocols such as Modbus, PROFINET, Ethernet/IP, and more. This versatility allows it to integrate seamlessly into existing industrial networks.
- **Real-time Capabilities**: The RevPi Core S might offer real-time processing options for applications requiring real-time processing precision, enhancing control and response times.
- **Linux-based OS**: The module typically runs on a Linux-based operating system, providing a stable and customizable environment conducive to industrial applications.
- **Programming and Configuration**: The RevPi Core S can likely be programmed and configured using popular languages and development tools in industrial automation, including Python, C/C++, and more.
- **DIN Rail Mounting**: To ensure seamless integration into industrial setups, the RevPi Core S may offer DIN rail mounting, allowing easy installation within control cabinets.
- **Watchdog Feature**: The RevPi Core S has a hardware watchdog mechanism that ensures system stability. The application software must restart the watchdog timer periodically to prevent automatic system restarts and disturbances during development.

In conclusion, Revolution Pi Core S represents a pivotal shift in the industrial landscape. Integrating IoT, automation, and data analysis marks a significant leap towards Industry 4.0's vision of intelligent, interconnected, and efficient industrial processes.

# 3. Project Management

This chapter outlines the project management process for the Industry 4.0 workpiece transfer unit and its corresponding communication capabilities. The project aims to develop a asset that satisfies specific requirements for handling workpieces and communication with other devices using Industry 4.0 standards.

## 3.1. Requirements

The requirements of the project outline the essential features, functionalities, and objectives that need to be fulfilled for the successful completion of the project. The requirements are categorized into functional ,non-functional, and connectivity requirements.

### 3.1.1. Functional Requirements

#### Seamless Module Integration and Quick Adaptation

The Industry 4.0 workpiece transfer unit must exhibit enhanced modularity and adaptability to accommodate changing production requirements. The unit must gain the capacity for seamless module integration, reconfiguration, and rapid exchange or replacement. The efficient adaptation to dynamic production demands and enhancement of the unit's flexibility, adaptability, and operational agility within the digital factory ecosystem must be provided.

#### Improved Interconnectivity and Data Exchange

The Industry 4.0 workpiece transfer unit must achieve improved interconnectivity and data exchange among its modules. The unit must expose operational data, machine status, maintenance information, and insights. This heightened visibility empowers operators to manage production workflows effectively. The unit must allow operators precise control over the Universal Robot UR5e, enhancing operational efficiency and overall digital factory flexibility.

#### Streamlined Production Management and Decision-making

The Industry 4.0 workpiece transfer unit must offer comprehensive, constant access to nameplate information, user manuals, technical specifications, and other crucial data. The unit must empower stakeholders with constant access to critical operational insights. This enables informed decision-making, robust production planning, and optimized resource allocation.

#### Operate in All Six Directions

The Industry 4.0 workpiece transfer unit must operate in all six directions within the digital factory environment to manipulate versatile workpieces. This operational flexibility allows the unit to collaborate seamlessly with surrounding modules, contributing to the cohesive functioning of the digital factory ecosystem.

#### Standalone Operation

The Industry 4.0 workpiece transfer unit is required to possess the capability of standalone operation. This entails that the unit can function independently without constant external intervention. In order to achieve this level of autonomy, the unit must establish reliable data exchange and standardized information-sharing mechanisms. These mechanisms will enable seamless communication between the unit and other components of the digital factory ecosystem.

#### Virtual Representation

The Industry 4.0 workpiece transfer unit must provide a comprehensive virtual representation, including 3D CAD models and CAD drawings, that allows stakeholders to:

1. **Designers of Additional Modules**: Designers of additional modules within the digital factory must have access to these visual assets for various purposes, including workspace planning, reference, and documentation. This access empowers designers to ensure that their new modules align with the limitations and capabilities of the current workpiece transfer unit.
2. **Maintenance Personnel**: Maintenance personnel must have access to the 3D CAD assembly of specific components. This access empowers maintenance personnel to efficiently comprehend and efficiently perform assembly tasks for additional components, ensuring correct assembly procedures and optimal utilization.

#### 3D Equipment Model Maintenance

The Industry 4.0 workpiece transfer unit must assist maintenance stakeholders in renewing or reproducing 3D models of equipment and tools in case of damage or malfunction. This assistance includes:

1. **Availability of 3D Model Files:** The unit should store and make accessible 3D model files of equipment and tools used within the digital factory.
2. **Ease of Access:** Maintenance personnel should be able to easily locate and download 3D model files relevant to the equipment or tools that require replacement.
3. **Compatibility:** The provided 3D model files must be compatible with common 3D printing software and hardware, allowing maintenance personnel to use these files for 3D printing directly.
4. **Documentation:** Alongside 3D model files, relevant CAD drawing documentation for the printed equipment or tools must be made available to ensure correct and safe reproduction.

This assistance empowers maintenance stakeholders to efficiently renew or reproduce equipment and tools using 3D printing technology, minimizing downtime and enhancing maintenance capabilities.

### 3.1.2. Non-Functional Requirements

#### CP Notation Positioning

The workpiece transfer unit must classify its current position and, after digitalization, within the Classification of Presentation and Communication (CP) notation. This facilitates effective task coordination and understanding of the unit's status. This foundation enhances subsequent digitalization efforts by providing precise position information for strategic actions.

#### RAMI 4.0 Conformance

The Industry 4.0 workpiece transfer unit must adhere to the Reference Architecture Model for Industry 4.0 (RAMI 4.0) principles. Aligning with RAMI 4.0 principles ensures compatibility with Industry 4.0 standards, enabling seamless integration and interoperability with other components of the digital factory ecosystem.

### 3.1.3. Connectivity Requirements

#### Industry 4.0 Protocol and Interface

The used protocol and interface for connecting to any other modules must match Industry 4.0 requirements and be an industry standard. The Industry 4.0 workpiece transfer unit's connectivity protocol and interface must adhere to Industry 4.0 standards, ensuring seamless integration and communication with other components of the digital factory ecosystem. The chosen protocol and interface should be widely recognized within the industry to facilitate interoperability.

## 3.1.4. Documentation Requirement

### Comprehensive Project Background

This project report must comprehensively overview various concepts related to developing the Industry 4.0 workpiece transfer unit. It should convey the fundamental principles and context necessary for understanding the objectives and significance of the unit's digitalization within the broader scope of Industry 4.0.

As a result, the table 3.1 summarises the most important requirements for the technical solution. These requirements are the results of a task analysis. The requirements are checked for their fulfilment at the end of the project and thus form a benchmark for its success.

| ID | Priority | Requirement | Type | Description |
|---|---|---|---|---|
| Req.1 | High | Seamless Module Integration and Quick Adaptation | Functional | The Industry 4.0 workpiece transfer unit must exhibit enhanced modularity and adaptability to accommodate changing production requirements. |
| Req.2 | High | Interconnectivity and Data Exchange | Functional | The Industry 4.0 workpiece transfer unit must be able to allows operators to monitor and control over the Universal Robot UR5e. |
| Req.3 | High | Streamlined Production Management and Decision-making | Functional | The Industry 4.0 workpiece transfer unit must provide access to nameplates, user manuals, and technical specifications. |
| Req.4 | Medium | Operate in All Six Directions | Functional | The Industry 4.0 workpiece transfer unit must operate in all six directions within the digital factory environment. |
| Req.5 | High | Standalone Operation | Functional | The Industry 4.0 workpiece transfer unit is required to possess the capability of standalone operation. |
| Req.6 | High | Virtual Representation | Functional | The Industry 4.0 workpiece transfer unit must provide a comprehensive virtual representation |
| Req.7 | High | 3D Model Replacement Assistance | Functional | Industry 4.0 workpiece transfer unit must assist maintenance stakeholders in renewing or reproducing 3D equipment and tool models when damaged. |
| Req.8 | High | CP Notation Positioning | Non-Functional | The project must classify its current position and after digitalization within the CP notation. |
| Req.9 | High | RAMI 4.0 Conformance | Non-Functional | The project must adhere to the principles of the RAMI 4.0. |
| Req.10 | Medium | Industry 4.0 Protocol and Interface | Connectivity | Industry 4.0 protocols and interfaces must be used to connect modules. |
| Req.11 | Medium | Comprehensive Project Background | Documentation | The report must convey essential concepts related to unit development and Industry 4.0 context. |

Table 3.1.: Industry 4.0 Workpiece Transfer Unit Requirements

## 3.2. Specification

The specification phase defines the functional, non-functional, and connectivity aspects of the Industry 4.0 workpiece transfer unit's incoming detailed design. This phase ensures alignment between project requirements and technical implementation by bridging the gap. Functional specifications describe unit behaviours, whereas non-functional specifications ensure conformity with industry standards and Industry 4.0 principles. These specifications constitute a technical plan that guides the development and implementation of the project.

### 3.2.1. Functional Specification

#### Seamless Module Integration and Quick Adaptation

The Industry 4.0 workpiece transfer unit will be designed to integrate seamlessly with other modules, allowing for rapid adaptation to changing production requirements. This design strategy adheres to the Industry 4.0 principles, emphasising modular, flexible, and adaptable manufacturing systems. Integrating the Asset Administration Shell (AAS) framework will enable the unit to rapidly reconfigure or swap out modules, a key characteristic of Industry 4.0 manufacturing environments..

#### Improved Interconnectivity and Data Exchange

The Industry 4.0 workpiece transfer unit will standardise information between digital factory modules using the AAS framework. Applying the AAS, the UR5e can demonstrate its capabilities and provide valuable operational data, such as queue state, machine status (e.g., busy, idle), and maintenance data. This data will be accessible to other modules and external systems, allowing for efficient data exchange. The integration with the AAS will also enable operators to control the pick-and-place operations of the Universal Robot UR5e by specifying the intended picking and placing directions and gaining access to other operation-related features.

#### Streamlined Production Management and Decision-making

The AAS integration of the Industry 4.0 workpiece transfer unit will provide a centralised repository for technical documentation such as nameplate data, user manuals, and technical specifications. The unit provides the integrator with comprehensive module operational insights by adhering to industry 4.0 standards and exposing this data via the AAS. This increases production efficiency and competitiveness through informed decision-making, production planning, and resource allocation.

#### Operate in All Six Directions

The Industry 4.0 workpiece transfer unit will feature the six-axis movement Universal Robot UR5e robotic arm. The UR5e will be programmed to perform precise and versatile workpiece manipulation in all six directions, including linear motion along the X, Y, and Z axes and rotational motion around the roll, pitch, and yaw axes. The configuration of the UR5e will allow the machine to manage workpieces of various orientations. This capability will facilitate collaboration between adjacent modules, allowing for the efficient transfer and positioning of workpieces in a digital industrial environment.

#### Standalone Operation

The project will implement an OPC UA server in accordance with the Asset Administration Shell (AAS) framework. This OPC UA server will be hosted on a RevPi Core S, providing the platform required for data exchange and communication. The standalone operational mode denotes that the Industry 4.0 workpiece transfer unit can function independently, making decisions based on its programming and incoming instructions. The OPC UA server facilitates dependable data exchange and communication, increasing the unit's autonomy within the larger manufacturing system.

#### Virtual Representation

The Industry 4.0 workpiece transfer unit's AAS integration will provide 3D CAD files in eDrawing Web HTML format and CAD Drawing PDF files, ensuring accessibility for various purposes, including workspace planning and reference. This approach enables users and stakeholders to conveniently access and utilize the visual assets for their specific needs.

#### 3D Equipment Model Maintenance

The AAS integration of the Industry 4.0 workpiece transfer unit will provide maintenance stakeholders with comprehensive support for renewing or reproducing 3D models of damaged or malfunctioning equipment and tools. This support consists of the availability of 3D model files in STL format for each required component, visualisation of assembly using 3D CAD files in eDrawing Web HTML format, assembly instructions in CAD Drawing PDF files, and compatibility with standard 3D printing software and hardware. This comprehensive support enables maintenance personnel to replace or reproduce equipment and tools efficiently, minimising downtime and enhancing maintenance capabilities.

### 3.2.2. Non-Functional Specification

#### CP Notation Positioning

Accurate CP notation positioning requires a two-step procedure. Initially, the asset of the Industry 4.0 workpiece transfer unit has been precisely defined, establishing a solid comprehension of its properties and capabilities. Then, a thorough evaluation of the Communication Capability and Presentation of the asset within the information system is performed. Following the asset's definition and communication capability analysis, the CP notation positioning of the Industry 4.0 workpiece transfer unit will be realised.

#### RAMI 4.0 Conformance

The Industry 4.0 workpiece transfer unit will be designed and implemented according to the Reference Architecture Model for Industry 4.0 (RAMI 4.0) principles. This design strategy will ensure the unit's architecture, communication protocols, and data formats comply with Industry 4.0's broader standards. By adhering to RAMI 4.0, the unit will also support "Digital Twins," where a virtual representation of the unit's physical counterpart can be created, strengthening monitoring, simulation, and optimisation capabilities.

### 3.2.3. Connectivity Specification

#### Industry 4.0 Protocol and Interface

The connectivity solution implemented for the Industry 4.0 workpiece transmission unit will employ OPC UA (Open Platform Communications Unified Architecture) as the industry-standard communication protocol. OPC UA enables secure and standardised data exchange between nodes, assuring interoperability and Industry 4.0 compatibility. By adopting OPC UA, the Industry 4.0 workpiece transfer unit will be well-positioned to collaborate with other modules and entities within the digital factory ecosystem in accordance with the broader Industry 4.0 objectives for connectivity and interoperability.

### 3.2.4. Documentation Specification

#### Comprehensive Project Background

The project report will provide a comprehensive overview of the various concepts of developing the Industry 4.0 workpiece transfer unit. It will communicate the fundamental principles and context required to comprehend the objectives and significance of the unit's digitalization within the context of Industry 4.0. This documentation will allow the reader to comprehend the project's foundations and objectives.

## 3.3. Work Plan

The V-model is a common systems development life cycle model that prioritizes testing and verification to ensure system quality. This model emphasizes the necessity of conducting testing activities concurrently with system development, with testing activities intensifying as the system approaches completion. Figure 3.1 depicts the project's work plan phases, which outline the schedule and tasks necessary to complete the project.

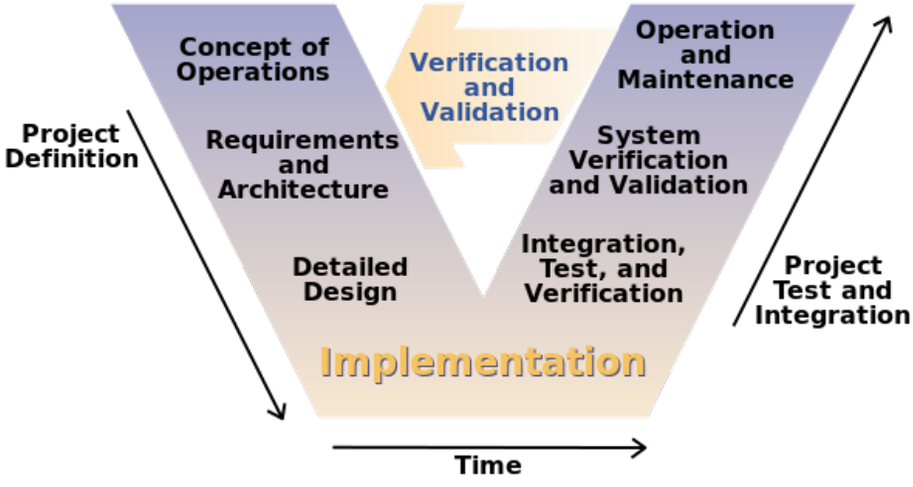



Figure 3.1.: V-model for project management, based on[Col21]

By adopting the V-model approach, the workpiece transfer system project can benefit from a structured development process that ensures requirements are clearly defined and met and the system is thoroughly tested. It is deployed and maintained to align with user needs and expectations. The phases in the work plan for the workpiece transfer system project are as follows:

By adopting the V-model methodology, the project can take advantage of a structured development process that ensures requirements are explicitly defined and met and the system is thoroughly evaluated. It is implemented and maintained in line with user requirements and anticipations. The following are the phases of the work plan for the project involving the development of Industry 4.0 workpiece transfer unit:

### Phase 1: Requirements Analysis

During this phase, the project's requirements are defined and analysed. This includes gathering and analysing user requirements, creating a comprehensive requirements specification, and validating the requirements with stakeholders. Included in this phase are:

- Analyzing functional, non-functional, and connectivity project requirements.
- Collaborating with stakeholders to refine and prioritize requirements.

### Phase 2: Detailed Design

During this phase, the concepts defined in the functional, non-functional, and connectivity specifications are translated into a detailed design that serves as the foundation for the implementation of the system. During this phase, the following objectives are achieved:

1. **Asset Definition**: Define the Industry 4.0 workpiece transfer unit as a fundamental asset within the digital factory ecosystem. This encompassing task entails specifying both its software and hardware components.
2. **Positioning Within CP Notation Framework**:Prior to and following the digitalization process, the workpiece transfer unit's Communication and Presentation (CP) notation positions are defined with exactitude.

3. **RAMI 4.0 Positioning Specification**: Establish the Industry 4.0 workpiece transfer unit's position within the Reference Architecture Model for Industry 4.0 (RAMI 4.0). Determine how the unit interacts with other elements of the digital factory ecosystem based on RAMI 4.0 principles.

4. **AAS Structure Specification for the Defined Asset**: Establish the architecture of the Asset Administration Shell (AAS) dedicated to the workpiece transfer unit. This involves crafting a comprehensive blueprint encompassing the information and interfaces slated for representation within the AAS. This involves identifying key data points and services the unit will expose through the AAS.

5. **Software Architecture Design**: Design the software architecture to drive the functionalities of the Industry 4.0 workpiece transfer unit. Determine the modules, their interactions, and the data flow within the unit.

By the end of Phase 2, the detailed design that lays the foundation for implementing Industry 4.0 workpiece transfer units will be fully defined and documented. This design will serve as the blueprint for subsequent project development phases.

### Phase 3: Implementation

The previous phase's comprehensive design is transformed into a physical product during this phase. The implementation phase comprises several essential duties, each of which contributes to the realization of the Industry 4.0 workpiece transfer unit:

1. **Configuration of UR5e Robot Programming**: Configure the Universal Robot UR5e by updating its programming to execute precise and dynamic manipulations in all six dimensions, adding variables to its program to represent the robot's status, and aligning it with the project's requirements. The modification ensures the workpiece transfer unit has the operational flexibility necessary for collaborating with adjacent modules within the digital factory ecosystem. In addition, the URP programming will be updated to reveal the precise status of the robot for future applications.

2. **Creation of Asset Administration Shell for Defined Asset**: Develop the Asset Administration Shell (AAS) tailored to the industry 4.0 workpiece transfer unit. This involves translating the AAS blueprint created in the previous phase into a functional digital representation that encapsulates the unit's data, interfaces, and capabilities.

3. **Creation of OPC UA Server - Aligned with AAS**: Create the OPC UA server while assuring its compatibility with the existing Asset Administration Shell (AAS). The OPC UA server will be the conduit through which the workpiece transfer unit communicates, allowing for safe and standardized data exchange with other components in the digital manufacturing environment.

4. **Deployment of OPC UA Server into RevPi Core S**: Implement the designed OPC UA server by deploying it onto the RevPi Core S hardware platform. This deployment facilitates the establishment of a communication hub that facilitates the interaction between the workpiece transfer unit and other interconnected modules within the larger manufacturing system.

By the end of Phase 3, the Industry 4.0 workpiece transfer unit will have taken its first steps toward actualization. This phase lays the groundwork for subsequent phases and facilitates the unit's incorporation into the digital factory ecosystem.

### Phase 4: Integration, Test, and Verification

In this crucial phase, the emphasis shifts to ensuring the integration, consistency, and functionality of the Industry 4.0 workpiece transfer unit. The system's compatibility for deployment is confirmed by extensive testing and verification procedures. The following responsibilities comprise this comprehensive procedure:

1. **Operational Validation of Robot Movements and Programming**: Perform a series of tests to verify the accuracy and dependability of the Universal Robot UR5e's operation in all six directions, ensuring that the updated programming configured in Phase 3 enables precise and dynamic manipulations. Verify

that the robot can effectively implement its functions, including the precise planar pick-and-place operation on the designated module, in accordance with the modified programming. In addition, the URP programming will be thoroughly evaluated to ensure that it exposes the precise status of the robot for subsequent use. This operational validation is necessary to ensure that the I4.0 workpiece transfer unit possesses the operational versatility required to collaborate seamlessly with adjacent modules within the future digital manufacturing ecosystem.

2. **Validation of AAS-based Information Access**: Verify that the Asset Administration Shell (AAS) framework effectively enables access to essential information, including nameplate data, user manuals, technical specifications, and 3D CAD models of the Industry 4.0 workpiece transfer unit. This validation demonstrates that the system is capable of providing essential operational insights.

3. **Control and Monitoring Verification via OPC UA Server**: Test thoroughly the OPC UA server, which serves as the control and monitoring interface for the Industry 4.0 workpiece transfer unit. Provide control capabilities by ensuring the server effectively translates user commands into unit actions.

4. **Continuous Operation Verification of OPC UA Server**: Validate the stability and constant operability of the OPC UA server within the RevPi Core S. Ensure that the server functions seamlessly, providing uninterrupted data exchange and communication between the workpiece transfer unit and the broader digital factory ecosystem.

By the culmination of Phase 4, the Industry 4.0 workpiece transfer unit will have undergone meticulous integration, testing, and verification procedures. The successful completion of these tasks instills confidence in the unit's operational capabilities, positioning it for deployment and validation in the subsequent phase.

### Phase 5: System Verification and Validation

In this phase, the focus turns towards the comprehensive verification and validation of the Industry 4.0 workpiece transfer unit. The primary objective is to ensure that the unit adheres to the initially defined requirements while encompassing any supplementary requirements that may have emerged during the development.

1. **Functional Requirements Verification**: The workpiece transfer unit is tested against its initial functional specifications. This step serves to confirm that the unit can perform its intended functions accurately and reliably.

2. **Non-Functional Requirements Verification**: Verification of non-functional requirements, such as CP Notation Positioning, and RAMI 4.0 Conformance, is conducted to ensure that the unit meets these critical criteria.

3. **Connectivity Requirements Verification**: Testing the unit's connectivity and compatibility with other components within the digital factory ecosystem is essential. This verification guarantees seamless integration with the broader system.

4. **Documentation Requirement Verification**: The documentation is reviewed for completeness and accuracy to ensure that the workpiece transfer unit's comprehensive project background is presented. It enables the reader to comprehend the foundations and objectives of the project.

This phase serves as the final confirmation before moving forward, ensuring the project is ready for subsequent stages.

### Phase 6: Deployment and Maintenance

During this phase, the Industry 4.0 workpiece transfer system is integrated into the production environment and maintained to ensure that it meets user requirements. Due to the ongoing development of the digital factory, however, certain components of this phase will be postponed until the factory is complete. Following the completion of the digital factory, the following tasks will be carried out:

1. **Transferring System Ownership**: The system will be transferred to the production team responsible for its ongoing operation and maintenance.
2. **Ongoing Maintenance and Support**: Consistent maintenance and support will be conducted to guarantee the system's continual correct operation and fulfillment of user requirements. This includes monitoring system performance, performing routine maintenance, and resolving emerging issues.

This phase marks the culmination of the project's lifecycle, ensuring the Industry 4.0 workpiece transfer system functions reliably within the operational framework of the digital factory.

# 4. Detailed Design

This chapter elaborates on the detailed design of the Industry 4.0 product transfer unit. This phase translates the requirements specified in the previous chapter into a comprehensive technical design that is the foundation for system implementation. In addition, the comprehensive design includes the unit's system architecture and how it will interact with other elements of the digital factory ecosystem.

This chapter describes the numerous aspects of the design process, such as the following: Section 4.1 presents the asset definition, describing the component of the unit, both hardware and software. Section 4.2 discusses the precise positioning of the defined asset within the CP notation, both prior to and following the digitalization process. Section 4.3 specifies the asset's position within the RAMI 4.0 model, emphasizing its interactions with other components. Section 4.4 outlines the structure of the Asset Administration Shell dedicated to the workpiece transfer unit and provides in-depth information regarding the asset. Finally, Section 4.5 delves into the design of the software architecture, detailing the modules, interactions, and data flow.

The culmination of the comprehensive design phase will be a technical plan that serves as a roadmap for subsequent development, integration, and testing phases. This chapter sets the groundwork for the successful development of the Industry 4.0 workpiece transfer unit by carefully addressing each design aspect.

## 4.1. Asset Definition

This project's defined asset focuses primarily on integrating the Universal Robot UR5e, a component of the workpiece transfer unit, into the digital factory. It includes hardware and software components that contribute to the project's success. The Universal Robot UR5e is essential in the workpiece transfer unit, allowing for precise positioning and controlled manipulation of workpieces, as depicted in the figure 4.1.

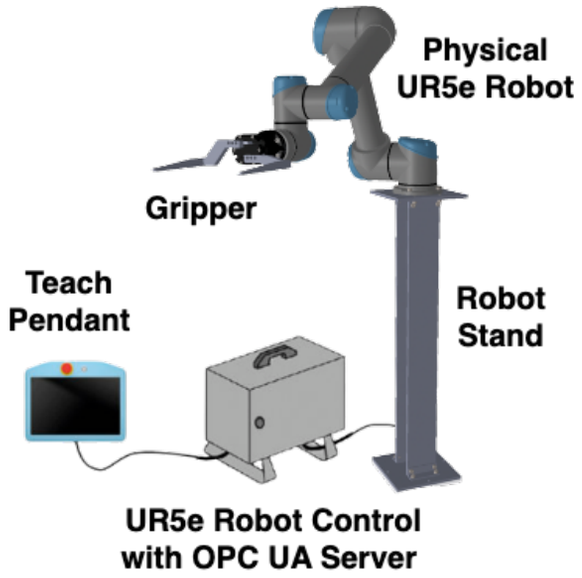



Figure 4.1.: Asset including hardware and software components, own illustration

### 4.1.1. Hardware Components

This section will thoroughly explain the hardware components of the Industry 4.0 workpiece transfer device. Collectively, these components, namely the Universal Robot UR5e (UR5e), Robotiq Hand-E gripper, Robot Stand, and Base Tool, define the unit's physical capabilities and functionalities within the digital manufacturing ecosystem.

#### Universal Robot UR5e (UR5e)

The hardware components of the asset are the foundation of the Industry 4.0 workpiece transfer unit, defining its physical capabilities and enabling its functionalities within the digital manufacturing ecosystem. At the foundation of these components is the Universal Robots version 5 of e-series (UR5e) collaborative robotic arm, which is renowned for its flexibility, precision, and adaptability. Consisting of the robot control, robot arm, and instruct pendant, the UR5e represents the unit's mechanical dexterity and mobility. Its six-axis design enables the intricate motions, positioning precision, and dexterity necessary for the manipulation of workpieces without interruption. [Uni18].

#### Robotiq Hand-E gripper

The gripping mechanism of this asset is founded on the Robotiq Hand-E gripper. This gripper type, distinguished by its parallel motion two-jaw design, provides a solid foundation for the workpiece manipulation duties in the digital factory. To meet the project's requirements, the conventional fingers of the Robotiq Hand-E gripper are replaced with a **3D printed gripper**. This custom-made design guarantees optimal compatibility with the unit's workpiece geometry and operational requirements. The configuration of the The **3D printed gripper** ensures a secure and efficient grip on workpieces, allowing for precise movement and positioning.

#### Robot Stand

The asset consists of a robot pedestal to complement the robot's capabilities. This stand serves a dual purpose: securely securing the robot and optimizing its height concerning neighboring modules. The stand guarantees precise and controlled movements during workpiece transfer operations by providing the robot with a stable foundation. In addition, the robot's adaptability to various production scenarios and tasks is increased by the adjustable nature of the stand.

#### Base Tool

A pivotal element in the asset's design is the Base Tool. This tool is pivotal in facilitating the seamless docking of modules to any of the six sides of the system. By providing predefined bases or planes, the Base Tool eliminates the need to reprogram the robot for every module and every side. This approach significantly reduces the programming effort and expedites the commissioning of new modules. The Base Tool's function of setting these bases enhances the precision and repeatability of the positioning process, a crucial aspect in achieving efficient and error-free workpiece transfer.

### 4.1.2. Software Component

The software architecture of the asset revolves around the Open Platform Communications Unified Architecture (OPC UA) protocol, a widely adopted standard for industrial communication and interoperability. The OPC UA protocol facilitates consistent and secure communication between different system components, enabling efficient control, monitoring, and data exchange within the digital factory [MLD09].

#### UR5e OPC UA Server

The UR5e incorporates an OPC UA server within its robot control, enabling wireless communication with external systems using the OPC UA protocol. The UR5e's OPC UA server serves as an interface between the UR5e and the broader digital ecosystem, facilitating the transfer of variables and operational data.

The defined asset consists of the Universal Robot UR5e, the Robotiq Hand-E effector, the Robot Stand, and the Base Tool, forming a cohesive unit that enables precise pick-and-place and service operations within the digital factory. The UR5e's OPC UA server is a crucial component, providing remote control and monitoring capabilities, such as access to position, trajectory, occupancy status, and maintenance data, and enabling wireless communication and the exchange of variables and operational data within the digital ecosystem. All of these components contribute to the asset's functionality, effectiveness, and seamless integration within the digital factory environment.

## 4.2. Positioning Within CP Notation Framework

˙ The preceding chapter, in particular, section 2.2 of the background chapter, elaborated on the concept of the Communication and Presentation notation (CP Notation). This notation is used to depict an asset within an information system and its communication capabilities. Nevertheless, within the context of the asset defined and discussed in section 4.1, the application of CP notation aims to clarify its functionalities and communication characteristics within the information system. CP Notation provides a valuable avenue for effectively showcasing the capabilities and communication aspects of the workpiece transfer unit, thereby providing profound insights into its role and seamless integration within the digital factory landscape [PAB+16].

The CP Notation diagram adopts a 2D format, delineating the Communication capabilities class (C-Class) along the horizontal axis and the Presentation in information system class (P-Class) along the vertical axis.

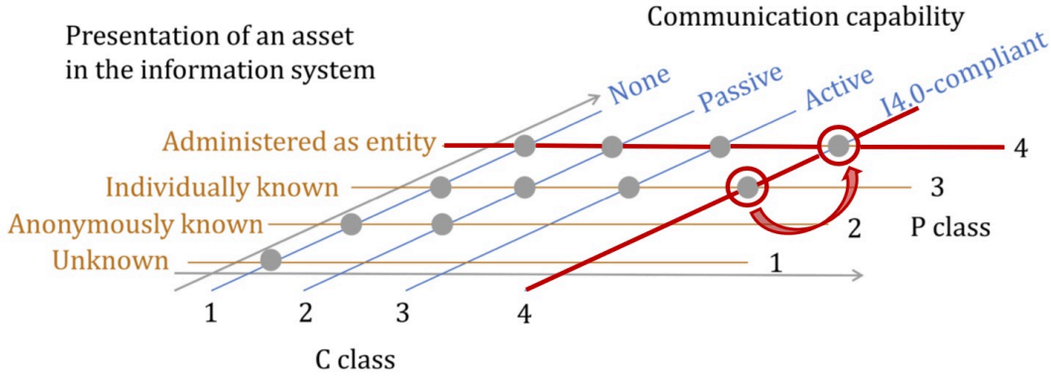



Figure 4.2.: CP Notation Diagram: Defined Asset Communication and Presentation, based on [PAB+16]

### Communication Capability

The Communication Capability axis of the C-Class class consists of four categories that represent assets with differing communication capabilities. These categories range from assets without communication capabilities to assets with communication capabilities compliant with I4.0.

Prior to the implementation of Asset Administration Shell (AAS) for the specified asset, it was already positioned in the "Assets with I4.0-compliant communication capability (I4.0 components)" segment along the Communication Capability (C-Class) axis. This indicates that the asset and its OPC UA server, which is the software component described in subsection 4.1.2, already possessed communication capabilities that adhered to the communication standards defined by the Industry 4.0 paradigm, specifically the "DIN SPEC 91345 RAMI 4.0" standard, as depicted in Figure 4.2.

### Presentation of asset in information System

The P-Class axis encompasses four delineations, representing varying levels of asset knowledge. Currently, for the presentation of the asset in the information system (P-Class) axis, the defined asset, including hardware and software components, is classified under the "Individually known assets" category. This denoted that the asset is identifiable and recognized as a distinct entity within the information system. However, its presentation lacked the comprehensive administration and standardized structure provided by the AAS.
In the P-class axis, upon digitalization through implementing the AAS for the asset, the asset will be transitioned to the "Assets administered as entities" segment. This indicated that the workpiece transfer unit's presentation and representation within the information system became more comprehensive and structured, aligning with the guidelines outlined in the "DIN SPEC 91345 RAMI 4.0" standard as shown in Figure 4.2.

## 4.3. RAMI 4.0 Positioning Specification

Incorporating the defined asset discussed in section 4.1 within the RAMI 4.0 model involves considering it an integral element within the broader context of a digital factory. Aligning the workpiece transfer unit, which encompasses the UR5e robot, its associated hardware components, and the OPC UA Server with the RAMI 4.0 Specification facilitates comprehensive integration and management strategies within the digital factory ecosystem. This approach ensures that the asset's role in workpiece transfer is effectively addressed, enhancing its functionality and contributing to seamless operations in the digital factory. Figure 4.3, providing an integrator's perspective, visually depicts the integration of the defined asset into the digital factory environment [PAB+16].

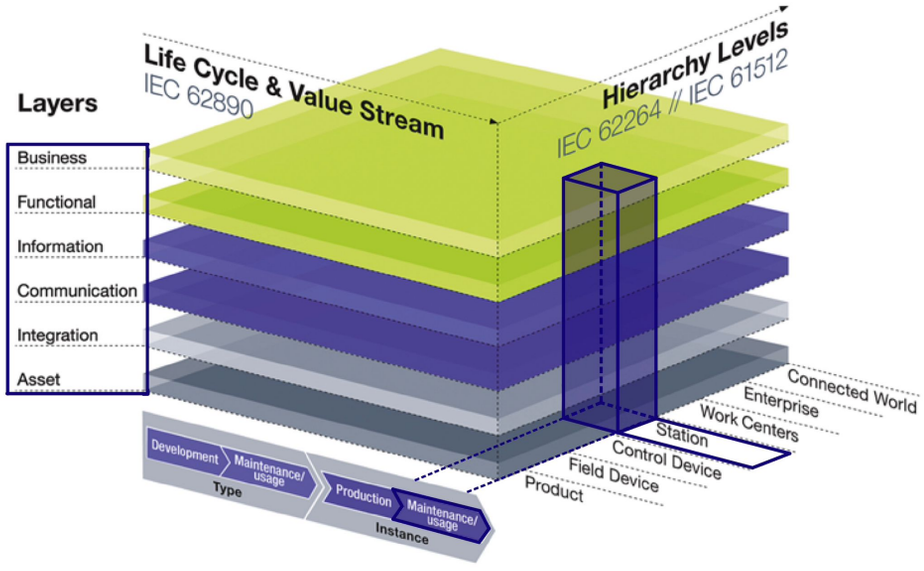



Figure 4.3.: RAMI 4.0 specification for the integrator view, based on [PAB+16]

Furthermore, the digitalization of the defined asset through implementing the Asset Administration Shell (AAS) yields a standardized and comprehensive representation of its functions, capabilities, and data. This transformation empowers the asset to assume the role of a digital twin within the RAMI 4.0 framework, providing an encompassing view of its digital representation. This transformation enhances integration and interoperability with other digital factory components.

### Life cycle axis

Within the RAMI 4.0 paradigm, the defined asset is located at the life cycle axis instance level. This positioning encompasses the asset's utilization and maintenance throughout the digital factory's operational phases. The asset ensures sustained operational efficiency by facilitating precise workpiece transfers and supporting ongoing maintenance activities, emphasizing its continuous contribution.

### Hierarchy levels axis

As a digital factory component, the defined asset can be positioned below the station level in the digital factory's hierarchical structure. The asset could also be classified as a work center but as a component of the digital factory, which consists of numerous stations, the classification as a station is more applicable.

### Layer axis

When considering digitalization, the defined asset can be placed in all layers because it affects all of them. Beginning with the business layer, it can be seen as a top-down approach.

- **Business Layer**:TThe primary purpose of this asset is to fulfill the integrator's requirements within the digital industrial environment to optimize production and permit scheduling. It improves modularity and adaptability by facilitating seamless module integration, reconfiguration, and interoperability. The asset facilitates enhanced interconnectivity and data exchange by providing valuable operational data and crucial information, thereby facilitating streamlined production management and informed decision-making. Its functionalities enhance the performance and operational efficacy of the digital factory.
- **Functional Layer**: The defined asset, also known as the workpiece transfer unit, mainly focuses on the Universal Robot UR5e and provides essential technical functionalities within the functional layer of the Industrie 4.0 system. It offers a wide range of functionalities as summarized below:
  1. **Data Storage and Organization**:
     - **Structured Data Storage**: Data Storage and Organization Storing diverse data of the defined asset in a structured and integrated manner.
     - **Information Management**: Managing essential information of the defined asset, such as manufacturer information, model specifications, and adherence to particular standards.
  2. **Operational Aspect Management**:
     - **Occupancy Status**: Optimizes production procedures and resource allocation by displaying the robot's activity status.
     - **Maintenance Status**: Provides details on ongoing maintenance and pinpoints its position within the digital factory.
     - **Operation Status**: Provides information regarding ongoing pick-and-place operations and identifies the location and module IDs within the digital factory.
     - **Queue Management**: Tracks workpieces in the queue, facilitating the scheduling and planning of production tasks for increased efficiency and reduced waiting time.
     - **Pick & Place Function**: Enables precise pick-and-place operations by allowing users to specify workpiece manipulation directions, module IDs, and positions.
     - **Maintenance Function**:Allows users to initiate service maintenance on a running Universal Robot UR5e, ensuring that maintenance operations are safe and effective.
  3. **Documentation Management**:
     - **Documentation Handling**: Facilitating the management of critical documentation, including user manuals, technical specifications, and maintenance procedures.
     - **Documentation Access**: Providing access to essential documentation contributes to a comprehensive understanding and efficient operation of the asset, thereby facilitating its operation.

4. **Virtual Representation Enhancement**:
   - **Workspace Layout Planning**: Allows users and stakeholders to plan and optimize the layout of their workspace within the digital factory.
   - **Design Integration and Testing Support**: Assists designers in virtually integrating new modules or components into existing units within the digital factory, enabling virtual testing and validation.
   - **Maintenance Visual Guide**: Assists maintenance stakeholders by providing a visual guide for assembling auxiliary components to the unit during maintenance procedures.
   - **Reference and Collaboration Support**: Serves as a reference for stakeholders, including users, designers, and maintenance personnel, facilitating collaboration among teams.
5. **Maintenance 3D Model Resources**:
   - **3D Model Repository**: Maintain a repository of 3D model files in STL format for components and tools used within the workpiece transfer unit. Ensure easy and quick access to these files when needed for maintenance purposes.

– **Information Layer**: The Information Layer of the Asset Administration Shell (AAS) for the workpiece transfer unit is a comprehensive repository for instance data that is structured. It includes a variety of vital details, including:

1. **Manufacturer Information**: This data contains specific information about the UR5e robot, the primary component of the workpiece transfer unit. It provides manufacturer information and comprehensive model specifications, including design and technical details, to ensure traceability and accountability.
2. **Operational Aspects**: This includes the unit's behavior, capabilities, parameters, and interconnections, providing a comprehensive view of its functionality within the digital factory ecosystem.
3. **Documentation**: Specific documentation relating to the UR5e robot, such as technical specifications and user manuals, provides essential information to facilitate efficient operation and maintenance.
4. **3D CAD Visualization**: The Information Layer contains a 3D CAD representation of the workpiece transfer unit, enabling users to interact with a virtual representation of the unit and its components. This visualization enhances comprehension, facilitates workspace optimization, aids in troubleshooting, and promotes well-informed decision making.
5. **CAD Drawings**: CAD drawings of the defined asset, including detailed technical drawings and schematics, provide additional resources for users and stakeholders.
6. **Repository of 3D Model Files**: This repository contains 3D model files in STL format for components and tools used within the workpiece transfer unit. These files are readily accessible and serve a critical role in maintenance and replacement procedures.

– **Communication Layer**: The Communication Layer serves as a standardized interface for communication between the various layers of the digital factory. The technology enables the secure and consistent exchange, management, and monitoring of system data. Utilizing the OPC UA specification, the Communication Layer of UR5e enables reliable and interoperable communication between the asset and other components in the digital manufacturing environment. [MLD09].

– **Integration Layer**: The Integration Layer functions as a bridge between the physical and digital domains, enabling the conversion of information from the physical world into digital data. In the context of the workpiece transfer unit, the UR5e robot controller plays a central role in this layer. This component is the primary mechanism for converting physical data into digital format. The significance of the UR5e OPC UA server lies in its capacity to facilitate communication, data exchange, and interoperability between diverse digital factory environment components. By functioning as an intermediary between the real and virtual worlds, the Integration Layer improves the system's overall coherence and performance.

– **Asset Layer**: As elaborated in section 4.1, This project's defined asset focuses on the workpiece transfer unit, with the Universal Robot UR5e as its central component. This unit combines physical and digital components for efficient manipulation of workpieces. The physical component consists of a UR5e robot with a flexible arm and control mechanisms, as well as a specialized gripper. The software component of the asset is the OPC UA Server, which is incorporated into the software for the robot controller. This integration of hardware and software yields an entity that optimizes the transmission of workpieces within the expansive framework of the digital factory.

## 4.4. AAS Structure Specification for the Defined Asset

As described in 2.6, Asset Administration Shell (AAS) is a fundamental concept that enables the integration, management, and interoperability of assets within the digital factory ecosystem. A critical aspect of realizing the potential of AAS is its structured representation, which defines the organization and composition of various asset-related data and functionalities.

Consequently, this section delves into the AAS structure specific to the defined asset, the workpiece transfer unit comprising the Universal Robot UR5e. The AAS structure for this asset consists of multiple submodels, each of which has been meticulously crafted to encapsulate particular aspects of the asset's behavior, attributes, and interactions. These submodels not only provide an exhaustive representation of the asset's characteristics, but also enable a holistic view of the asset's digital twin.

The following submodels are integral components of the AAS structure for the defined asset:

1. **Namesplate Submodel**: This submodel provides the asset's essential identification and classification data. It contains the asset's name, manufacturer information, and serial number. The Namesplate submodel is the foundation for comprehending the asset's fundamental characteristics.

2. **Operational Submodel**: The Operational submodel captures the operational state of the asset and provides insight into its current condition and availability. This submodel stores data regarding the unit's occupancy status, maintenance status, and ongoing operations and permits user control. By providing a dynamic perspective, the Operational submodel contributes to effectively utilizing assets.

3. **Documentation Submodel**: Ensuring informed decision-making and streamlined maintenance, the Documentation submodel acts as a repository of critical documents. These documents include user manuals, technical specifications, and maintenance procedures. This submodel enriches the asset's understanding and contributes to its optimal operation throughout its lifecycle.

4. **CAD Submodel**: The CAD Submodel significantly enhances the visual representation of the workpiece transfer unit and its components. It enables advanced virtual visualization, supporting workspace planning, design validation, and collaborative decision-making. The CAD Submodel enhances the unit's understandability and utility within the digital factory ecosystem by bridging the physical and digital realms.

5. **Model 3D Print Submodel**: Within the Model 3D Print Submodel, a repository of 3D model files is maintained for critical tools and components used within the workpiece transfer unit. The primary advantage of this submodel is its quick and simple accessibility, ensuring that maintenance personnel can promptly obtain necessary 3D model files for maintenance tasks. This accessibility significantly reduces downtime and enhances the unit's overall maintenance capabilities, strengthening its position within the digital factory environment.

In the forthcoming subsections, each of these submodels will be explored in-depth. The aim is to provide a comprehensive understanding of their purpose, data composition, and utilization within the AAS framework. By elaborating on these submodels, this section endeavors to present a detailed overview of how the defined asset's AAS structure empowers its seamless integration and effective management within the digital factory environment.

### 4.4.1. Nameplate Submodel

The Asset Administration Shell (AAS) Namesplate submodel provides a structured framework for organizing and managing asset-related information. It establishes a distinct identifier and context for assets, improving navigation and interoperability. In the context of the discussed workpiece transfer unit, the Namesplate submodel prominently displays information about the Universal Robot UR5e, its central component. As the primary manipulator of workpieces, the UR5e robot represents the capabilities and purpose of the unit. By encapsulating properties such as unique identifiers, manufacturer information, and product names, the Namesplate submodel promotes consistency and facilitates effective stakeholder communication and collaboration. It serves as the basis for managing assets throughout their entire lifecycle.

The properties included within the Namesplate submodel for the UR5e robot within workpiece transfer unit are as follows:

- **Uri of the product:** The Uniform Resource Identifier (URI) assigned to the UR5e robot is "https://www.universal-robots.com/pro It serves as a specific reference for locating and gaining access to robot-related information.
- **Manufacturer Name:** Universal Robots, a well-known and reputable company in the robotics industry, manufactures the UR5e robot.
- **Contact Information:** Multiple contact information is available for the UR5e product, including email address, telephone number, and contact person.
- **Manufacturer Product Designation:** The UR5e robot is designed to transport workpieces to their designated locations, demonstrating its purpose and functionality.
- **Manufacturer Product Root:** The UR5e robot belongs to the category of collaborative robots, indicating that it can work cooperatively with humans.
- **Manufacturer Product Family:** Among the products offered by Universal Robots, the UR5e robot is a member of the e-Series, a family of robotic systems renowned for their advanced capabilities.
- **Manufacturer Product Type:** The UR5e is a specific type or variant of the Universal Robots e-Series, featuring unique characteristics and specifications.
- **Serial Number:** The UR5e robot has a unique serial number, which is 20205500995 in this instance. This serial number allows for the identification and tracing of individual items.
- **Year Of Construction:** The UR5e robot was built in the year 2020, emphasizing its modernity and relative recency of construction.
- **Date Of Manufacture:** July 1, 2020
- **Software Version:** The UR5e robot utilizes version 5.8.0.10253 of UR-Software, which was released on March 22, 2020. This version of the software offers advanced control and functionality.
- **Country Of Origin:** Malaysia
- **Company Logo:** The Asset Administration Shell contains the logo of Universal Robots, the manufacturing company of the UR5e robot.

### 4.4.2. Operational Submodel

The Operational submodel within the Asset Administration Shell (AAS) forms the core of capturing and representing the operational facets of the workpiece transfer unit. This submodel plays a pivotal role in conveying the unit's real-time operational status, positioning, and workload aspects. By encapsulating properties that illuminate the availability, service status, position, and ongoing operations, the Operational submodel enables comprehensive monitoring and control of the workpiece transfer unit. It becomes a bridge between the digital representation and the physical functionality of the unit.

### Propertiy Submodel Elements

Within the Operational submodel, a range of Submodel Element Property (Prop) has been meticulously defined to precisely capture the dynamic status and positioning characteristics of the workpiece transfer unit. These properties include:

- **is busy:** A Boolean property representing the unit's operational status. It indicates whether or not the unit is actively engaged in a mission or operation. A "true" value indicates operational engagement, whereas a "false" value indicates availability.
- **is under service:** This Boolean property indicates whether the device is undergoing service or maintenance. A "true" value indicates active maintenance, whereas a "false" value implies no ongoing servicing.
- **is at service position:** A numeric variable indicates whether the device is currently at a predetermined service location. The integer value corresponds to the service position number assigned.
- **number of queue:** The number of tasks or operations in the unit's queue is an integer variable. This value provides information regarding the unit's load and upcoming operations.
- **current picking at module number:** An integer variable indicating the module number from which the unit is currently picking an item or object.
- **current placing at module number:** An integer variable identifying the module number where the unit is currently placing an item or object.
- **current picking at position:** An integer variable specifying the precise position within a module from which the unit is currently picking an item or object.
- **current placing at position:** An integer variable representing the specific position within a module where the unit is currently placing an item or object.

### Operation Submodel Elements

Under the Operational submodel, two Submodel Element Operation (Opr) have been defined: "pick and place" and "service." These operations act as methods or functions that can be executed when running the server in OPC UA.

1. **Pick & Place Operation**: The "pick and place" operation requires four input parameters: the pick id, the pick dir, the place id, and the place dir. These arguments encapsulate crucial information regarding the picking and placing processes, which corresponds to the unit's fundamental functions.
    - **pick id:** Identifies the module from which an item will be picked.
    - **pick dir:** Specifies the particular direction for picking the item within the selected module.
    - **place id :** Identifies the module where the item will be placed.
    - **place dir:** Specifies the particular direction for placing the item within the selected module.
2. **Service Operation**: The Service operation, integral to the Asset Administration Shell (AAS), empowers integrators to command the workpiece transfer unit to navigate to specific service locations for maintenance or service tasks.
    - **service id:** The service id input argument empowers integrators to specify the desired service location for the unit.
    - **result s:** A Boolean value conveying the unit's service status ("true" for under service, "false" for not under service).

### Range Submodel Elements

Ranges in the Asset Administration Shell (AAS) provide constraints and specifications for various system parameters. By defining specific ranges, such as load capacities and workspace dimensions, the AAS guarantees that the system operates within predetermined limits and under optimal performance conditions. Therefore, the AAS for the workpiece transfer unit must contain two ranges: "load capacity" and "workspace."

- **Load Capacity:** This range indicates the maximum allowable load capacity. It is defined as 0 to 2 kg, which indicates that the current UR5e robot within the unit is able to transport objects within this weight range.

- **Workspace:** This range defines the allowed workspace or operating range of the UR5e robot within the device. It specifies the distances that the end effector of the robot can extend in different directions. In this instance, the workspace dimensions are 151 mm to 1050,5 mm.

Consequently, the Operational submodel within the Asset Administration Shell (AAS) for the workpiece transfer unit includes numerous properties, operations, and ranges to represent and convey its operational aspects. These components provide information regarding the unit's status and location, enabling monitoring and control. In addition, the Range submodel specifies specific constraints and requirements for parameters such as load capacity and workspace dimensions, ensuring optimal performance within predetermined limits. Therefore, the following table 4.1 summarizes the elements included in the Operational submodel by integrating all of the information regarding the properties, operations, and ranges.

| **Submodel Element** | **Element** | **Description** |
|---|---|---|
| Property | Occupancy Status | The status indicates whether the unit is active or inactive. This information is necessary for optimizing production procedures and resource allocation. |
| | Maintenance Status | The status indicates whether or not the unit is being maintained. The asset identifies the digital factory location of maintenance in progress. |
| | Operation Status | Tracking the current operation of the unit.This indicate the picking and place position and module id |
| | Queue Management | Tracking the workpieces in the queue assists in scheduling and planning production tasks. This increases production output and decreases waiting time. |
| Operation | Pick & Place Function | The function offer the pick-and-place operations of the unit. Users are able to specify picking and placing directions, module IDs, and positions for precise workpiece manipulation. |
| | Maintenance Function | The function offer the maintenance operation of the unit. This function allows users to initiate service maintenance on a running unit. The unit will complete its assigned task before stopping, allowing for safe and effective maintenance. |
| Range | Load Capacity | Specifies the maximum load capacity of the unit (0 to 2 kg). |
| | Workspace | Defines the permissible workspace or operating range of the unit (151 mm to 1,050.5 mm). |

Table 4.1.: Summary of properties, operations, and ranges within Operational Submodel

### 4.4.3. Documentation Submodel

Technical Specification and User Manual are two Submodel Element Collections that are part of the Documentation Submodel within the Asset Administration Shell (AAS) for the UR5e robot. Due to the constraints of hosting PDF file formats on an OPC UA Server, the Documentation Submodel employs a different methodology. To enable access to documentation, it accesses external storage sites rather than embedding these files directly.

#### Technical Specification Submodel Collection

One essential resource for thorough technical details regarding the UR5e robot is the Technical Specification Collection. These attributes provide information about the robot's characteristics, facilitating maintenance-free operation and well-informed decision-making. Included are the subsequent attributes:

- **Technical Specification - German**: This property has a link that opens a German PDF file containing the UR5e robot's comprehensive technical specifications. The file can be stored elsewhere, such as on Dropbox. It provides information on technical data, operating settings, and performance indicators, enabling a thorough comprehension of the robot's capabilities.

- **Technical Specification - English**: Similar to the German version, this property provides a link to an external storage location where the PDF file containing the English translation of the UR5e robot's technical specifications can be downloaded. It ensures that this vital information is accessible to stakeholders who prefer the English language.

#### User Manual Submodel Collection

The User Manual Collection is intended to instruct users on how to operate and sustain the UR5e robot. This collection contains a property linked to the external storage location (e.g., Dropbox) where the comprehensive English user manual can be accessed. This method ensures that users have access to crucial instructions for optimal operation despite the limitations of directly embedding PDF documents.

- **User Manual - English**: This property contains a link to the external storage location where the PDF user manual for the UR5e robot can be found. It includes setup procedures, operational guidelines, troubleshooting instructions, and maintenance guidelines, augmenting user proficiency and facilitating the robot's operation.

In summary, the Documentation Submodel, The UR5e robot can be effectively understood, operated, and maintained with its Technical Specification and User Manual Collections. These compilations ensure that technical specifications and user instructions are easily accessible, facilitating well-informed decisions and optimizing the performance of robots.

### 4.4.4. CAD Submodel

The CAD Submodel within the Asset Administration Shell (AAS) is an essential repository for three distinct Submodel Collections, each of which aims to improve the comprehension and utilization of the workpiece transfer unit. These collections include the Gripper Assembly, the Setting Base Tool Assembly, and the Workpiece Transfer Unit with Digital Factory Integration components.

Each Submodel Collection contains essential properties that provide insight into the physical characteristics of the workpiece transfer unit. Due to the limitations of hosting complex file formats, such as 3D CAD files and CAD Drawing PDFs, on an OPC UA Server, the CAD Submodel employs an alternative strategy. Instead of directly embedding these files, it refers to external storage locations, facilitating quick access to visual representations and vital data. Consequently, the specific properties of each Submodel Collection are as follows:

### Gripper Assembly Submodel Collection

This collection revolves around the design and visualization of a 3D-printed gripper assembly, custom-tailored to meet project-specific requirements. It comprises the following properties:

- **CAD 3D Model Gripper Assembly**: This property links to 3D CAD files in eDrawing Web HTML format. These files vividly illustrate the 3D-printed gripper assembly, specially designed to replace the finger of the Robotiq Hand-E gripper. It provides a clear visualization of this critical component. The mentioned files can be found in Appendix A.1.1.
- **CAD Drawing Gripper Assembly**: A link to CAD drawing files in PDF format is included. These drawings offer comprehensive measurements and visuals of the 3D-printed gripper assembly, showcasing its form and functionality. Appendix A.1.2 shown in the mentioned file.

### Setting Base Tool Assembly Submodel Collection

Focused on the Setting Base Tool Assembly, this collection provides insights into the assembly process and its compatibility with the Robotiq Hand-E gripper. It encompasses the following properties:

- **CAD 3D Model Setting Base Tool Assembly**: This property links to 3D CAD files in eDrawing Web HTML format, which can be found in Appendix A.2.1, offering a detailed view of the 3D printed Setting Base Tool. It illustrates how this component can be assembled with the Robotiq Hand-E gripper to establish a new base for the UR5e robot.
- **CAD Drawing Setting Base Tool Assembly**: A link to CAD drawing files in PDF format can be found. These drawings provide precise measurements and visualizations of the Setting Base Tool Assembly, highlighting its role in reconfiguring the robot's base. The mentioned files are shown in Appendix A.2.2.

### Workpiece Transfer Unit with Digital Factory Integration Submodel Collection

This comprehensive collection addresses the integration of the Workpiece Transfer Unit within a digital factory environment. It consists of the following properties:

- **CAD 3D Model Workpiece Transfer Unit with Digital Factory**: This property links to eDrawing Web HTML-formatted 3D CAD files, which can be found in Appendix A.3.1. These files demonstrate how the Workpiece Transfer Unit can be integrated into a digital factory. They include the workspace of the unit, indicating the maximum range for transporting workpieces to various modules within the digital factory.
- **CAD Drawing Workspace for Workpiece Transfer Unit**: This property links to CAD drawing files in PDF format. These drawings outline the required workspace concerning the expected layout of the digital factory, ensuring efficient operation of the unit. Appendix A.3.2 shown the mentioned file.
- **CAD Drawing Maximum Area for Workpiece Transfer Unit**: This property links to CAD drawing files in PDF format, as shown in Appendix A.3.3. These drawings illustrate the maximum available area for implementing the defined asset within the planned digital factory, optimizing space utilization.
- **CAD Drawing Conveyor Belt**: This property includes a link to CAD drawing files in PDF format. The drawings visualize the conveyor belt within the digital factory from which the Workpiece Transfer Unit will pick up or place workpieces, enhancing the understanding of its operational environment. The mentioned files can be found in Appendix A.3.4.
- **CAD Drawing Workpiece**: A link to CAD drawing files in PDF format can be found. These drawings depict the workpieces to be transported by the Workpiece Transfer Unit, aiding in visualizing their shape and size. Appendix A.3.5 shown in the mentioned file.
- **CAD Drawing Adapter Plate**: This property links to PDF files containing CAD drawings of the adapter plate used to mount the UR5e robot to the existing robot stand in the digital factory. It also can be found in Appendix A.3.6

In conclusion, the CAD Submodel facilitates the visualization and comprehension of essential components and their integration within the workpiece transfer unit and digital factory. Providing links to external resources ensures that 3D CAD models and CAD drawings are accessible while maintaining OPC UA specification connectivity. This strategy improves the overall comprehension and administration of the physical components of the workpiece transfer unit.

### 4.4.5. Model 3D Print Submodel

The Model 3D Print Submodel within the Asset Administration Shell (AAS) is a central repository, facilitating the accessibility and seamless integration of 3D printed components essential to the workpiece transfer unit. This submodel consists of two distinct Submodel Collections: the Gripper Collection and the Setting Base Tool Collection, each designed to facilitate the maintenance and replacement of essential components.

The primary objective of this submodel is to provide maintenance stakeholders with a streamlined solution for quickly replacing or repairing critical components. The initial intent was to directly embed STL-formatted 3D model files within the Asset Administration Shell, facilitating 3D printer accessibility. However, the limitations of the OPC UA Server, which necessitate exporting the AAS to XML format, posed difficulties in terms of hosting STL files. As a result, an alternate strategy was adopted in which the actual STL files are stored externally on a platform such as Dropbox, and the AAS references links to these files within its properties. Consequently, the specific properties of each Submodel Collection are as follows:

#### Gripper Submodel Collection

The Gripper Collection focuses on the maintenance and replacement of the 3D printed gripper, specifically designed to replace the finger of the Robotiq Hand-E gripper for project-specific requirements. This collection comprises two essential properties:

- **Gripper Left 3D Model**: This property provides a link to the STL-formatted 3D model file of the left part of the gripper, as shown in Appendix A.4.1. Stakeholders can access and download this file, enabling them to swiftly 3D print a replacement in the event of damage to the existing left gripper component.
- **Gripper Right 3D Model**: Similar to the left gripper, this property offers a link to the STL-formatted 3D model file of the right part of the gripper. Users can conveniently download and utilize this file with a 3D printer to replace the right gripper component when necessary. Appendix A.4.2 shown the mentioned file.

#### Setting Base Tool Submodel Collection

The Setting Base Tool Collection addresses the maintenance and replacement requirements associated with the Setting Base Tool Assembly. This collection consists of two properties, each corresponding to a critical component:

- **Setting Base Tool Holder 3D Model**: This property includes a link to the STL-formatted 3D model file of the holder component within the Setting Base Tool Assembly, which can be found it Appendix A.5.1. Maintenance stakeholders can access and download this file, facilitating the rapid replacement of the holder component when needed.
- **Setting Base Tool Tip 3D Model**: Likewise, this property provides a link to the 3D model file in STL format for the rod component within the Setting Base Tool Assembly. In the event that the existing tip component is damaged, users can quickly obtain and use this file with a 3D printer to create a replacement. The mentioned files can be found in Appendix A.5.2.

In conclusion, the Model 3D Print Submodel, along with its Gripper and Setting Base Tool Submodel Collections, provides a standardized method for managing and preserving necessary 3D printed components within the workpiece transfer unit. By providing direct access to STL-formatted 3D model files, stakeholders can seamlessly address maintenance requirements, ensuring the continuous and efficient operation of the unit.

### 4.4.6. Views of AAS

Within the Asset Administration Shell (AAS) of the workpiece transfer unit, four distinct views accommodate the different requirements of various stakeholders: Technical Information, Operational Information, Design Information, and Maintenance Information. As depicted in Figure 4.4, each of these viewpoints serves a distinct function, enabling stakeholders to access information that is specific to their roles and responsibilities.

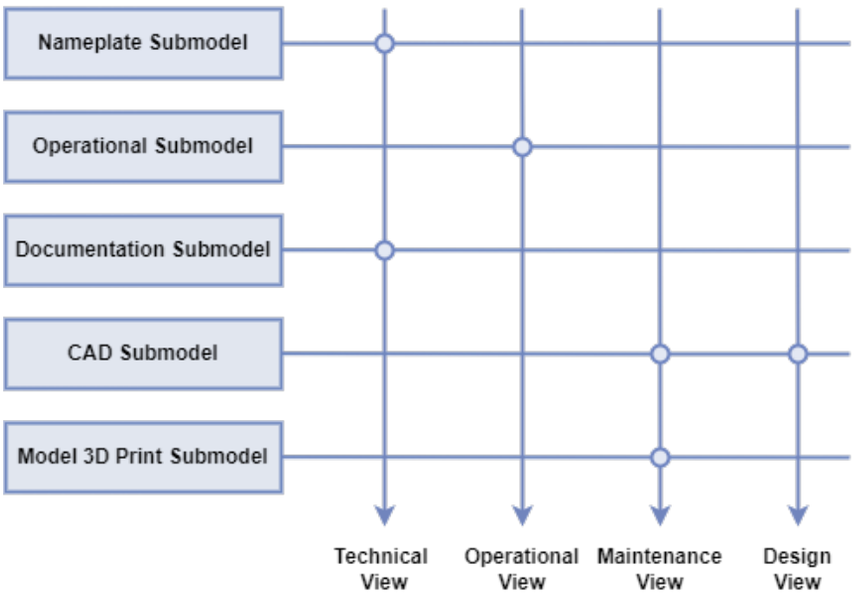



Figure 4.4.: Four Views of the AAS Workpiece Transfer Unit, own illustration

As a result, the four distinct views of the AAS provide stakeholders with precisely the information they need, enhancing efficiency and effectiveness in their respective role as follow:

#### Technical Information View

The Technical Information View delves into the technical aspects of the workpiece transfer unit, concentrating primarily on its central component, the UR5e robot. This perspective incorporates important technical documentation, including:

- **Technical Specification**: Detailed technical specifications of the UR5e robot, covering performance metrics, operating parameters, and technical data.
- **User Manual**: Comprehensive guidance for operating and maintaining the UR5e robot, including setup procedures, operational guidelines, troubleshooting, and maintenance instructions.
- **Nameplate Information**: Essential identification and classification details of the asset, such as its unique identifiers, manufacturer details, and product designations.

The Technical Information View is tailored to technical stakeholders, offering comprehensive access to the information crucial for understanding and managing the workpiece transfer unit.

#### Operational Information View

The Operational Information View is a comprehensive collection of data and functions related to the operational aspects of the workpiece transfer unit. This view is primarily designed to cater to operational stakeholders, enabling them to efficiently monitor and control the unit's functions and status. The Operational Information View encompasses the following key components, with a primary focus on the Operational Submodel:

- **Operation Range**: This section provides information about the unit's range, including load capacity and workspace dimensions. These specifications are essential for understanding the unit's operational capabilities.

- **Operation Functions**: The view includes access to two main operational functions: so called:
  - **Pick & Place Function**: This function allows stakeholders to initiate pick-and-place operations, specifying details such as pick IDs, pick directions, place IDs, and place directions.
  - **Service Function**: The service function empowers integrators to command the workpiece transfer unit to navigate to specific service locations for maintenance or service tasks. It also provides information on the unit's service status.

- **Operation Properties**: This section offers insights into the unit's operational status including:
  - **Occupancy Status**: Indicates whether the unit is busy or idle, optimizing production procedures and resource allocation.
  - **Maintenance Status**: Specifies whether or not the unit is currently undergoing maintenance, including its digital factory location.
  - **Operation Status**: Tracks the current operation of the unit, including picking and placing positions and module IDs.
  - **Number of Queues**: Provides information on the number of tasks or operations currently in the unit's queue, assisting in scheduling and planning production tasks.

The Operational Information View ensures that stakeholders involved in the operational aspects of the unit can efficiently access and utilize the information and functions necessary for its effective control and management.

### Design Information View

The primary purpose of the Design Information View is to provide stakeholders with access to critical CAD models and related information regarding the workpiece transfer unit and its components. It focuses on the design and assembly aspects, allowing users to investigate the unit's structure in complete detail. This view provides:

- **3D CAD Models**: Access to 3D CAD drawings of various components, including the workpiece transfer unit within the digital factory environment. These CAD models provide highly detailed visualizations and precise numerical measurements, enabling designers, users, and maintenance personnel to gain valuable insights into the unit's design.

- **Numerical Measurements**: The CAD drawings contain numerical measurements of assembly equipment, offering comprehensive information about the unit's components. These measurements are instrumental in understanding the unit's physical attributes and compatibility with other equipment.

- **Assembly Visualizations**: Detailed CAD drawings provide visual representations of how components are assembled, offering insights into the unit's assembly process. Maintenance personnel and assemblers can utilize these visualizations to understand the correct arrangement and integration of components for efficient assembly.

- **Comprehensive Design Data**: The Design Information View is a repository of comprehensive design data, empowering stakeholders involved in design, assembly, and maintenance to make informed decisions and optimize the unit's performance.

In essence, the Design Information View focuses on providing access to CAD models and related design data, enabling stakeholders to gain a deep understanding of the workpiece transfer unit's structure and design intricacies.

#### Maintenance Information View

The Maintenance Information View is intended for maintenance stakeholders who require access to information regarding maintenance tasks. The Maintenance Information View derives its content primarily from the Model 3D Print Submodel and specific parts of the CAD Submodel, such as the Gripper Assembly Submodel Collection and the Setting Base Tool Assembly Submodel Collection. In this view, stakeholders can locate STL 3D files for various components, allowing for simple replacement in the event of damage. CAD drawings also provide assembly visualization for incorporating newly printed components into an existing unit. Therefore, the Maintenance Information View ensures that maintenance personnel can efficiently access and utilize the information necessary for the workpiece transfer unit's servicing.

## 4.5. Software Architecture Design

In this section, the software architecture design for this project shall be delved, building upon the foundation laid during the previous summer semester. The primary goal of this phase is to align the workpiece transfer unit with Industry 4.0 principles by implementing the Asset Administration Shell (AAS) framework and utilizing OPC UA specifications. Two key aspects shall be clarified:

### 4.5.1. Software Component of Defined Asset within AAS Framework

As detailed in Subsection 4.1.2 - Software Component, the project already includes an essential component: the UR5e OPC UA Server. This server acts as the interface connecting the UR5e robot with the broader digital ecosystem, facilitating the transfer of variables and operational data.

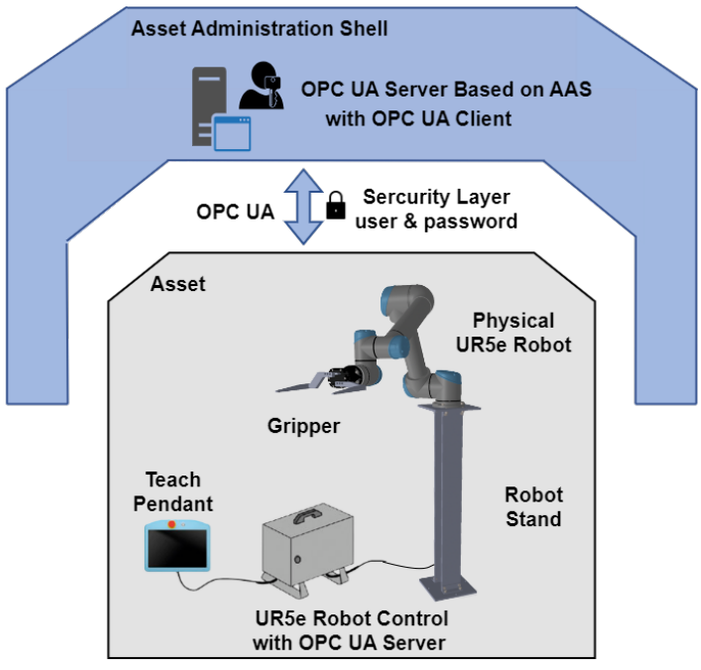



Figure 4.5.: Software Component of Defined Asset within AAS Framework, own illustration

Hence, to enhance this software architecture according to the project's requirement, a new component is introduced: an OPC UA server based on the structure defined in Section 4.4 - AAS Structure Specification for the Defined Asset. This OPC UA server is developed to ensure seamless communication and integration with the UR5e OPC UA Server through the existing OPC UA Client from the workpiece transfer unit winter semester project, as illustrated above in Figure 4.5.

Security is a further consideration for this implementation. A robust security layer is implemented to ensure that the OPC UA Client and UR5e OPC UA Server communicate securely. This layer restricts access to the defined asset and allows only authorized users to send variable robot commands. Utilizing a username and password, these credentials are configured in the UR5e OPC UA Server and the OPC UA Client. This dual authentication safeguards the system by ensuring only authorized users can access the UR5e OPC UA Server.

In summary, the software component of the defined asset within the AAS framework consists of the following key elements:

1. **OPC UA Based on AAS:** This server is responsible for receiving variables from clients or users, which can be other modules in the digital factory or users. The server then passes these variables to the OPC UA Client.
2. **OPC UA Client (from the workpiece transfer unit winter semester project):** This client communicates with the UR5e OPC UA Server via a Python program using the opcua-asyncio library. The OPC UA uses functions provided by the client Based on AAS to send variables to the UR5e OPC UA Server.
3. **UR5e OPC UA Server:** This server is responsible for receiving variables and transferring them to the robot control's URP (Universal Robot Program). The URP processes the variables to execute the required operations.

### 4.5.2. Overall Communication Architecture within the Digital Factory

The I4.0 workpiece transfer unit is just one module integrated within the broader digital factory environment. Therefore, additional modules or clients within the communication architecture must be considered to ensure seamless operations.

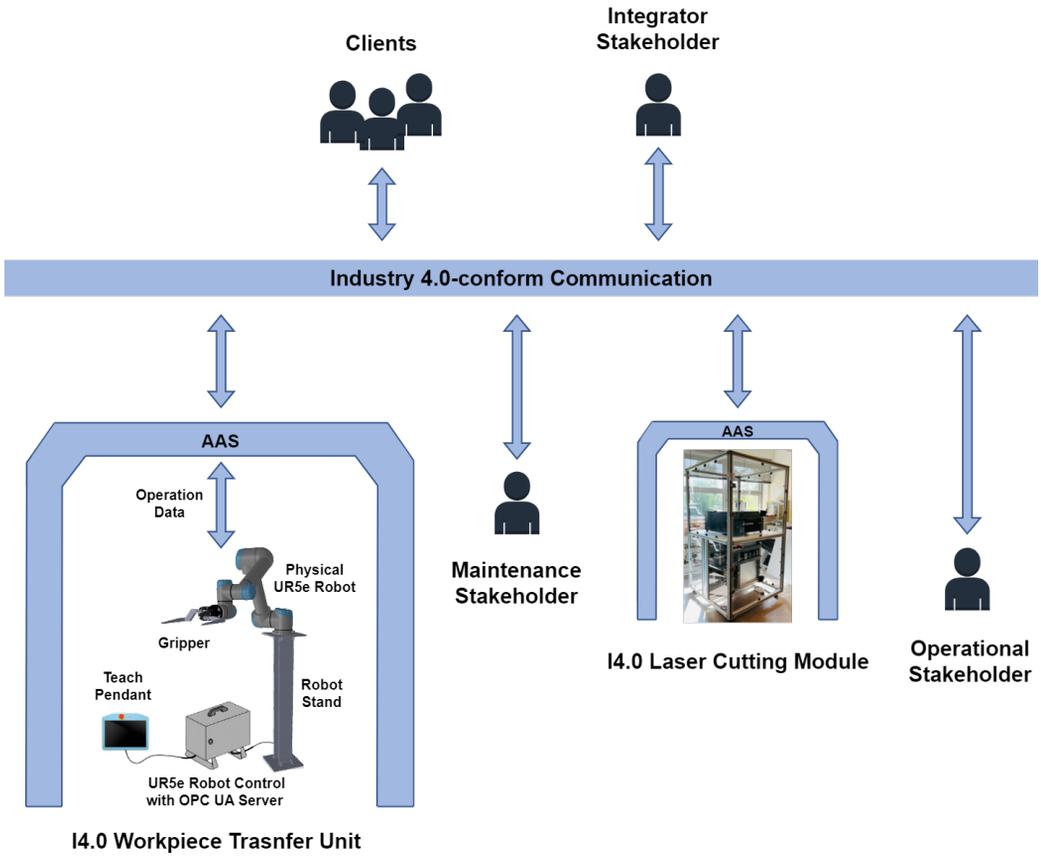



Figure 4.6.: Overall Communication Architecture within the Digital Factory, own illustration

Figure 4.6 illustrates the overall communication architecture within the digital factory. It showcases how various modules and clients, including the I4.0 workpiece transfer unit, interact harmoniously within this architecture. This collaborative approach ensures that all stakeholders can access the necessary data and functionalities to optimize their operations within the digital factory.

For instance, operational stakeholders have a role in monitoring and controlling the I4.0 workpiece transfer unit. They rely on the system to provide status and access to operational control. These stakeholders subscribe to the OPC UA server, allowing them to receive updates and access relevant information. This empowers stakeholders to make well-informed decisions based on the robot's status and operational data.

A second instance relates to maintenance stakeholders, who are essential in assuring the digital factory's smooth operation. These professionals have specific requirements for optimizing maintenance processes. Utilizing I4.0 workpiece transfer unit maintenance data, maintenance personnel optimize their duties within this architecture. This is possible via a subscription to the OPC UA Server Based on AAS. When maintenance is required, they can swiftly retrieve the required data directly from the server, facilitating precise and efficient maintenance procedures.

In conclusion, the software architecture design integrates not only the AAS framework for efficient data management but also the digital factory's broader communication architecture. This comprehensive strategy facilitates seamless integration and empowers stakeholders to maximize the unit's full potential.

# 5. Implementation

The implementation phase is where the detailed design, developed in the previous phase, is transformed into a tangible and functional product. This chapter outlines the key tasks and activities carried out during this phase, leading to the realization of the Industry 4.0 workpiece transfer unit. The implementation phase encompasses several crucial tasks, each contributing to the successful development and deployment of the unit.

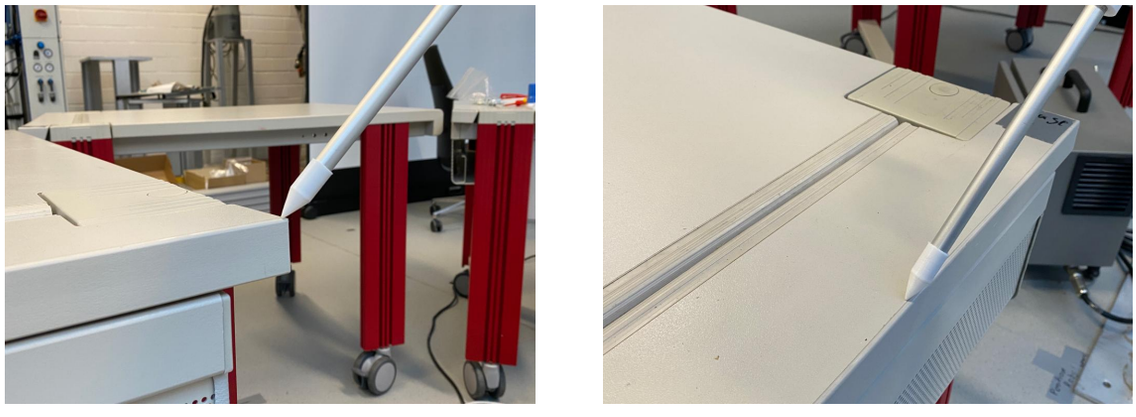
## 5.1. Configuration of UR5e Robot Programming

In order to perform precise and dynamic manipulations in all six dimensions, the Universal Robot UR5e was updated with new programming for this mission. This encompassed both linear and rotational motions, ensuring that the workpiece transfer unit possessed the operational flexibility required for seamless collaboration with adjacent modules within the digital factory ecosystem. The configuration involved adding variables to the robot's program to depict the robot's status and aligning it with the project's specifications. To attain this configuration, however, the following tasks were performed:

1. **Updated the Installation Files of the UR5e:** The installation files of the UR5e were updated to enable operation in all six directions.
2. **Modified the Configuration Files (INI Files):** The configuration files (INI files) were modified to define variables specifically tailored to this project's robot programming requirements.
3. **Enhanced the Existing URScript Program (URP Program):** The existing URScript program (URP program) was enhanced to align with the project's unique robot programming needs.

### 5.1.1. Updated the Installation Files of the UR5e

During this phase, the UR5e's installation files were modified to facilitate operation in all six dimensions, including precise linear and rotational motion. Notably, detailed instructions for configuring the Tool Center Point and Plane parameters can be found in the previous project report for the winter semester under Chapter 5 - Implementation, Section 5.2 - Software Implementation, Subsection - 5.2.1 Robot Programming, Subsubsection - Robot Configuration. This section provides only a summary of the modifications made to the installation files to ensure compliance with the project's specific requirements [NM23].

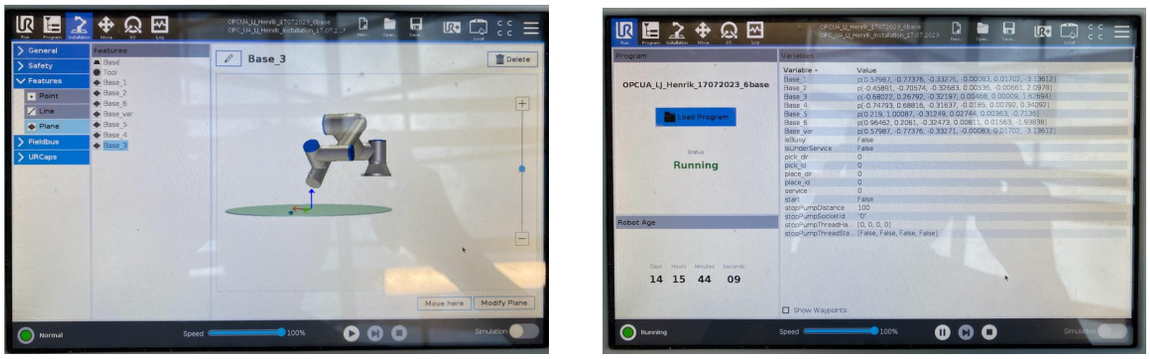


(a) Configured the origin of the plane

(b) Established the positive X and Y directions of the plane

Figure 5.1.: Setting a plane using the base tools, own illustration

Moving forward, Figure 5.1 above offers a visual representation of the implementation process, which entailed the addition of new planes into the installation file. It is important to note that in the previous winter project, only three base planes were available, referred to as Base1, Base2, and Base6. To introduce the new planes, the setting base tools were equipped onto the gripper and utilized as TCPs (Tool Center Points) for setting these additional planes. Figure 5.1a demonstrates how the TCP of the setting base tool accurately positions the plane's origin, ensuring precision and alignment with project specifications. Meanwhile, Figure 5.1b showcases the process by which the setting base tool establishes the positive X and Y directions of the plane defined a new plane using right-hand rules.




(a) Graphic representation of the plane for all Base

(b) The coordinate value of the plane

Figure 5.2.: Summarise of added planes in robot control, own illustration

After the additional planes were successfully configured, Figure 5.2 provides a visual overview of the robot control interface, displaying all the planes added to the installation file. This representation offers a dual perspective, comprising graphical depictions and the corresponding coordinate values presented as variables. Figure 5.2a presents a visual list of graphical representations of the planes for all Bases, including Base1 to Base6, which were seamlessly integrated into the robot's control. These planes are visually juxtaposed with the original bases of the robot and its tool, providing a comprehensive visual reference.

Additionally, Figure 5.2b depicts that the 'Run' tab complements the visual representation by displaying the coordinate values of these added planes. These values are represented as pose variables within the installation file. A pose variable is a vector that describes the location and orientation in Cartesian space. It combines a position vector (x, y, z) and a rotation vector (rx, ry, rz) representing the orientation, denoted as p[x, y, z, rx, ry, rz]. The table 5.1 below summarizes all the pose variables for the configured planes:

| | Position Vector | | | Rotation Vector | | |
|---|---|---|---|---|---|---|
| Base | x | y | z | rx | ry | rz |
| Base_1 | 0.57987 | -0.77376 | -0.33271 | -0.00083 | 0.01702 | -3.13612 |
| Base_2 | -0.45891 | -0.70574 | -0.32683 | 0.00536 | -0.00661 | 2.0973 |
| Base_3 | -0.76191 | 0.45255 | -0.3194 | 0.00889 | 0.00245 | 0.7004 |
| Base_4 | -0.74793 | 0.68816 | -0.31637 | -0.0185 | 0.00792 | 0.34092 |
| Base_5 | 0.219 | 1.00087 | -0.31249 | 0.02744 | 0.00363 | -0.7136 |
| Base_6 | 0.96462 | 0.96462 | -0.32473 | 0.00811 | 0.01563 | -1.93838 |

Table 5.1.: Position and Rotation Vector of Added Base Plane

### 5.1.2. Modifications to the Configuration Files

It is appropriate to mention that the detailed instructions for altering initialization files have been extensively covered in the previous winter semester project report. Interested readers can refer to Chapter 5 - Implementation, Section 5.2 - Software Implementation, Subsection - 5.2.1 Robot Programming, Subsubsection - OPC UA Robot (Robot Control) of the previous report for an in-depth guide. In this section, a brief overview of the activities involved in updating the Configuration Files is provided, along with insights into the rationale for these modifications [NM23].

The configuration files (.ini files) for the existing workpiece transfer unit, as developed during the previous winter semester, exposed a limited set of variables:

```
start;bool;false
isBusy;bool;false
isUnderservice;bool;false
service;int;0
pick_id;int;0
pick_dir;int;0
place_id;int;0
place_dir;int;0
```

However, as the unit now aspires to evolve into an Industry 4.0 (I4.0) compliant workpiece transfer unit, it must align with the current project's requirements. Therefore, a crucial step in this evolution is the updating of the Configuration Files to meet these new demands. The modified configuration files now include the following changes:

```
start;bool;false
isBusy;bool;false
isUnderService;bool;false
atServicePosition;int;0
service;int;0
pick_id;int;0
pick_dir;int;0
place_id;int;0
place_dir;int;0
```

Notable in this update is the addition of the integer variable 'atServicePosition,' which is initially initialized to 0. This variable indicates the unit's service position status, an important aspect for maintenance purposes. It is essential to note that these modifications are just the beginning, as additional enhancements are planned. The upcoming task, "Enhancement of the Existing URScript Program (URP Program)," will supplement these modifications in later phases.

### 5.1.3. Enhanced the Existing URScript Program (URP Program)

The existing URScript program, originally designed for the workpiece transfer unit, has undergone substantial enhancements to align with evolving project requirements. This section offers an overview of the key modifications and improvements implemented within the program. Initially, the program functioned as the control logic for the workpiece transfer unit, managing tasks such as workpiece handling and unit servicing. However, in response to the unit's transformation into an Industry 4.0 workpiece transfer unit, several critical updates were necessary. The robot programvis presented in Appendix B.1, providing a comprehensive view of the program's evolution and enhanced capabilities.

In the subsequent sections, the changes made to the URScript program will be delved into, detailing the additions and alterations that enable seamless integration into the Industry 4.0 framework.

1. **Additional OPC UA Variables:** To enhance communication and data exchange with other components in the digital factory ecosystem, several new OPC UA variables were incorporated into the program. These variables convey the unit's status, service needs, and operational state. The newly added variables include:
    - **isUnderService**: This boolean variable indicates whether the unit is currently under service or not. It serves as a crucial indicator of the unit's maintenance status.
    - **atServicePosition**: An integer variable representing the specific service position of the unit. This variable helps in precisely tracking the unit's location during service operations.
2. **Improved Robot Control:** The program's robot control logic was significantly improved to enhance its versatility and adaptability. Specifically, the following enhancements were introduced:
    - **Six Base Positions:** The program was updated to accommodate six distinct base positions, denoted as `Base_1_const` through `Base_6_const`. These base positions enable precise robot positioning for different operational scenarios, adding flexibility to the unit's movements.
    - **Service Pose Tracking:** The introduction of the `ServicePose` variable allows for the recording of the robot's pose during service operations. This tracking capability is valuable for maintaining accurate records of the robot's positions during maintenance tasks.
3. **Enhanced User Interaction:** The program now includes features that enhance user interaction and provide valuable feedback regarding the unit's status. This includes:
    - **isUnderService Indicator:** The `isUnderService` variable provides a real-time indication of whether the unit is currently under maintenance or in regular operation mode.
    - **Service Position Feedback:** The `atServicePosition` variable offers insights into the unit's exact location during service tasks, aiding in coordinating maintenance activities.
4. **Expanded Service Cases:** The program's logic was extended to include additional service cases, covering service positions 3, 4, and 5. This expansion allows a broader range of service scenarios to be handled effectively.
5. **Comments and Documentation:** To ensure clarity and facilitate future maintenance and development, extensive comments and explanatory text have been added throughout the program. These comments provide insights into the program's logic, making it more accessible to both current and future users.
6. **Robust Error Handling:** The program now incorporates robust error-handling mechanisms to address scenarios where unexpected or invalid values are received from OPC UA variables. This ensures that the program maintains stable performance across various operational scenarios.

In conclusion, the comprehensive enhancements to the existing URScript program have significantly improved its adaptability and functionality. These modifications have empowered the workpiece transfer unit to align with the requirements of an Industry 4.0 environment. With improved communication, expanded service capabilities, and enhanced user interaction, the program stands ready to contribute to the unit's increased efficiency and effectiveness in the digital factory ecosystem. For a detailed explanation of the program's flowchart and structure, interested readers can refer to Chapter 5 - Implementation, Section 5.2 - Software Implementation, Subsection - 5.2.1 Robot Programming, Subsubsection - Robot Program in the previous winter semester project report [NM23].

## 5.2. Creation of Asset Administration Shell for Defined Asset

Development of the Asset Administration Shell (AAS) tailored to the workpiece transfer unit for Industry 4.0 is an essential stage. This task entails transforming the AAS blueprint developed in the preceding phase into a digital representation that incorporates the unit's data, interfaces, and capabilities. Within the digital factory, the AAS is the foundation for efficient data administration and communication.

This section explains the practical implementation of constructing the Asset Administration Shell (AAS) for the defined asset, the workpiece transfer unit featuring the Universal Robot UR5e. The AAS is an essential part of the digital factory ecosystem, allowing for seamless data integration, administration, and interoperability.

### 5.2.1. Creating the Asset in Asset Administration Shell Package Explorer

The initial phase involves the creation of the Asset within the Asset Administration Shell Package Explorer (AASX). In this step, key attributes are defined, providing referable details that include the asset's **idShort**, **category**, and **description**. Additionally, the **idType** and **id** for identifiability are specified. Finally, the **kind** of asset is determined.

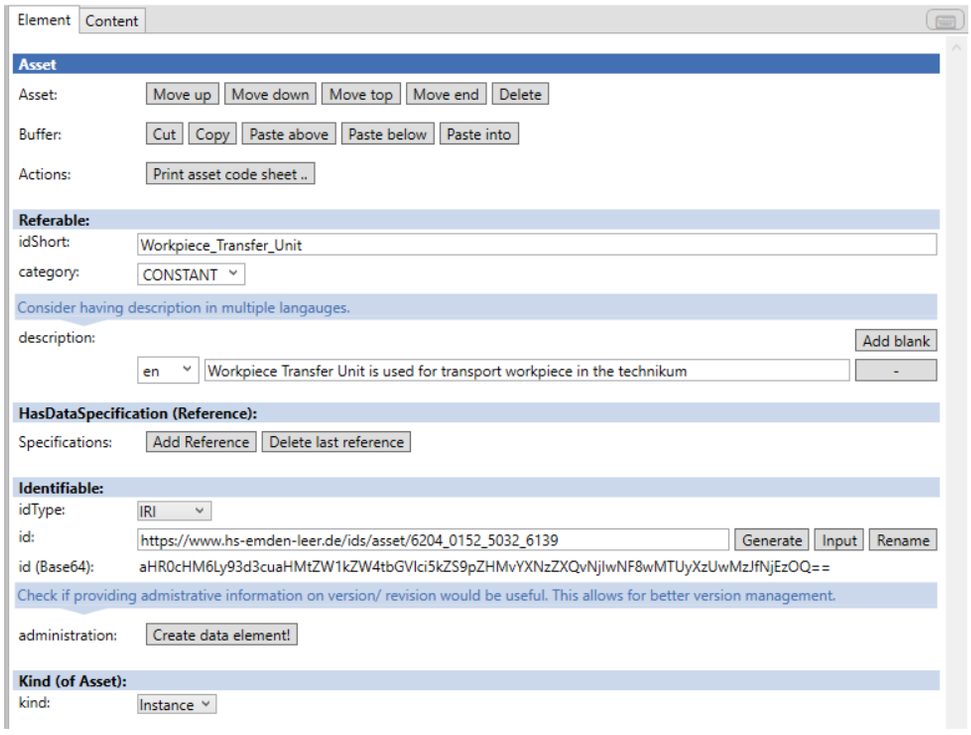



Figure 5.3.: Creation of Asset in AASX, own illustration

Figure 5.3 above depicts a screenshot illustrating the asset creation process within the AASX.

### 5.2.2. Developing the Asset Administration Shell

Following the creation of the asset, the next phase involves the development of its corresponding Asset Administration Shell (AAS). This comprehensive administration shell serves as a digital representation of the asset, encapsulating essential data, interfaces, and capabilities. It adheres to a structured model that defines the organization and composition of the asset's attributes and functionalities.

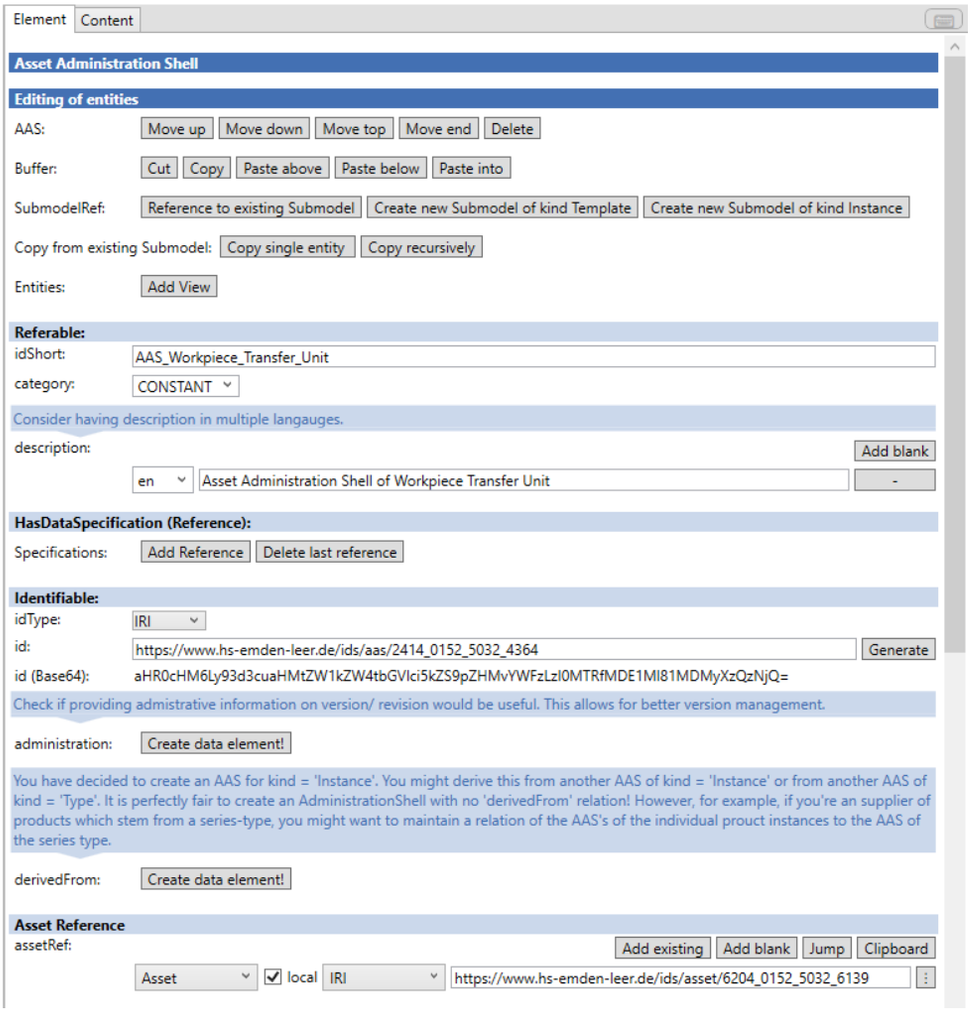



Figure 5.4.: Developing the Asset Administration Shell, own illustration

In this stage, similar to the asset creation process, key information, including referable details and identifiability, is provided to the Asset Administration Shell. Notably, a critical step is the establishment of a **Asset Reference** to the created asset within the Asset Administration Shell. This linkage ensures that the Asset Administration Shell and the asset are interconnected, allowing for seamless integration and management of the asset's data and interactions within the digital factory ecosystem. Hence, figure 5.4 above depicts a screenshot illustrating the asset administration shell development process within the AASX.

### 5.2.3. Submodels Within the AAS

The AAS for the defined asset is structured into several submodels, each designed to encapsulate specific aspects of the asset's behavior, attributes, and interactions. These submodels provide a comprehensive representation of the asset's characteristics and contribute to its digital twin within the digital environment. Below, a summary of the key information provided in each submodel is presented, with detailed information in section 4.4 AAS Structure Specification for the Defined Asset.

For this project, it is important to note that in all submodels, the Semantic ID concept is fundamental. This Semantic ID, represented by the Concept Description (CD), guarantees the identification and semantic representation of various submodel components. This method improves the asset's interoperability and comprehension within the ecosystem of the digital factory.

1. **Nameplate Submodel**

   The Nameplate Submodel has been made to serve as the repository for crucial identification and classification information of the asset. This submodel encompasses essential attributes such as the asset's name, manufacturer details, serial number, and relevant certificates. It acts as the cornerstone for comprehending the fundamental attributes of the asset. Detailed information regarding these attributes can be found in Subsection 4.4.1 - Nameplate Submodel. For a broader context regarding the asset's structure and its various submodels, please consult Chapter 4 - Detailed Design, Section 4.4 - AAS Structure Specification for the Defined Asset.

   The development of the Nameplate Submodel involved using a template provided by the Industrial Digital Twin Association (IDTA). The IDTA template, available for download from their website, offers a standardized and widely recognized format for representing asset identification and classification data within the AAS framework [eVar].

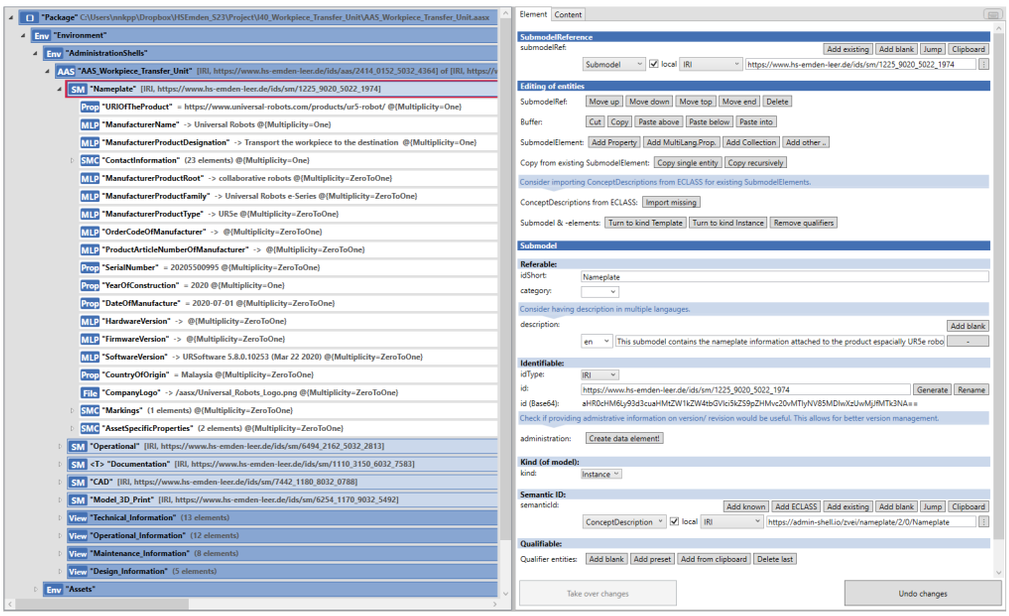



Figure 5.5.: Development of Nameplate Submodel in AASX, own illustration

Figure 5.5: A screenshot depicting the development process of the Nameplate Submodel within the Asset Administration Shell Package Explorer (AASX).

2. **The Operational Submodel**

   The Operational Submodel has been developed to function as a dynamic representation of the asset's real-time operational status, offering valuable insights into its condition and availability. Within this submodel, data concerning occupancy status, maintenance status, ongoing operations, and user accessibility through methods like 'pick & place' and 'service operation' are comprehensively documented. For a more detailed breakdown of the attributes and data contained within the Operational Submodel, please refer to Subsection 4.4.2 - Operational Submodel in Chapter 4 - Detailed Design.

   Hence, Figure 5.6 below depicts visual representation of the Operational Submodel's development process within the Asset Administration Shell Package Explorer (AASX).

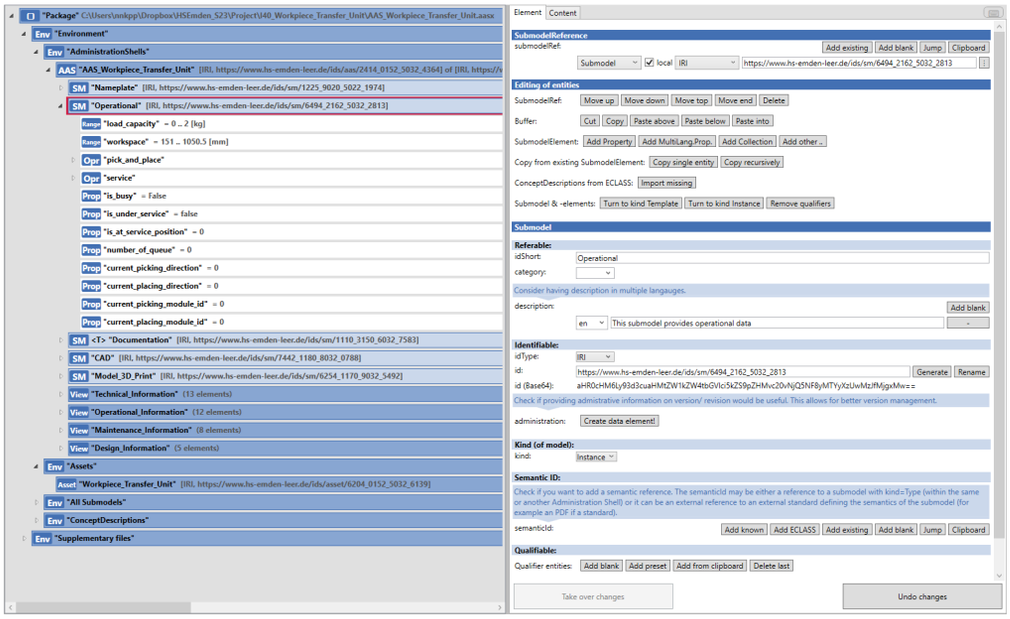


Figure 5.6.: Development of Operation Submodel in AASX, own illustration

3. **Documentation Submodel**

The Documentation Submodel has been developed to centralize critical UR5e robot documents, facilitating informed decision-making, streamlined maintenance, and lifecycle management within the digital factory ecosystem. This implementation ensures accessible information for users and maintenance personnel, improving operational efficiency and integration. For a comprehensive breakdown of the attributes and content contained within the Documentation Submodel, please refer to Subsection 4.4.3 - Documentation Submodel in Chapter 4 - Detailed Design.

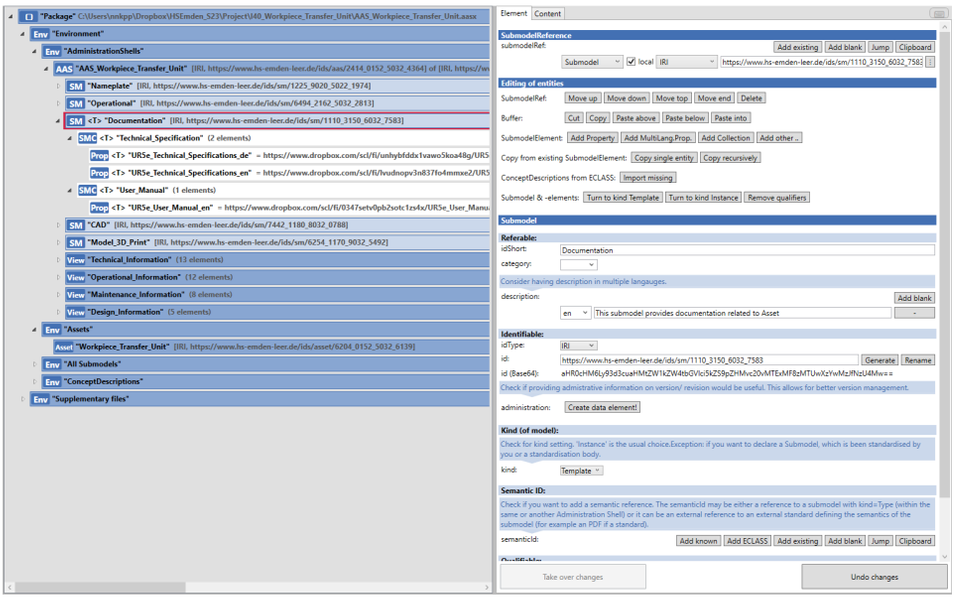


Figure 5.7.: Development of Documentation Submodel in AASX, own illustration

As shown in Figure 5.7, it is important to highlight that this Documentation Submodel is designated as a "Template," underscoring its standardized information. This designation ensures that the contained documentation can be universally applied to any UR5e robot, regardless of location or configuration.

4. **CAD Submodel Implementation**

   The CAD Submodel has been created to provide the required CAD information for the UR5e robot's components, including the Gripper, Setting Base Tools, and Workpiece Transfer Unit in the digital factory. This information provides access to 3D CAD models and CAD drawings, enabling detailed visualizations of these components. The CAD Submodel significantly enhances the UR5e robot's digital representation, supporting various tasks such as design validation, workspace planning, and collaborative decision-making within the digital factory ecosystem.

   For an in-depth breakdown of the attributes and content contained within the CAD Submodel, please refer to Subsection 4.4.4 - CAD Submodel in Chapter 4 - Detailed Design.

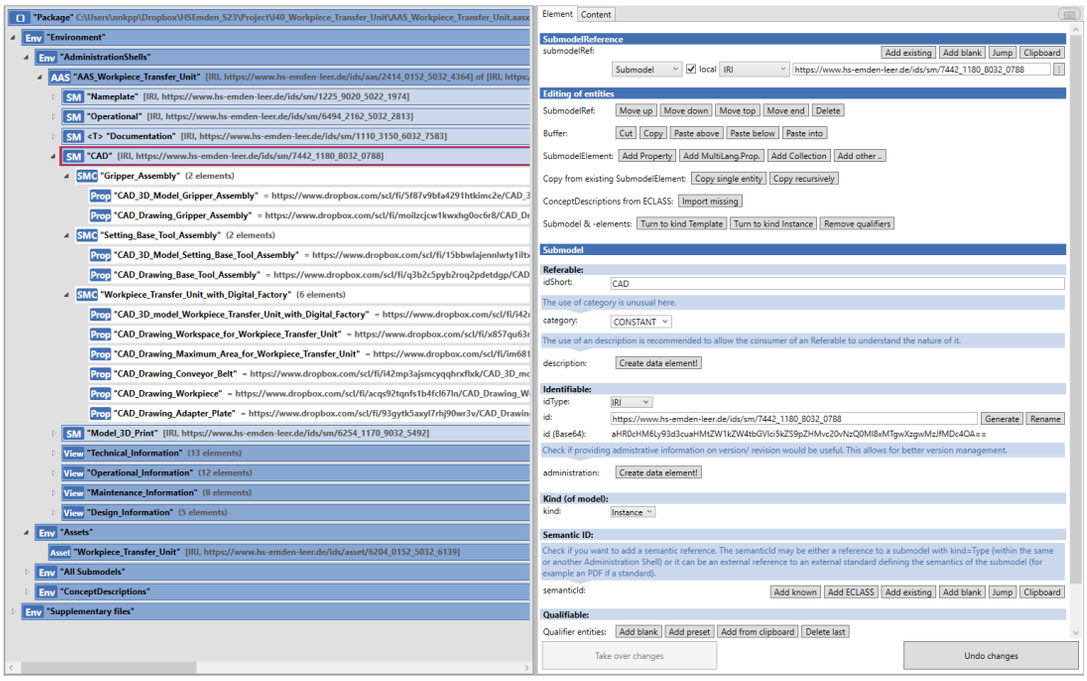



Figure 5.8.: Development of CAD Submodel in AASX, own illustration

Figure 5.8 provides a visual representation of the CAD Submodel's development process within the Asset Administration Shell Package Explorer (AASX).

5. **Model 3D Print Submodel Implementation**

   This submodel has been developed to serve as a centralized repository of 3D model files essential for critical tools and components used within the digital representation of the UR5e robot. The primary aim is to provide quick and straightforward accessibility to these 3D model files, ensuring that maintenance personnel can efficiently obtain the necessary resources for any required maintenance tasks. Within the Model 3D Print Submodel, an organized collection of access to 3D model files representing various tools and components associated with the workpiece transfer unit has been presented.

   One of the key advantages of this submodel is its potential to reduce downtime in the digital factory environment significantly. When maintenance is required for any part of the workpiece transfer unit, maintenance personnel can promptly access the relevant 3D model file from the submodel. Hence, For a comprehensive breakdown of the attributes and content contained within the Model 3D Print Submodel, please refer to Subsection 4.4.5 - Model 3D Print Submodel in Chapter 4 - Detailed Design.

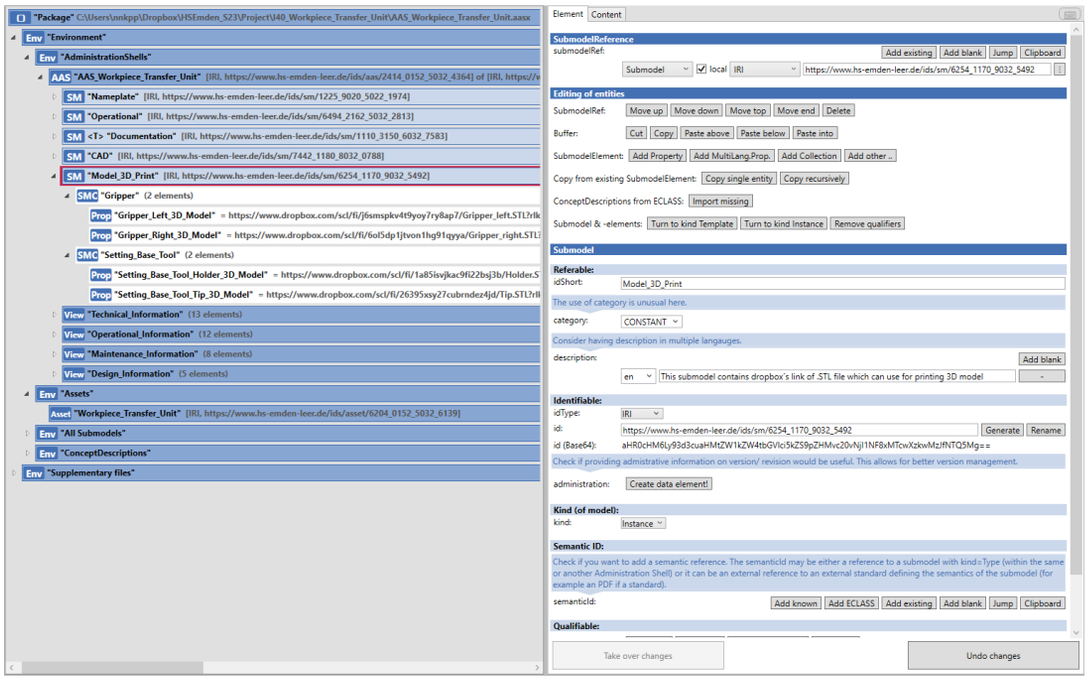



Figure 5.9.: Development of Model 3D Print Submodel in AASX, own illustration

However, Figure 5.9 offers a visual representation of the Model 3D Print Submodel's development process within the Asset Administration Shell Package Explorer (AASX).

### 5.2.4. Exporting the AAS to XML Format

The assembled Asset Administration Shell (AAS) can be exported to an XML file after development. This XML file is a structured representation of the AAS, making it compatible with various systems and programs.

To initiate the export process, follow these steps:

1. Open the Asset Administration Shell Package Explorer (AASX).
2. Open the .AASX file that contains the AAS which is to be exported.
3. Navigate to the "File" menu at the top of the interface.
4. From the dropdown menu, select "Export."
5. In the subsequent menu, opt for "Export OPC UA Nodeset2.xml (via UA Server plug-in)." This option is essential for creating a nodeset compatible with the OPC UA Server.

As a consequence of this procedure, an XML file representing the Asset Administration Shell (AAS) is generated. This XML file functions as a structured digital twin of the physical asset, encapsulating all the critical data, attributes, and interfaces thoroughly defined during the creation of the AAS. In essence, this XML file becomes a comprehensive digital representation of the asset, modeling its physical counterpart in the digital environment.

Figure 5.10 provides an illustrative step-by-step guide to the above-described export process for those who prefer visual direction. This illustration illustrates each step, making it simpler to follow the process.

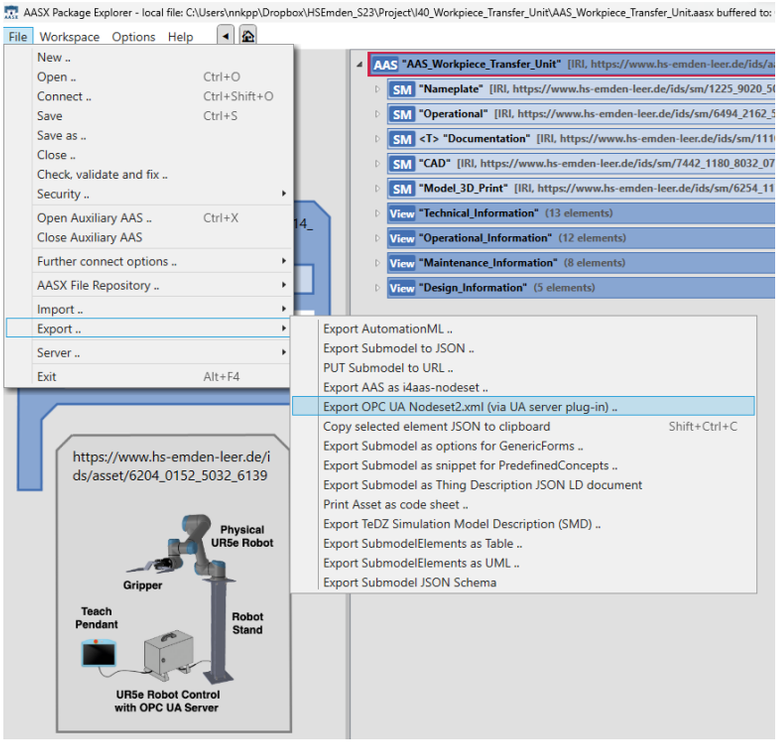



Figure 5.10.: Exporting the AAS to XML Format Process, own illustration

As depicted in Figure 5.11, the interface will promptly display the result upon completion of the exporting procedure. This interface confirmation signifies that the nodeset has been successfully created and is now available for integration with the OPC UA Server. Connecting the digital twin represented by the AAS to the physical Workpiece Transfer Unit, the nodeset facilitates robust communication and data exchange within the digital industrial environment.


Figure 5.11.: Result of Exporting Process, own illustration

Practically, this exported XML file, along with the nodeset, is essential for establishing a strong connection between the AAS-embodied digital twin and the physical asset. This connection supports a dynamic and interconnected ecosystem within the digital factory, allowing for efficient operations, data exchange, and decisions.

## 5.3. Creation of OPC UA Server - Aligned with AAS

Developing an OPC UA server compatible with the Asset Administration Shell (AAS) is an essential step. This server is a communication bridge for the workpiece transfer unit, facilitating the secure and standardized data exchange with other components within the digital factory, enhancing interoperability.

### 5.3.1. Integration of XML Data

Now that the XML file has been generated, it is necessary to integrate it into the pre-existing code from the previous winter semester project, which can be found in Chapter 5 - Implementation, Section 5.2 - Software Implementation, Subsection - 5.2.2 OPC UA, Subsubsection OPC UA Server (PC). Hence, the XML file will be imported into the code, and its nodes will be utilized within it. In order to streamline this procedure, the opcua-modeler tool shall be employed to analyze the nodes produced by the AASX.

#### Accessing OPC UA Modeler

In order to access the OPC UA Modeler program, users who have performed the prior installation can easily open it from their terminal. Open any terminal program and type "opcua-modeler," and the program will automatically launch, as shown in Figure 5.12.

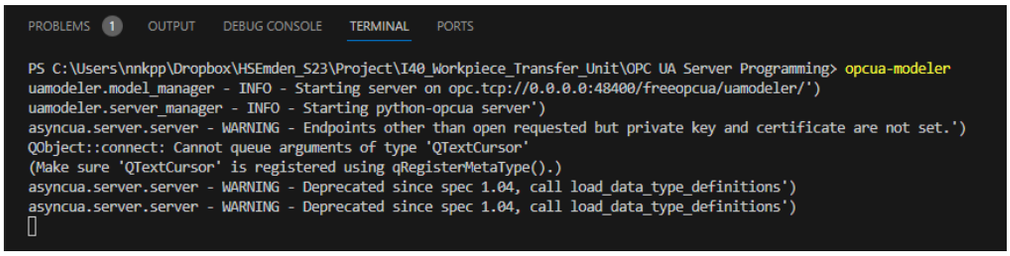



Figure 5.12.: Instructions for Opening FreeOPCUa Modeler in Terminal, own illustration

To begin, open the FreeOPCUA Modeler and load the AAS XML file. The tool will display the nodeIDs associated with the AAS. These nodeIDs are of utmost importance as they will be used in the subsequent stages of integration. To provide a visual reference, Figure 5.13 illustrates the user interface of the OPC UA Modeler, where these essential nodeIDs can be located.

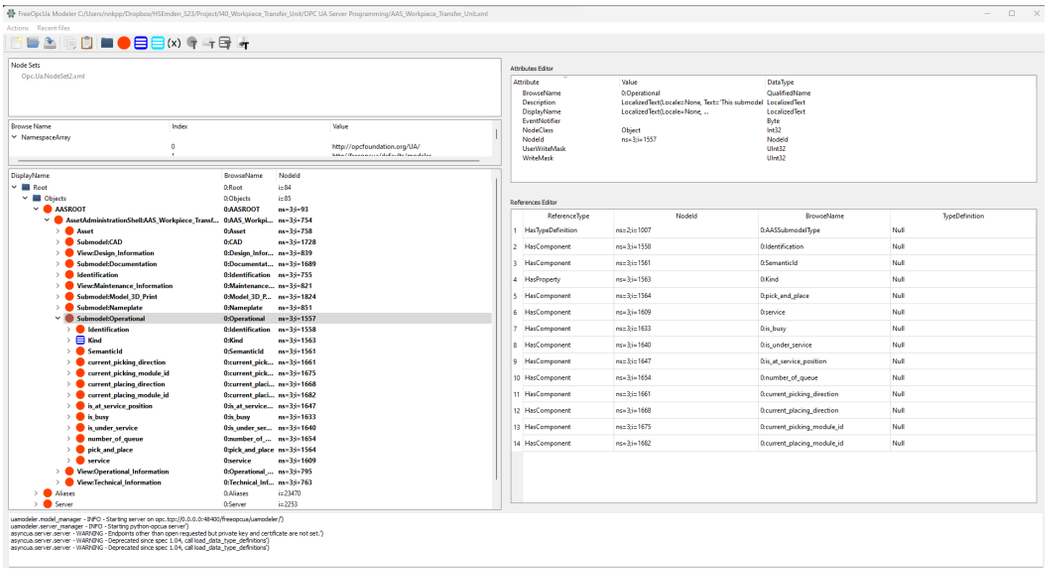
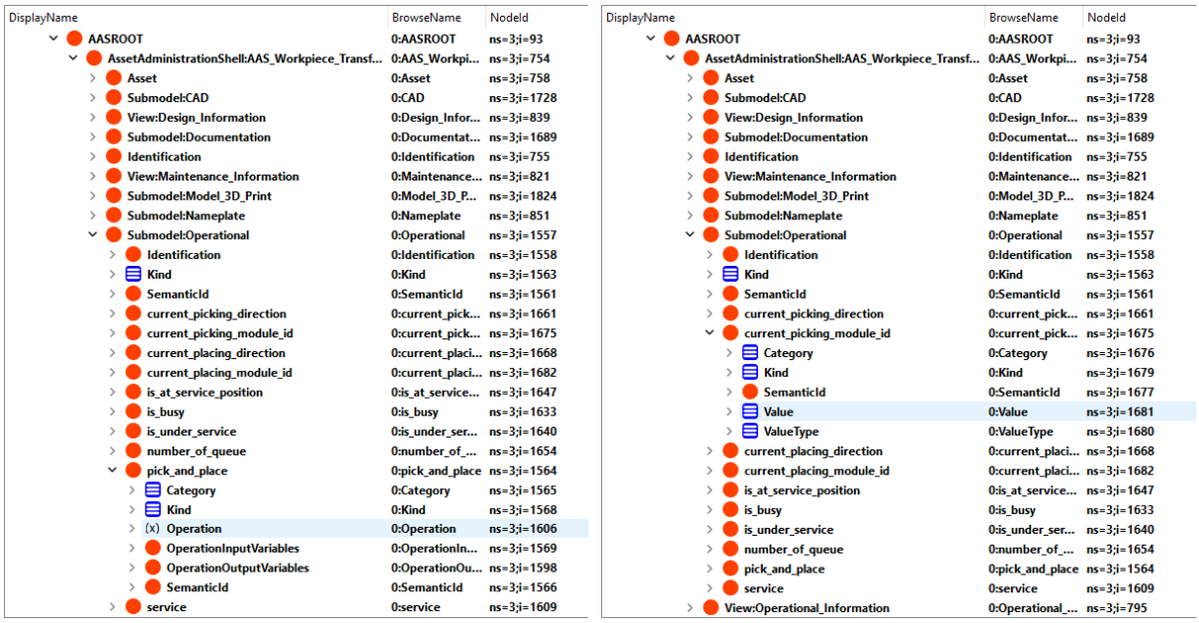


Figure 5.13.: Interface of FreeOPCUa Modeler, own illustration

### Identifying NodeIDs

By inspecting the "Operational" Submodel in FreeOPCUA Modeler, the nodeIDs of methods and values in the XML can be discovered. These nodeIDs are used to connect Python to the existing OPC UA Server.




(a) NodeID of the Operation Method

(b) NodeID of the Operation Value

Figure 5.14.: Locating NodeIDs in the FreeOPCUa Modeler, own illustration

## 5.3.2. Integration with Existing Code

Integration of nodeIDs into the existing Python code of the OPC UA Server is the next stage, following the identification of nodeIDs. This coding task will be modified to conform to the I4.0 Workpiece Transfer Unit's requirements. This coding adaptation is an important part of the project, as it bridges the gap between the AAS-defined information and the monitored operations information of the Workpiece Transfer Unit, allowing for efficient data exchange and communication within the digital factory ecosystem.

Therefore, to map these AAS components to the OPC UA Server, the process involves translating the identified nodeIDs into functional components within the OPC UA Server, including steps as follows:

### Mapping AAS to OPC UA Server

After successfully loading the AAS XML file into the OPC UA Modeler, the following step is to map these AAS components to the OPC UA Server. This procedure entails translating identified nodeIDs into functional OPC UA Server components as follows:

1. **Importing AAS XML:** The procedure begins with importing the AAS XML file, which produces a collection of nodeIDs, as mentioned previously.

```
# Import the AAS of the Workpiece Transfer Unit in XML format
aas_nodes = await server.import_xml("AAS_Workpiece_Transfer_Unit.xml")

nodes = [server.get_node(node) for node in aas_nodes]
```

2. **Linking Methods:** After identifying the nodeIDs, the next step is associating these nodeIDs with the appropriate methods or functions within the OPC UA Server. This step guarantees that the OPC UA Server can execute specific actions in response to incoming requests.

```
# Get node from AAS of Workpiece Trasnfer Unit in XML format and link method for
                                                    function of checking number of queue
pick_and_place_method_from_xml = server.get_node("ns=3;i=1606")
server.link_method(pick_and_place_method_from_xml, pick_and_place)

# Get node from AAS of Workpiece Trasnfer Unit in XML format and link method for
                                                    function of service
service_method_from_xml = server.get_node("ns=3;i=1630")
server.link_method(service_method_from_xml, service)
```

3. **Accessing Attributes:** The final step involves retrieving attributes associated with the operation submodel of AAS components. This procedure involves the customization of each node, in which attributes such as occupancy and maintenance information are specified to depict the operational behaviour of the workpiece transfer unit accurately.

```
# Get nodes from AAS of the Workpiece Transfer Unit in XML format for various
                                                    attributes
is_busy_node_from_xml = server.get_node("ns=3;i=1639")
is_under_service_node_from_xml = server.get_node("ns=3;i=1646")
is_at_service_position_node_from_xml = server.get_node("ns=3;i=1653")
number_of_queue_node_from_xml = server.get_node("ns=3;i=1660")
current_picking_direction_node_from_xml = server.get_node("ns=3;i=1667")
current_placing_direction_node_from_xml = server.get_node("ns=3;i=1674")
current_picking_module_id_node_from_xml = server.get_node("ns=3;i=1681")
current_placing_module_id_node_from_xml = server.get_node("ns=3;i=1688")
```

Consequently, this mapping process bridges the structured AAS data and the OPC UA Server, allowing seamless communication between the digital twin and the physical Workpiece Transfer Unit.

### Service Operation Method Integration

In conjunction with the prepared nodeIDs, the next stage entails integrating the extracted methods from the XML file with the operation methods developed in the earlier OPC UA server code from the previous winter semester project.

```
# Service operation
service_id = ua.Argument()
service_id.Name = "service_id"
service_id.DataType = ua.NodeId(ua.ObjectIds.Int32)
service_id.ValueRank = -1
service_id.ArrayDimensions = []
service_id.Description = ua.LocalizedText("Service Positions: 0 = none, 1-6 = Maintenance
                                                Positions")

result_s = ua.Argument()
result_s.Name = "result_s"
result_s.DataType = ua.NodeId(ua.ObjectIds.Boolean)
result_s.ValueRank = -1
result_s.ArrayDimensions = []
result_s.Description = ua.LocalizedText("Call successful")

# Populating the address space
await objects.add_method(idx, "service", service, [service_id], [result_s])
```

### Data Exchange

In the winter semester project, the OPC UA Server for the Workpiece Transfer Unit solely monitored the value of **nodeID_isBusy** from the UR5e Robot controller, utilizing it as a condition to determine whether the Workpiece Transfer Unit should transmit instructions to the UR5e OPC UA Server. However, in this project, this process has been significantly improved. The primary program now actively subscribes to multiple values from the UR5e OPC UA Server, encompassing occupancy, maintenance status, and pick and place operations. Subsequently, it publishes these acquired values to the relevant nodeIDs within the OPC UA Server-based AAS Workpiece Transfer Unit, establishing a robust data exchange mechanism.

```
# Read/update variables from UR5e OPC UA server
robot_busy = await read_var(nodeID_isBusy)
is_under_service = await read_var(nodeID_isUnderService)
service_position = await read_var(nodeID_atServicePosition)
current_pick_direction = await read_var(nodeID_pick_dir)
current_place_direction = await read_var(nodeID_place_dir)
current_pick_module_id = await read_var(nodeID_pick_id)
current_place_module_id = await read_var(nodeID_place_id)

# Send pick and place instruction if one is in the queue
if pap_queue.qsize() > 0 and not robot_busy:
    instruction = pap_queue.get()
    await asyncio.create_task(pap_action(instruction[0], instruction[1], instruction[2],
                                         instruction[3]))

# Write values back to AAS-based OPC UA server
await is_busy_node_from_xml.write_value(robot_busy)
await number_of_queue_node_from_xml.write_value(pap_queue.qsize())
await is_at_service_position_node_from_xml.write_value(service_position)
await is_under_service_node_from_xml.write_value(is_under_service)
await current_picking_direction_node_from_xml.write_value(current_pick_direction)
await current_placing_direction_node_from_xml.write_value(current_place_direction)
await current_picking_module_id_node_from_xml.write_value(current_pick_module_id)
await current_placing_module_id_node_from_xml.write_value(current_place_module_id)
```

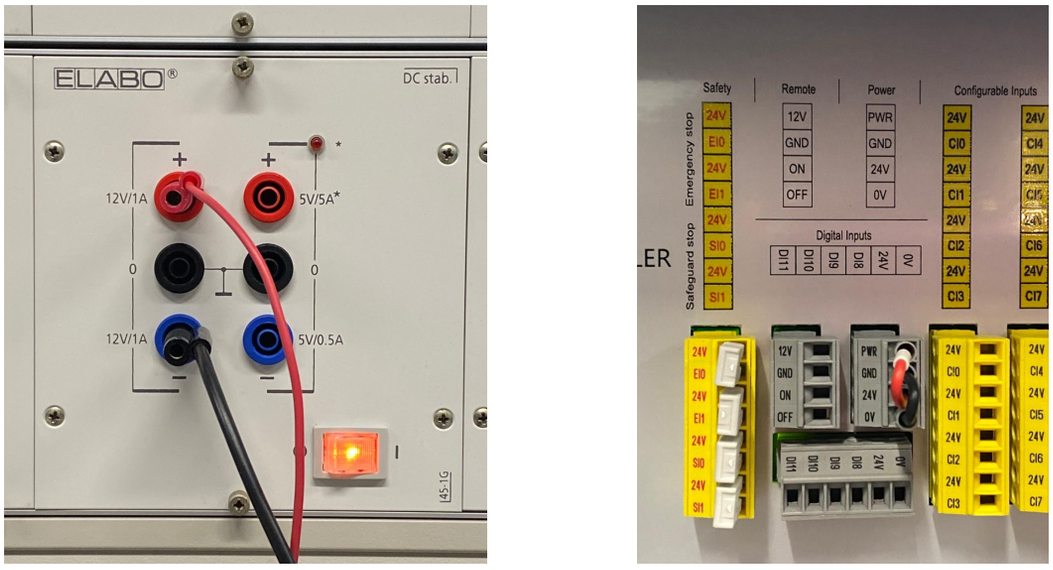
This code demonstrates how values from the UR5e OPC UA Server are subscribed to and then published to the corresponding nodeIDs in the OPC UA Server-based AAS Workpiece Transfer Unit. This promotes a dynamic and interconnected ecosystem within the digital factory by enhancing the interaction between the AAS-representative digital twin and the physical asset. Appendix B.2 contains the entire code of the updated OPC UA Server-based AAS Workpiece Transfer Unit. As well as appendix B.3 contains the entire code of the OPC UA Client used for communication with UR5e OPC UA Server.

## 5.4. Deployment of OPC UA Server into RevPi Core S

The designed OPC UA server must be deployed on the RevPi Core S hardware platform to be implemented to conform with the project's requirements. This deployment procedure establishes a communication center that facilitates interaction between the workpiece transfer unit and other interconnected modules in the manufacturing system. The RevPi Core S hardware platform is essential to assuring the reliability and stability of this communication infrastructure.

### 5.4.1. Power Supply and Connection Setup

Powering the Rev Pi in this project requires a 24VDC power source. It is ideal to connect the Rev Pi to the power source of the UR5e robot controller. This approach meets industry standards and ensures a stable and synchronized power supply. Alternatively, one can obtain 24VDC from the electrical laboratory table in the technikum laboratory. The Figure 5.15 below shows both options for the 24VDC power supply setup:

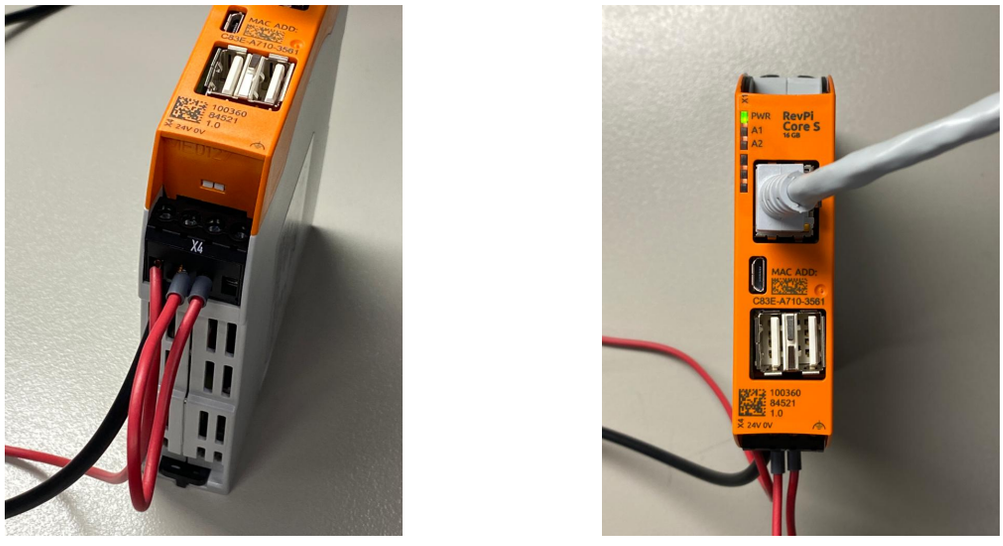



(a) 24VDC source - electrical table

(b) 24VDC source - robot control

Figure 5.15.: Options for 24VDC power source, own illustration

To enable effective communication, the Rev Pi must be connected to the Ethernet. In this project, it will be linked to the same Local Area Network (LAN) as the UR5e Robot Control. The following Figure illustrates the Rev Pi's connection to both the 24VDC power supply from the electrical laboratory table and an Ethernet cable in the technikum laboratory.




(a) Power Supply Connection to Rev Pi

(b) LAN Connection to Rev Pi

Figure 5.16.: Power Supply and Connection to the Rev Pi, own illustration

In conclusion, the setup of the Rev Pi, including the power supply and network connection, is important in establishing a reliable and robust communication hub within the industrial environment. Utilizing a 24VDC power source, ideally coupled with the UR5e robot controller's supply, ensures synchronization and adherence to industry standards. Additionally, connecting the Rev Pi to the Ethernet, specifically to the same local area network (LAN) as the UR5e Robot Control, enables communication between devices. The options for the 24VDC power supply and the Rev Pi's connections are depicted in Figure 5.15 and Figure 5.16, respectively.

### 5.4.2. SSH Access

Secure Shell (SSH) is the preferred method for accessing the command-line interface of the Rev Pi. When accessing the Rev Pi via SSH, the user establishes a secure connection that enables remote system configuration and control. The Rev Pi's default login credentials are as follows:

```
USERNAME OF SHELL:          pi

DEFAULT PASSWORD FOR        n9t7do
WEBSITE AND SHELL:
```

The command to access the Rev Pi's shell via Visual Studio Code's Remote Explorer is as follows:

```
ssh pi@192.168.158.62
```

After executing this command, assuming the correct credentials are provided, access is granted to the Rev Pi's file system. Once inside, navigate to the **'/home/pi'** directory. Inside this directory, the **'I40_Workpiece_Transfer_Unit'** folder will contain essential components for the project. This includes the OPC UA Server, the XML representation of the AAS Workpiece Transfer Unit, and the OPC UA Client specifically designed for use in conjunction with the UR5e robot.

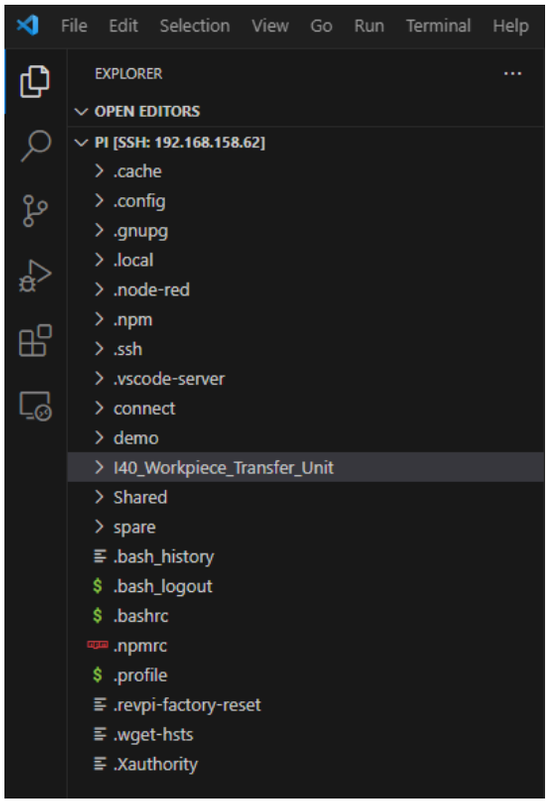



Figure 5.17.: SSH access to Rev Pi using Visual Studio Code, own illustration

In conclusion, SSH access is necessary for administering and configuring the Rev Pi and its associated components. It provides the necessary access for controlling and optimizing the system's performance.

### 5.4.3. Configuration Project Files

Several configuration steps are required to guarantee the OPC UA Server's correct operation. These stages are essential for aligning the server with the project's specific requirements:

1. **Setting the Server Endpoint:** The server's endpoint must be configured to correspond with the IP address of the connected terminal. This configuration enables the server to communicate with other devices over the network. In the Python script provided, this is accomplished as follows:

```
# Server setup
server.set_endpoint("opc.tcp://192.168.158.62:4840/AAS")
```

   This line identifies the server's communication endpoint, which consists of the IP address (in this instance, "192.168.158.62") and port number (typically 4840) used for OPC UA communication. The suffix "AAS" is added to the endpoint to identify the server as the "AAS Workpiece Transfer Unit."

2. **Importing the AAS in XML Format:** The server must be imported with the Asset Administration Shell (AAS) of the Workpiece Transfer Unit in XML format. This XML file contains the structured information about the AAS components and their relationships. In the provided Python script, this is accomplished using the following code:

```
# Import AAS of Workpiece Transfer Unit in XML format
aas_nodes = await server.import_xml("/home/pi/I40_Workpiece_Transfer_Unit/
                                     AAS_Workpiece_Transfer_Unit.xml")
```

   This code imports the AAS in XML format from the specified file path. The server then uses this XML data to obtain the necessary nodes and components for OPC UA communication. ("/home-/pi/I40_Workpiece_Transfer_Unit/AAS_Workpiece_Transfer_Unit.xml")

3. **Organizing Project Files:** After server configuration is complete, the next step is to organize the project files. This step is important for maintaining a structured and organized workspace. Relevant project files consist of the Python script for the AAS-based OPC UA Server, the script for the OPC UA Client used to communicate with the UR5e OPC UA Server, and the XML file for the AAS Workpiece Transfer Unit.

   These files should be placed in a dedicated folder named "I40_Workpiece_Transfer_Unit" The reason for this organization is to keep all project-related files together in one location, ensuring easy access and management.

   By centralizing these files in a single folder, it becomes straightforward to locate and work with them when needed. It enhances project efficiency and reduces the chances of files getting misplaced or lost. This structured approach to file organization contributes to a smooth and organized workflow throughout the project.

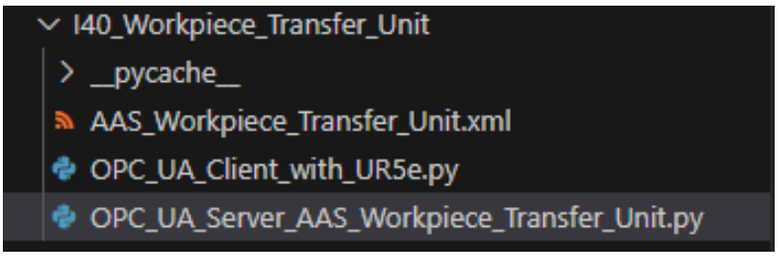



Figure 5.18.: I40 Workpiece Transfer Unit folder, own illustration

Proper configuration ensures that the OPC UA Server is correctly set up and prepare to facilitate communication within the industrial environment. These configuration steps align the server with the project's requirements and enable it to function in the digital manufacturing ecosystem.

### 5.4.4. Continuous Operation Setup

Once the essential files, including the Python script for the OPC UA Server, the OPC UA Client script, and the AAS Workpiece Transfer Unit XML file, have been placed into the 'I40_Workpiece_Transfer_Unit' folder, the OPC UA Server script must run continuously on the Rev Pi. This continuous operation is crucial for ensuring the seamless functioning of the system and its ability to handle any failures autonomously. Additionally, it should start automatically whenever the Rev Pi is powered on, providing the required stand-alone operation capability.

To accomplish this, a service must be developed on the Rev Pi. Follow the procedures below to configure the service:

1. **Create the Service File:** Begin by creating a service file in the directory **/etc/systemd/system**. For this report, the service file folder named 'I40_Workpiece_Transfer_Unit.' This can be accomplished manually by navigating to the specified path using Visual Studio Code's explorer tools, or via the terminal with the following command:

   ```
   sudo touch /etc/systemd/system/I40_Workpiece_Transfer_Unit.service
   ```

2. **Edit the Service File:** Use the sudo command to edit the newly created service file. Add the required information to define the service and enable it to restart in case of failure. Below is how the .service file can be configured:

   ```
   [Unit]
   Description=My long-running script
   After=network.target
   StartLimitIntervalSec=0

   [Service]
   Type=simple
   User=pi
   Restart=always
   RestartSec=1
   ExecStart=python3 /home/pi/I40_Workpiece_Transfer_Unit/
                                                OPC_UA_Server_AAS_Workpiece_Transfer_Unit
                                                .py

   [Install]
   WantedBy=multi-user.target
   ```

3. **Start and Enable the Service:** After creating and configuring the service, execute the following command in the Visual Studio Code terminal to initiate and enable it to run immediately when the Rev Pi restarts:

   ```
   sudo systemctl start I40_Workpiece_Transfer_Unit
   && sudo systemctl enable I40_Workpiece_Transfer_Unit
   ```

4. **Monitoring the Service:** Use the following command to observe the status of the service and ensure it is operating as expected:

   ```
   sudo systemctl status I40_Workpiece_Transfer_Unit
   ```

At this point, the deployment of the OPC UA Server on the RevPi Core S has been fully implemented. The 'I40_Workpiece_Transfer_Unit.service' is already running on the Rev Pi as part of its normal operation. By defining and enabling this service, the Python script of the OPC UA Server automatically activates as soon as the Rev Pi is powered on. It's essential to note that while the OPC UA Server runs autonomously, the UR5e Robot should also be powered on and running the URP Robot Programming for the complete functionality of the I4.0 Workpiece Transfer Unit.

## 5.5. Manual

This manual provides step-by-step instructions for operating the workpiece transfer unit. The following instructions should be followed carefully to ensure safe and efficient operation.

### 5.5.1. Operating Instructions

Before operating the unit, ensure that all necessary components are properly connected and in position. Detailed procedures related to system modifications or commissioning of new modules can be found in the respective chapters in the earlier described implementation.

#### Procedure

1. **Positioning**: Place the workpiece to be transported on one of the designated tables.
2. **Power On**: Turn on the power supply for the workpiece transfer unit.
3. **Connectivity**: Ensure that the robot control is connected to the network, and the UR5e OPC UA server is active.
4. **Initiate Robot Program**: Run the robot program on the robot control using the teach pendant. The robot can be in automatic or manual mode for this step.
5. **Initiate Rev Pi**: Connect the microcontroller to a 24VDC power source and ensure it's on the same local area network as the UR5e robot controller.
6. **Access OPC UA Server - Aligned with AAS of workpiece transfer unit**: To access the OPC UA server, utilize UaExpert or consider creating a custom OPC UA client. Submodels can be accessed as needed.
7. **Consume Unit's Operation Capability**: If the unit's operation is required, such as calling the "pick and place" function and "service" function, they can be utilized.
    a) **Pick and Place Function**: Call the "pick and place" function with the required id and direction for pick and place positions. After receiving the specified variables, the robot arm will move to the designated position to pick up the workpiece and transport it to the target position. The program will continue running and wait for further instructions.
    b) **Maintenance Function**: The robot program includes a maintenance function. If maintenance is required, use "service" function The robot arm will move to the designated maintenance position. Maintenance mode can be stopped by pressing the continue button on the teach pendant.

#### Emergency Stop

In case of an emergency, press the emergency stop button on the control panel to immediately halt the workpiece transfer unit's operation.

#### Workpiece Size and Weight

The workpiece transfer unit is designed to accommodate workpieces within a specific size range and weighing up to 2 kg. If a workpiece exceeds these specifications, conduct thorough testing before execution, as the gripper may not handle it effectively.

# 6. Test & Validation

This chapter describes the extensive testing and validation procedures to ensure the robustness, functionality, and conformance of the I4.0 Workpiece Transfer Unit project's requirements. The chapter is divided into two major sections titled "Integration, Test, and Verification" and "System Verification and Validation."

In the first section, the operational validation of robot movements and programming, which ensures precise and reliable robotic operations, is examined. In addition, information access through the Asset Administration Shell (AAS) and control and monitoring via the OPC UA server are validated. In addition, the server's capabilities for continuous operation are evaluated.

The second section revisits project requirements and evaluates their compatibility with system design and implementation. It also includes validating overall system functionality to ensure the I4.0 Workpiece Transfer Unit performs as intended and meets the project's objectives.

These strict testing and validation procedures are required to ensure the reliability and efficacy of the I4.0 Workpiece Transfer Unit in an Industry 4.0 environment.

## 6.1. Integration, Test, and Verification

This section explores the critical integration, testing, and verification processes that ensure the functionality and compliance of the I4.0 Workpiece Transfer Unit within the broader manufacturing ecosystem. It focuses on several important aspects, each discussed in its subsection.

### 6.1.1. Operational Validation of Robot Movements and Programming

This subsection focuses on validating the robot's movements and programming from an operational standpoint. It ensures the accuracy, precision, and safety of robotic operations, crucial to dependable and efficient manufacturing processes.

#### Robot Programming

As described in Section 5.1 - Configuration of UR5e Robot Programming, the UR5e Robot program's installation and INI files, as well as the URP program, have been updated. This subparagraph focuses on the proof of these modifications to the robot's programming.

To initiate the program, follow these steps:

1. Use the teach pendant and click the "Open" icon.
2. Access the **OPCUA_LJ_Henrik_17072023_6Base.urp** file and click "Open."
3. The installation file, named **OPC_UA_LJ_Henrik_Installation_17.07.2023.installation** will automatically load with the URP file.
4. To begin the robot programme, click the "Run" option. The robot can operate in either automatic or manual mode during this phase.

Figure 6.1 illustrates the successful execution of the configured UR5e Robot programming using the teach pendant. In the absence of instructions from the OPC UA server, the robot remains in a loop, waiting for commands.

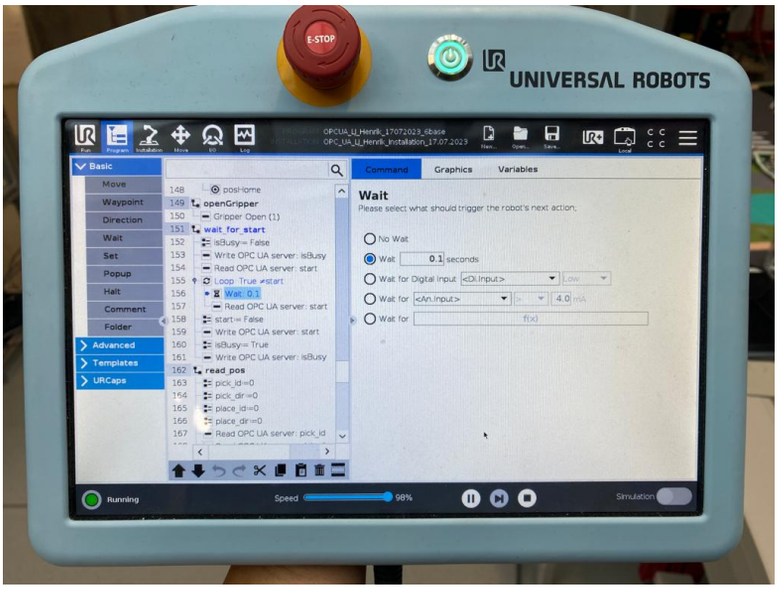



Figure 6.1.: Operational Validation of Robot Programming, own illustration

This process ensures that the robot's programming can be executed without error, providing a foundation for precise and controlled robotic operations within the industrial environment.

### Robot Movement

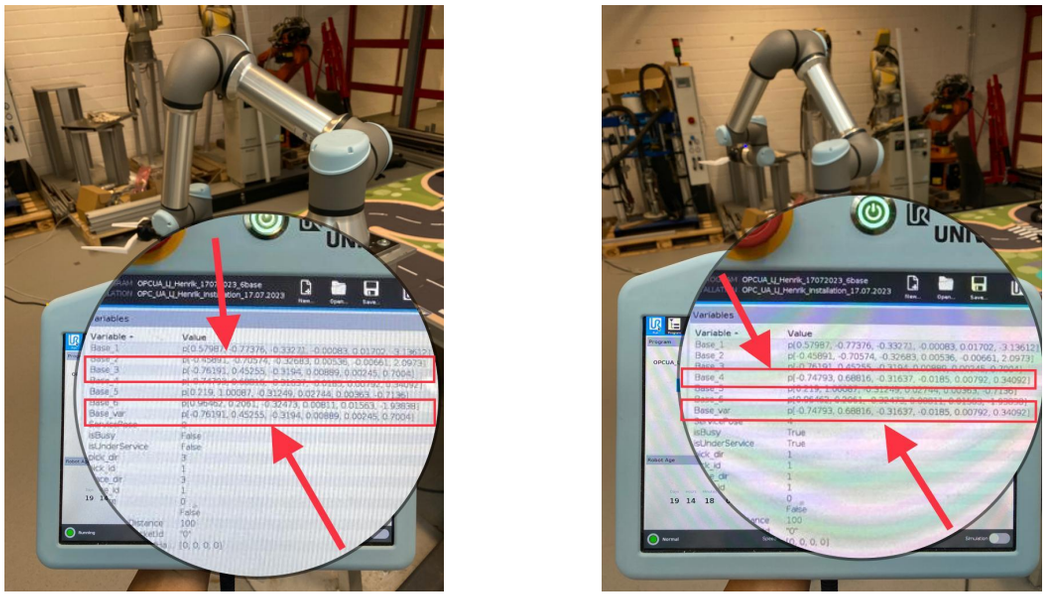
The validation of robot movement comes into focus once the robot's program is operating. This validation evaluates the robot's ability to accurately and safely implement predefined movements. The procedure entails generating and transmitting variable commands, including 'Place direction 3', 'Service 4', and 'Service 5', to the URP program via the OPC UA Server.

As shown in Table 6.1 below, three additional base planes have been configured in the robot for this project. These bases, designated Base 3, Base 4, and Base 5, function as reference points within the URP program for directing the robot's movement in directions 3, 4, and 5. These bases are utilized when the URP program receives a command through the OPC UA Server.

| | Position Vector | | | Rotation Vector | | |
|---|---|---|---|---|---|---|
| Base | x | y | z | rx | ry | rz |
| Base_3 | -0.76191 | 0.45255 | -0.3194 | 0.00889 | 0.00245 | 0.7004 |
| Base_4 | -0.74793 | 0.68816 | -0.31637 | -0.0185 | 0.00792 | 0.34092 |
| Base_5 | 0.219 | 1.00087 | -0.31249 | 0.02744 | 0.00363 | -0.7136 |

Table 6.1.: Position and Rotation Vector of 3 Added Base Planes

Figures 6.2a and 6.2b depict the robot's responses to various commands received via the OPC UA Server. In Figure 6.2a, when the command 'place direction = 3' is received, the robot moves in the direction 3, and the alignment of the pose variable of Base 3 and the Base variable indicates that the movement was successfully executed. In addition to Figure 6.2b, the robot's response to the'service = 4' command is illustrated. It moves to direction 4, entering a stop state indicating maintenance mode and requiring manual intervention via the teach pendant to resume. Similar to the former scenario, the alignment of the pose variable of Base 4 with the Base variable verifies the precise implementation of the movement.



(a) Robot Movement using Base 3

(b) Robot Movement using Base 4

Figure 6.2.: Operation Validation of Robot Movement, own illustration

Similarly to Figure 6.2b, Figure 6.3 depicts the robot's response to the command 'service = 5' received via the OPC UA Server. The robot moves to direction 5 and enters a halt state, denoting maintenance mode, which requires manual intervention to resume the program at the teach pendant of the robot control. Similar to the previous examples, it is indicated that the pose variable of Base 5 and the Base variable match up, validating the precise execution of the movement.

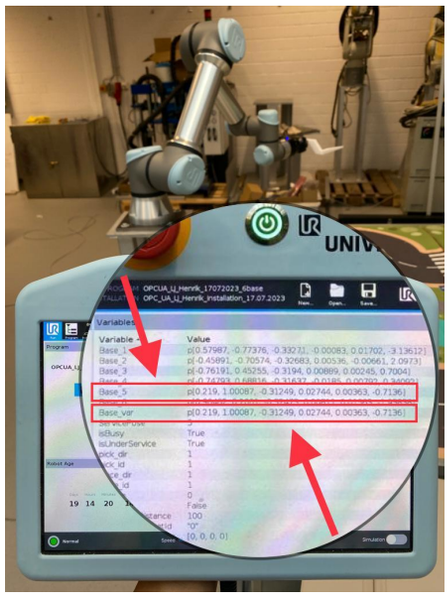


Figure 6.3.: Robot Movement using Base 5, own illustration

This validation procedure ensures the robot can implement precise and controlled movements following commands received via the OPC UA Server. This demonstration of its dependability in an industrial setting demonstrates its capacity to contribute to error-free and efficient manufacturing processes.

### 6.1.2. Validation of AAS - based Information Access

In this section, the effectiveness of information access via the Asset Administration Shell (AAS) framework is validated, assuring the retrieval of accurate data from the AAS.

#### Accessing the AAS of Workpiece Transfer Unit

The AAS for the workpiece transfer unit was created through the OPC UA Server, as explained in Section 5.3 - Creation of OPC UA Server Aligned with AAS. Furthermore, it has been deployed into the Rev Pi, as detailed in Section 5.4 - Deployment of OPC UA Server into RevPi Core S. While the robot program is operational, access to AAS-based information is facilitated through OPC UA specifications. In this report, UaExpert is employed as an OPC UA client, enabling access to the server via the following endpoint:

```
opc.tcp://192.168.158.62:4840/AAS
```

Upon accessing this endpoint, the AAS for the Workpiece Transfer Unit is displayed, and its structure, including submodels, views, and assets, aligns precisely with the configurations made during the development phase, as detailed in Section 5.2.2 - Developing the Asset Administration Shell within Chapter 5 - Implementation.

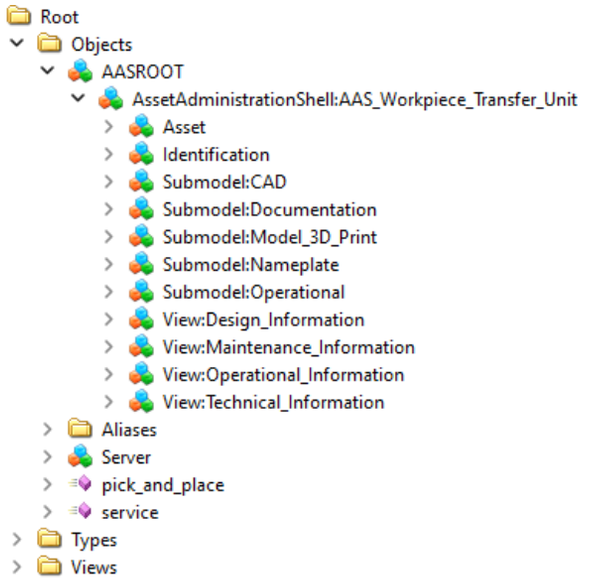



Figure 6.4.: Accessing AAS Worpiece Trasnfer Unit via UaExpert, own illustration

Figure 6.4 above depicts the Asset Administration Shell (AAS) for the Workpiece Transfer Unit when accessed via UaExpert. This image depicts the structure and components of the AAS, including submodels, views, and assets. It demonstrates how the AAS data is structured and made available for additional validation and analysis.

After this, complete validations are performed on each submodel to ensure its information is accurate and accessible. This validation process is essential, particularly for submodels that contain significant information about the unit. The subsequent submodels will be evaluated:

- Validation of CAD Submodel
- Validation of 3D Print Submodel
- Validation of Documentation Submodel

### Validation of CAD Submodel

This section emphasizes CAD submodel validation. To initiate the validation procedure, access 'Submodel:CAD.' Subsequently, three primary Submodel Element Collections will become visible, each comprising specific collection-specific properties.

For illustrative purposes, Figure 6.5 below displays the values of properties within the Submodel Element Collections of 'Gripper Assembly,' 'Setting Base Tool Assembly,' and 'Workpiece Transfer Unit with Digital Factory.' Notably, all property values are presented as string values in the form of URL Dropbox links, as previously detailed in the design. However, All the files included in Dropbox can be seen in Appendix A - Detailed Design.

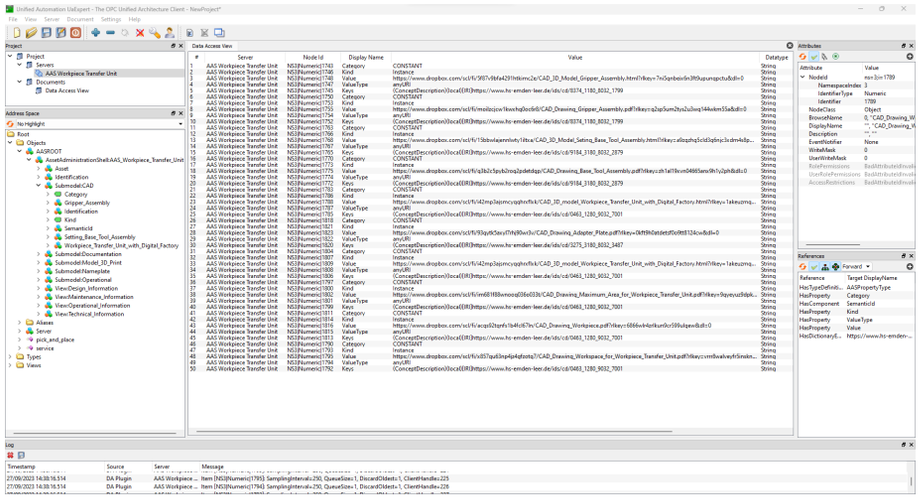


Figure 6.5.: Validation of CAD Submodel - All Property's Value, own illustration

Once all property values are accessible, the validation of information access involves evaluating one property from each Submodel Element Collection in the following manner:

- **Gripper Assembly SMC**

  Within the this SMC, the property 'CAD 3D Model Gripper Assembly' is chosen for inspection. Access the specified property to obtain its value, returned as a string containing a Dropbox URL. This link provides access to the 3D CAD files in eDrawings Web HTML format, as shown in Figure 6.6.

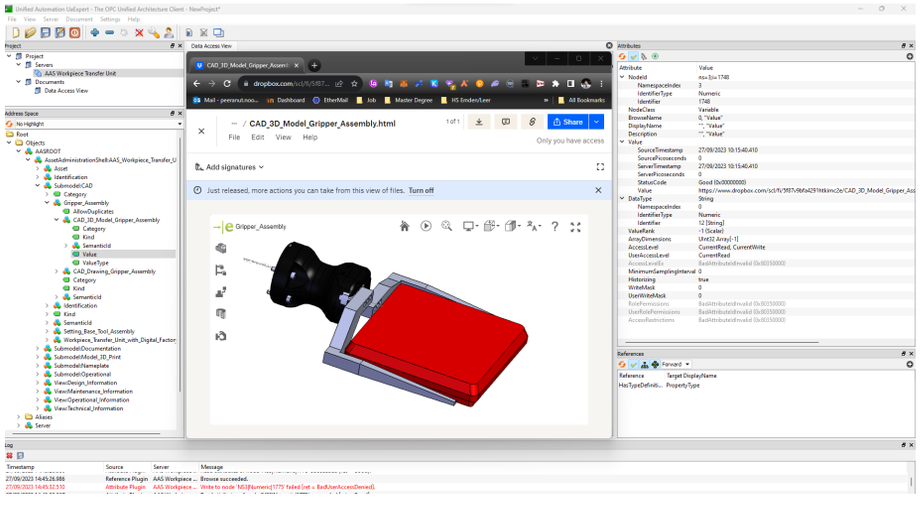


Figure 6.6.: Validating Access to Property of CAD 3D Model Gripper Assembly, own illustration

- **Setting Base Tool SMC**

  Within the Setting Base Tool Assembly SMC, the emphasis is on the property 'CAD Drawing Base Tool Assembly.' Accessing this property retrieves its value, which is a string containing a Dropbox connection URL. This link provides access to the PDF-formatted CAD drawing files, as depicted in Figure 6.7.

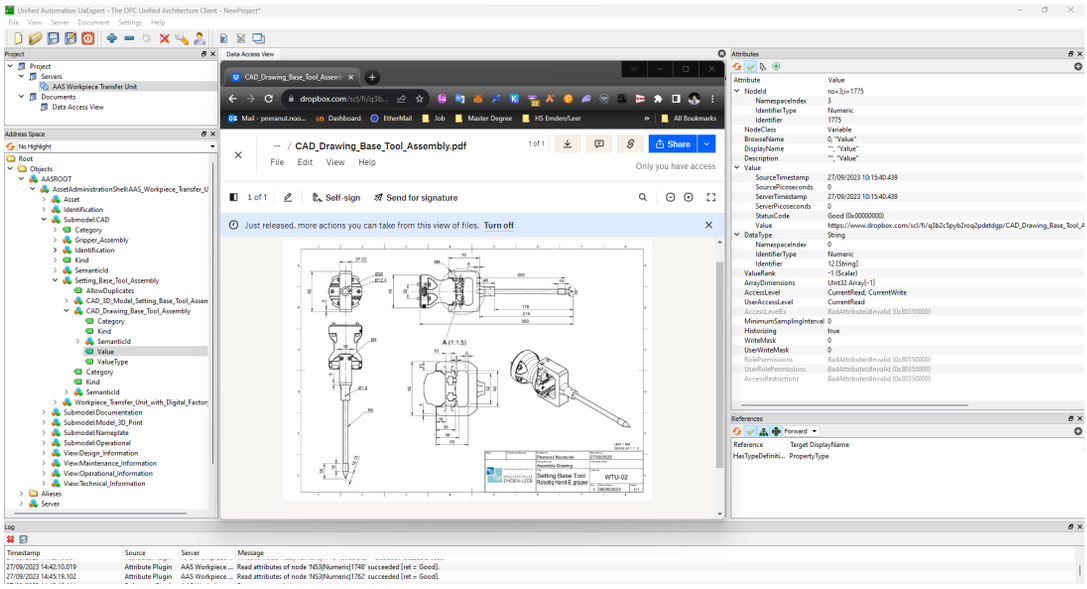



Figure 6.7.: Validating Access to Property of CAD Drawing Setting Base Tool, own illustration

- **Workpiece Transfer Unit with Digital Factory SMC**

  Examining the property 'CAD 3D Model Workpiece Transfer Unit with Digital Factory' within the SMC of Workpiece Transfer Unit with Digital Factory. Accessing this property retrieves its value, which is a string containing a Dropbox URL. As shown in Figure 6.8, this link provides access to the 3D CAD files in eDrawing Web HTML format.

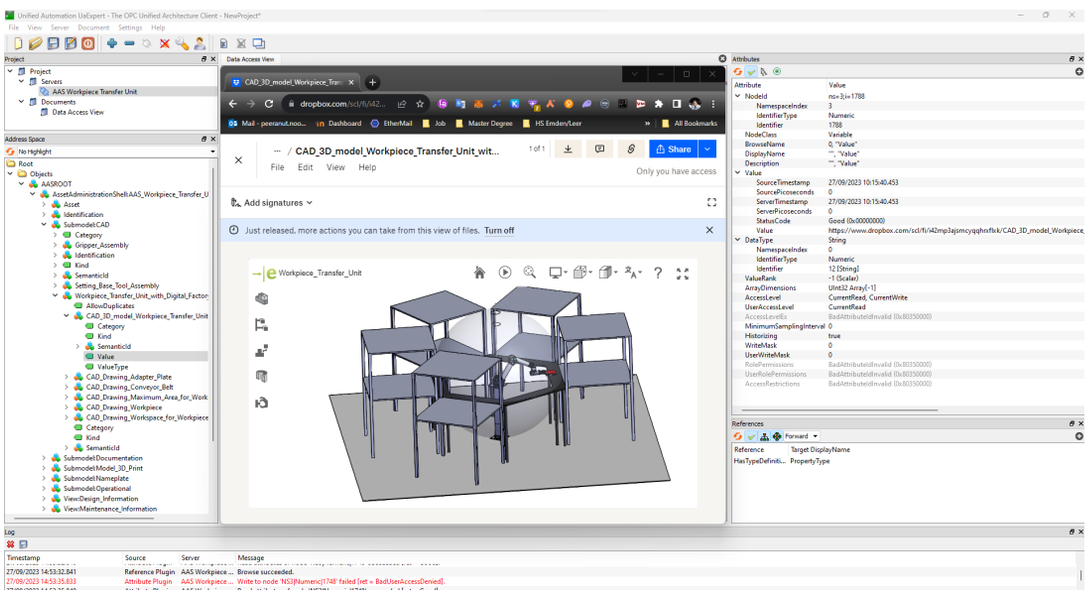



Figure 6.8.: Validating Access to Property of CAD 3D Model Workpiece Transfer Unit with Digital Factory, own illustration

### Validation of 3D Print Submodel

This subsection focuses on the validation of the 3D Print submodel. Navigating to a Submodel Element Collection inside the "Submodel:3D Print" node is the procedure. The selection made for verification is the "Gripper" Submodel Element Collection.

This collection contains properties for the "Gripper Left 3D Model" and the "Gripper Right 3D Model." In this illustration, the "Gripper Left 3D Model" property will undergo validation. Displays a string containing a Dropbox link to an STL-formatted 3D model file of the left portion of the gripper upon accessing its value.

This URL can be opened in any internet browser, allowing access to the 3D model for download without restrictions, as depicted in Figure 6.9.

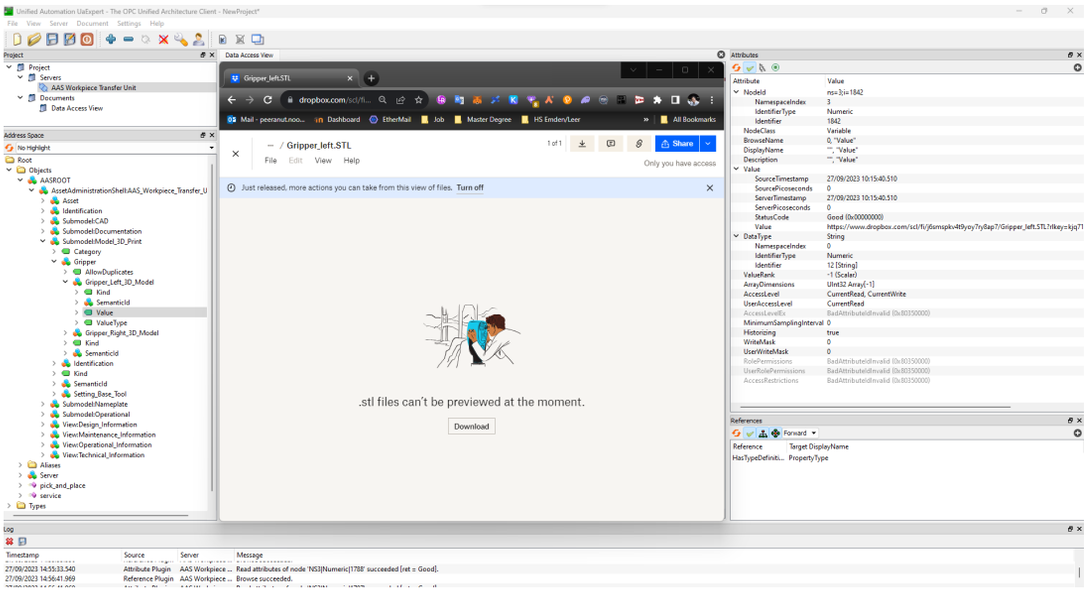



Figure 6.9.: Validating Access to Property of Gripper Left 3D Model, own illustration

This validation verifies that the 3D Print submodel provides functional and accessible connections to the 3D models of the gripper components.

### Validation of Documentation Submodel

This subsection focuses on the documentation submodel's validation. The procedure requires accessing "Submodel:Documentation" and navigating to the "Submodel Element Collection" section of Technical Specification. Within this compilation, both the German and English versions of the properties are accessible.

In this example, the validation will be performed on the "UR5e_Technical_Specification_de" property. Upon accessing its value, a string containing a Dropbox link to the respective document is displayed. This URL is accessible via any internet browser, enabling document retrieval.

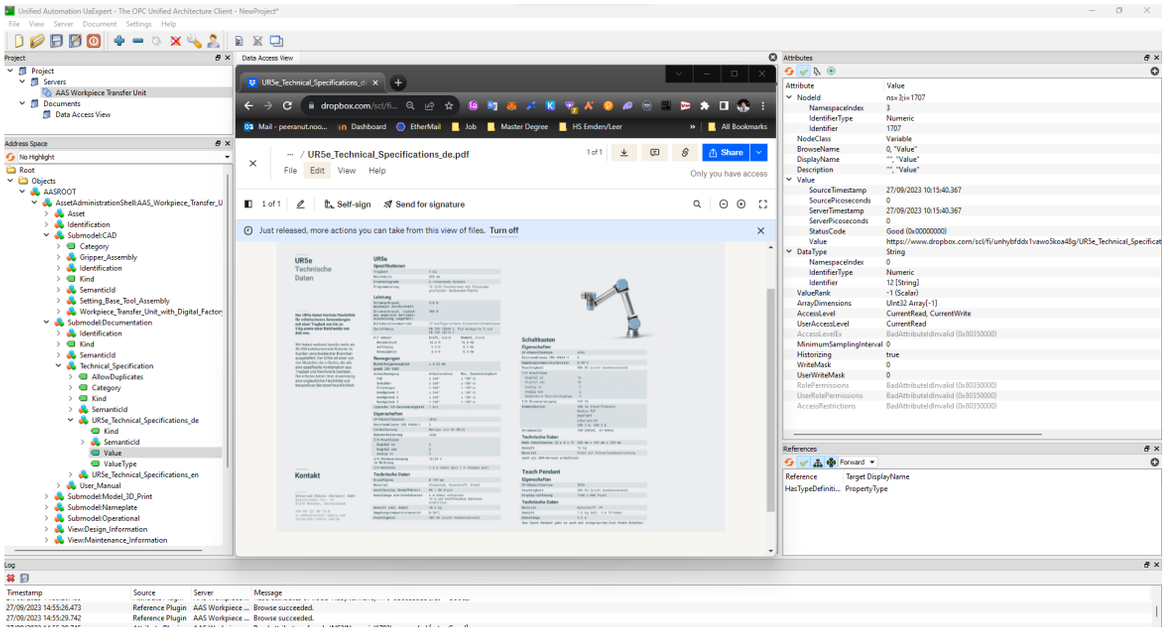


Figure 6.10.: Validation of Documentation Submodel - Technical Specification, own illustration

The result of this validation process confirms the successful retrieval of the PDF document for the UR5e Technical Specification, as illustrated in Figure 6.10.

## 6.1.3. Control and Monitoring Verification via OPC UA Server

Through the OPC UA server, the following evaluations validate the system's control and monitoring capabilities. These tests evaluate the system's capability to transmit control commands and receive monitoring data.

### Monitoring Verification via OPC UA Server

The OPC UA Server, implemented using Python programming, has been configured to write the occupancy status, maintenance status, and pick & and place operation information into specific node IDs. These node IDs are listed in Table 6.2, enabling comprehensive system monitoring.

| Property | Node Id |
|---|---|
| is_busy | ns=3;i=1639 |
| number_of_queue | ns=3;i=1660 |
| is_at_service_position | ns=3;i=1653 |
| is_under_service | ns=3;i=1646 |
| current_picking_direction | ns=3;i=1667 |
| current_placing_direction | ns=3;i=1674 |
| current_picking_module_id | ns=3;i=1681 |
| current_placing_module_id | ns=3;i=1688 |

Table 6.2.: Node Id of Operational Property

In this report, each of these Node IDs can be accessed through UaExpert, which is essential as an OPC UA Client. UaExpert is a software tool used for interacting with OPC UA Servers. To access these Node IDs, follow these steps:

- Access the **"Submodel:Operational"** within the **OPC UA Server**, where these Node IDs are located.
- Within the **Submodel:Operational**, the properties listed in Table 6.2 will be found.

Table 6.2 contains the Node IDs corresponding to various operational properties. These Node IDs are unique identifiers for specific data points or properties within the OPC UA Server.

Once the Node IDs are located, UaExpert can be used to interact with them:

- Open **UaExpert**, and establish a connection with the **OPC UA Server** where the **AAS Workpiece Transfer Unit** is hosted.
- Navigate to the **Data Access View** area within **UaExpert**.
- Drag and drop the Node IDs, corresponding to the properties of interest, from Table 6.2 into the **Data Access View**.

By doing this, data from the OPC UA Server shall be effectively monitored and accessed . Figure 6.11 below illustrates this process:

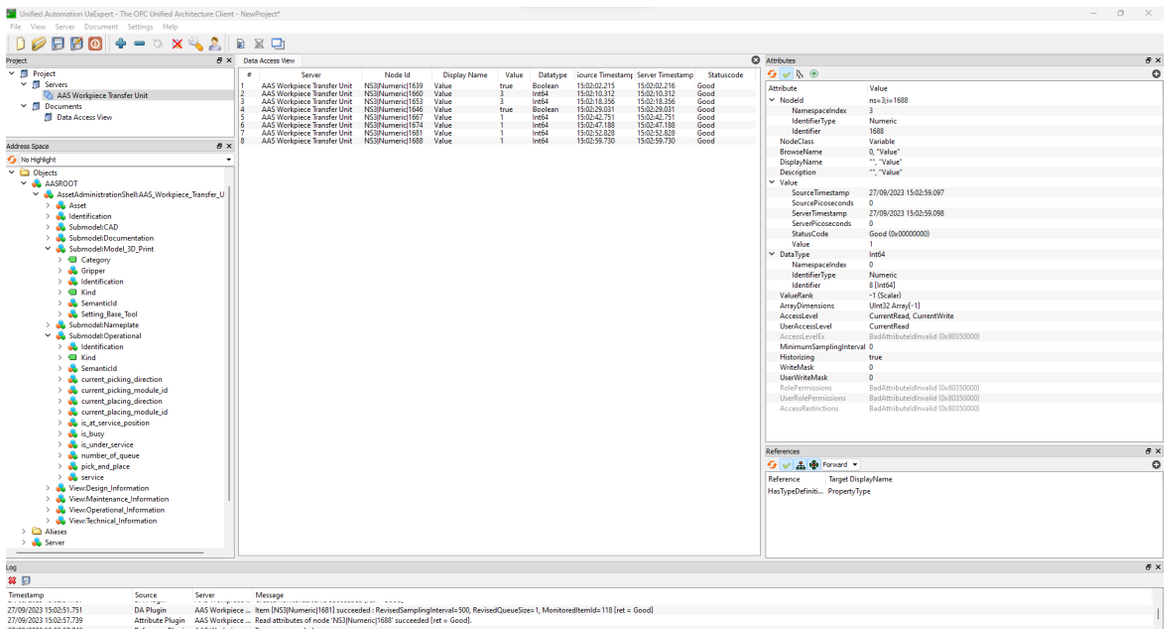


Figure 6.11.: Monitoring Verification via OPC UA Server, own illustration

On Figure 6.11, the operational information of the AAS Workpiece Transfer Unit can be observed. It provides a visual representation of the current condition, including information on occupancy, maintenance, and pick-and-place operations. Consequently, this graphical representation verifies the successful monitoring verification through the OPC UA Server.

Therefore, this subsubsection provides a comprehensive explanation of how to access and utilize UaExpert to monitor the OPC UA Server and validate the system's capabilities. It also highlights the importance of the graphical representation in confirming effective monitoring.

### Control Verification via OPC UA Server

Control Verification via OPC UA Server must be validated once access to the AAS Workpiece Transfer Unit's operating data is achieved. This validation procedure consists of two separate operations which connected to pre-existing methods within the OPC UA Server. These actions consist of:

- **Pick and Place Operation**

  For the Pick and Place operation, it can be initiated by accessing the "Submodel: Operation" and then proceeding to the "Opr" (Operation) related to Pick and Place. Upon selecting the operation icon, the operation interface will appear. In this interface, input arguments can be provided.

  In a particular case investigation, the workpiece is to be picked from module ID 2, which is positioned strategically in direction 1, and carefully placed in module ID 1, which is set in direction 3. As seen in Figure 6.12, entering these inputs is demonstrated.

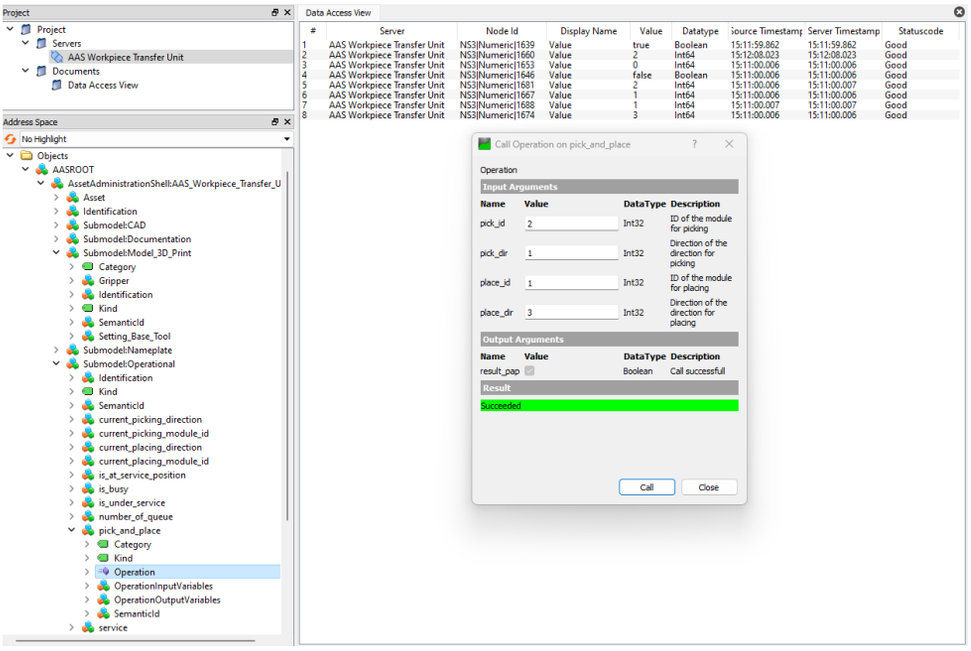



Figure 6.12.: Controlling Verification of Pick and Place Operation, own illustration

The successive procedures involved in inserting these instructions are also shown in Figure 6.12. It also acts as proof that the instruction was carried out successfully. It is important to note that the instruction was transmitted twice in this case, which is why the Data Access View interface's second row displays the number of queues as two, indicating that the unit is presently processing two instructions that are queued.

- **Service Operation**

  To initiate the Service operation, one should access the "Submodel: Operation" and then select the "Opr" (Operation) associated with Service. Upon selecting the operation icon, the interface for the operation will appear, allowing input of specific parameters.

  It is important to understand that using the Service operation is conditional upon certain requirements. Only when the unit is actively engaged in the pick and place operation can this function be initiated. This limitation results from the fact that the primary purpose of the Service operation is to cater to maintenance requirements. In a specific scenario, the unit is directed to enter maintenance mode at direction 5. In Figure 6.13, the detailed process of input parameter entering is clearly illustrated.

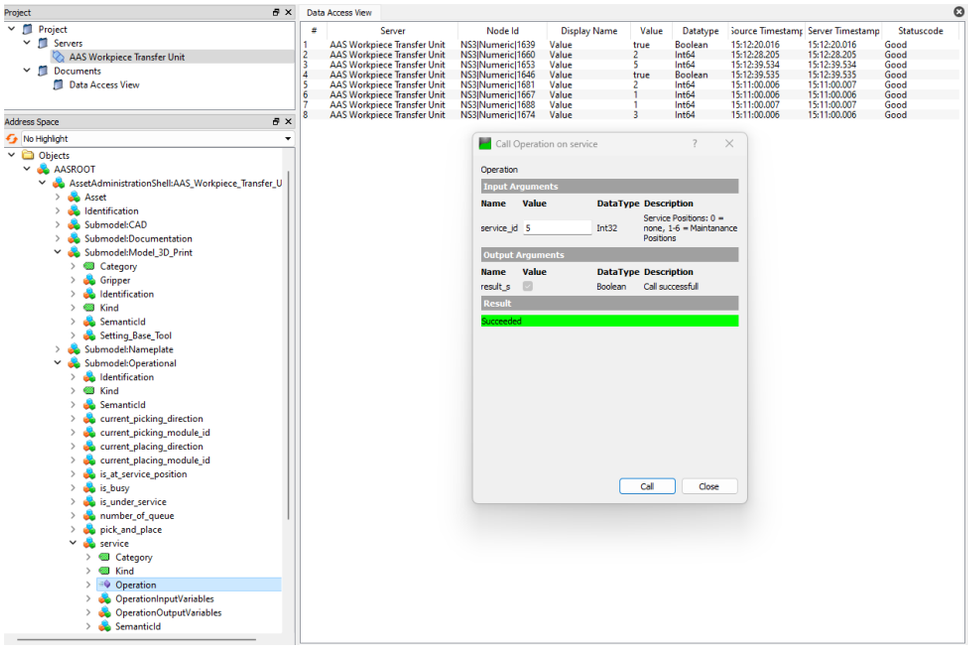



Figure 6.13.: Controlling Verification of Service Operation, own illustration

Following the successful execution of the Service operation, the UR5e robot's programming enters a temporary halt stage, operating under case 5. It can only be continued to the normal operation after the continue button in the UR5 teach pendant is pressed. Subsequently, it smoothly transitions back to its regular pick and place operations, as Figure 6.14 displays.

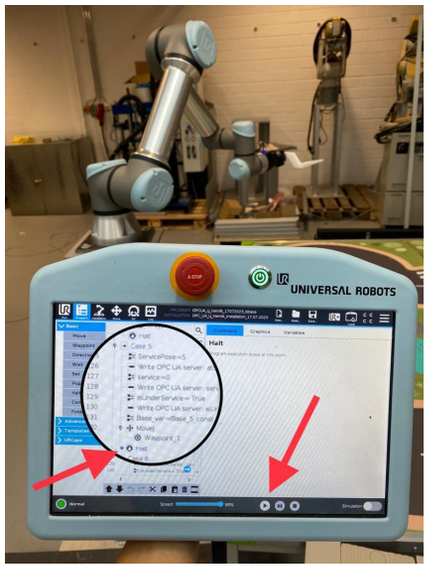



Figure 6.14.: Workpiece Transfer Unit Under Service Operation At Position 5, own illustration

This comprehensive overview encapsulates the intricacies of both the Pick and Place operation and the Service operation within the broader context of Control Verification via OPC UA Server.

### 6.1.4. Continuous Operation Verification of OPC UA Server

OPC UA server reliability during extended operations is evaluated via Continuous Operation Verification. It demonstrates the efficiency of the Rev Pi's deployed and operational automatic recovery systems. The OPC UA Server Based on the AAS of the Workpiece Transfer unit was placed into the Rev Pi and set up for automatic operation using a Service File, as explained in subsection 5.4.4 - Continuous Operation Setup. To verify the server's functionality in these circumstances, this subsection, Continuous Operation Verification of OPC UA Server, is provided.

Certain requirements must be satisfied to perform this verification. Initially, confirm that the programming of the UR5e robot is operational and that the Rev Pi is powered on and connected to the same local area network as the UR5e Robot controller. This will allow SSH access to be maintained for real-time monitoring. Two significant time intervals are utilized for verification: the first observation occurs at the 2-hour mark, and a subsequent one is conducted at the 5-hour mark. Using the following command, keep an eye on the status of the service during these assessments:

```
sudo systemctl status I40_Workpiece_Transfer_Unit
```

As depicted in Figure 6.15, the initial observation at 2 hours and 37 minutes indicates that the service script continues to execute error-free, providing information regarding the unit's status of occupancy and maintenance.

```
pi@RevPi84521:~ $ sudo systemctl status I40_Workpiece_Transfer_Unit
● I40_Workpiece_Transfer_Unit.service - My long running script
     Loaded: loaded (/etc/systemd/system/I40_Workpiece_Transfer_Unit.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2023-09-27 10:15:28 CEST; 2h 37min ago
   Main PID: 12552 (python3)
      Tasks: 2 (limit: 2049)
     CGroup: /system.slice/I40_Workpiece_Transfer_Unit.service
             └─12552 /usr/bin/python3 /home/pi/I40_Workpiece_Transfer_Unit/OPC_UA_Server_AAS_Workpiece_Transfer_Unit.py

Sep 27 12:53:20 RevPi84521 python3[12552]: Robot busy = False
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot is under service = False
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot is at service position = 0
Sep 27 12:53:20 RevPi84521 python3[12552]: Queue size = 0
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot busy = False
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot is under service = False
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot is at service position = 0
Sep 27 12:53:20 RevPi84521 python3[12552]: Queue size = 0
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot busy = False
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot is under service = False
pi@RevPi84521:~ $
```

Figure 6.15.: Continuous Operation Verification at 2 Hours, own illustration

OPC UA Server Based on the AAS Workpiece Transfer Unit on the Rev Pi remains operational, as determined by an examination at the 5-hour and 3-minute mark. The server efficiently restarts itself despite the brief display of a watchdog error in the terminal, as demonstrated in Figure 6.16.

```
pi@RevPi84521:~ $ sudo systemctl status I40_Workpiece_Transfer_Unit
● I40_Workpiece_Transfer_Unit.service - My long running script
     Loaded: loaded (/etc/systemd/system/I40_Workpiece_Transfer_Unit.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2023-09-27 10:15:28 CEST; 2h 37min ago
   Main PID: 12552 (python3)
      Tasks: 2 (limit: 2049)
     CGroup: /system.slice/I40_Workpiece_Transfer_Unit.service
             └─12552 /usr/bin/python3 /home/pi/I40_Workpiece_Transfer_Unit/OPC_UA_Server_AAS_Workpiece_Transfer_Unit.py

Sep 27 12:53:20 RevPi84521 python3[12552]: Robot busy = False
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot is under service = False
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot is at service position = 0
Sep 27 12:53:20 RevPi84521 python3[12552]: Queue size = 0
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot busy = False
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot is under service = False
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot is at service position = 0
Sep 27 12:53:20 RevPi84521 python3[12552]: Queue size = 0
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot busy = False
Sep 27 12:53:20 RevPi84521 python3[12552]: Robot is under service = False
pi@RevPi84521:~ $
```

Figure 6.16.: Continuous Operation Verification at 5 Hours, own illustration

This careful verification proves that the OPC UA Server operates reliably and can recover from errors during continuous operation, confirming its capability to sustain uninterrupted service in industrial applications.

## 6.2. System Verification and Validation

Evaluating compliance with established requirements is needed to assess the verification and validation of the Industry 4.0 workpiece transfer unit. The criteria mentioned in the text include connectivity, documentation, non-functional, and functionality. This section will provide a detailed analysis of how the design and implementation of the project conform to the specified criteria, as well as reflect on the verification and validation procedure.

### 6.2.1. Functional Requirements Verification

- **Seamless Module Integration and Quick Adaptation**

  This requirement demands a thorough assessment to ensure the system's capacity for seamless module integration and rapid adaptation to fluctuating production needs. Testing the system's capacity to easily incorporate new modules, reconfigure its operation, and quickly exchange and replace components is part of the verification procedure. Successful validation would indicate that the system can effectively adapt to changing production requirements and increase operational flexibility.

  Even with the successful completion of the AAS of the workpiece transfer unit, it is important to acknowledge that this particular requirement still needs to be fulfilled due to the incomplete construction of the digital factory.

- **Improved Interconnectivity and Data Exchange**

  This requirement aimed to enhance the system's data exchange capabilities among its modules. Hence, the goal was achieved by integrating the workpiece transfer unit with the Asset Administration Shell (AAS) framework, a standardized method for information exchange in digital factories. Utilizing the AAS framework, the unit seamlessly communicates with other modules, facilitating the efficient exchange of operational data, machine statuses (e.g., busy or idle), and maintenance information. This integration not only empowers operators to manage production workflows efficiently but also grants them control over the unit's pick-and-place operations and other related functionalities.

  By successfully integrating the workpiece transfer unit with the AAS framework, critical operational information becomes accessible to other modules and external systems. This upgraded approach significantly enhances operational efficiency and overall flexibility. Consequently, the verification process confirms that the system fully complies with the requirement for improved interconnectivity and data exchange, marking a crucial milestone in optimizing digital factory operations.

- **Streamlined Production Management and Decision-making**

  The system's capacity to provide constant access to critical information, such as nameplate data, user manuals, and technical specifications, was thoroughly examined. Since successfully implementing the Asset Administration Shell (AAS) for the workpiece transfer unit, it has established a unified repository for detailed technical documentation. This includes crucial information like nameplate details, user manuals, and technical specifications.

  By adhering to Industry 4.0 standards and exposing this valuable data through the AAS framework, the unit ensures that the integrator and stakeholders have comprehensive insights into module operations. This access empowers stakeholders to make informed decisions, plan production effectively, and optimize resource allocation based on the provided operational insights. This strategic approach significantly improves production efficiency and enhances competitiveness in the digital factory ecosystem. Therefore, this requirement is fully met,

- **Operate in All Six Directions**

  Validation encompassed a comprehensive assessment of the system's capability to operate in all six directions within the digital factory environment. During implemetation and verification, the Industry 4.0 workpiece transfer unit is equipped with the Universal Robot UR5e, a robotic arm capable of six-axis movement. The UR5e is designed to execute a wide range of workpiece manipulation operations with remarkable accuracy.

  Successful validation of this requirement confirms that the system can execute intricate workpiece manipulations in all six directions, contributing significantly to the cohesive functioning of the digital factory ecosystem. Consequently, it is evident that this requirement is fully met, given the unit's advanced capabilities and precise control over multidirectional movements.

- **Standalone Operation**

  In the context of verifying the 'Standalone Operation' requirement, the system's capability to function independently was assessed. An OPC UA server, based on the Asset Administration Shell (AAS) of the workpiece transfer unit, was implemented on the RevPi. In addition to service files and coding strategies, this server was capable of being booted up during operation and featured robust performance. Even in the face of unexpected disruptions like power supply interruptions or network connectivity loss, the RevPi was designed to restore itself. This validation demonstrated the system's ability to meet the 'Standalone Operation' requirement, showcasing its autonomous operation and its capacity to establish reliable data exchange and information sharing mechanisms within the digital factory ecosystem.

  It is critical to mention, nevertheless, that the system did experience a 'BadCommunicationError' in the OPC UA server on occasion, as evidenced by the UaExpert log. Typical causes of this error included malfunctions in the monitor loop and session renewal. Fortunately, in response to such errors, a program service file was implemented to resume the OPC UA server on the RevPi automatically. This proactive approach ensured that any server downtime was minimal, usually limited to one minute or less. This additional information highlights how the system effectively mitigated disruptions, maintaining its reliability in real-world manufacturing environments.

- **Virtual Representation**

  The system's capability to deliver a comprehensive virtual representation, including 3D CAD models and CAD drawings, has been validated. Designers of additional modules now have access to these visual assets for workspace planning, reference, and documentation. Maintenance personnel can efficiently comprehend and execute assembly tasks for additional components, ensuring correct assembly procedures and optimal utilization.

  The Industry 4.0 workpiece transfer unit's integration with the Asset Administration Shell (AAS) has fully realized this capability. It now provides 3D CAD files in eDrawing Web HTML format and CAD Drawing PDF files, guaranteeing accessibility for various purposes, including workspace planning and reference. This approach empowers users and stakeholders to effortlessly access and utilize the visual assets for their specific needs, facilitating efficient workspace planning, precise referencing, and streamlined assembly procedures. As such, this validation conclusively confirms that the requirement for Virtual Representation is entirely met, enhancing the system's utility in digital manufacturing environments.

- **3D Equipment Model Maintenance**

  Verification has conclusively established that the system offers robust support to maintenance stakeholders in the renewal or reproduction of 3D models of equipment and tools in the event of damage or malfunction. This comprehensive assistance package encompasses the availability of 3D model files in STL format for all necessary components, intuitive assembly visualization using 3D CAD files in eDrawing Web HTML format, and assembly instructions accessible through CAD Drawing PDF files. Furthermore, this support integrates with common 3D printing software and hardware.

The Industry 4.0 workpiece transfer unit's integration with the Asset Administration Shell (AAS) has ensured the fulfilment of this requirement in its entirety. Maintenance personnel now have at their disposal all the essential resources and tools required to efficiently renew or reproduce equipment and tools, thereby significantly reducing downtime and enhancing overall maintenance capabilities.

### 6.2.2. Non-Functional Requirements Verification

- **CP Notation Positioning**

  By validating this non-functional requirement, the system's accuracy in categorizing its present position within the Classification of Presentation and Communication (CP) notation has been guaranteed. This accomplishment sets forth a strong basis for understanding the characteristics and functionalities of the system, improving the coordination of tasks, and offering clarity on the approach to digitizing the current unit.

  How the assets of the Industry 4.0 workpiece transfer unit were defined is reflected in this validation procedure, allowing a comprehensive understanding of its presentation and communication capabilities. To summarize, the precise placement of CP notation has been effectively implemented.

- **RAMI 4.0 Conformance**

  Validation confirms that the system meticulously adheres to the principles of the Reference Architecture Model for Industry 4.0 (RAMI 4.0). This dedication ensures the system seamlessly aligns with Industry 4.0 standards, fostering effortless integration and robust interoperability within the digital factory ecosystem. The Industry 4.0 workpiece transfer unit's architectural design, communication protocols, and data formats have been crafted to harmonize with the comprehensive standards of Industry 4.0.

  By steadfastly adhering to RAMI 4.0's principles, the unit lends its support to the concept of "Digital Twins." This innovation enables the creation of a digital replica of a physical unit, thereby enhancing monitoring and optimization capabilities. With the successful implementation of these principles, it can be confidently asserted that every aspect of this requirement has been meticulously met, thereby bolstering the system is standing as a robust and compliant component of the digital factory ecosystem.

### 6.2.3. Connectivity Requirements Verification

- **Industry 4.0 Protocol and Interface**

  The Industry 4.0 Protocol and Interface requirement necessitates that the chosen connectivity protocol and interface conform strictly to the standards of Industry 4.0 while serving as recognized industry standards. This imperative ensures the integration and communication of the Industry 4.0 workpiece transfer unit with other components of the digital factory ecosystem.

  The unit has implemented the OPC UA (Open Platform Communications Unified Architecture), a widely recognized communication protocol in the industry, to fulfil this requirement. OPC UA is renowned for its standardized and secure data exchange capabilities, which ensure compatibility with other components of Industry 4.0. By implementing OPC UA, the Industry 4.0 workpiece transfer unit will be able to collaborate with many modules and entities that comprise the digital manufacturing ecosystem. By strategically selecting this option, one not only confirms the fulfilment of the prerequisite but also affirms complete adherence to the rigorous criteria of the Industry 4.0 ecosystem.

### 6.2.4. Documentation Requirement Verification

- **Comprehensive Project Background**

  The Documentation Specification places a key emphasis on providing a necessary background within the project report. This requirement demands the inclusion of a comprehensive overview of various concepts linked to the development of the Industry 4.0 workpiece transfer unit. The objective is to convey the fundamental principles for a deep understanding of the unit's digitalization within the extensive landscape of Industry 4.0.

  The project report has supplied what was required to meet this requirement. It carefully lays out an outline that covers all of the ideas involved in the development of the Industry 4.0 workpiece transfer unit. Readers will be well-equipped to understand the project's goals and its greater implications in the context of Industry 4.0 according to its extensive documentation's explanation of key concepts and contextual significance. Since this requirement is met, the project report becomes a resource that will help stakeholders understand the project's underlying principles and its crucial position within the context of Industry 4.0.

Integration, Testing, and Verification have been completed as the final phase of the Industry 4.0 workpiece transfer unit project. The unit has almost fulfilled all of the project's criteria by attending to each requirement. This concludes the seamless incorporation of Industry 4.0 principles into the undertaking.

Particularly, a specific requirement, which requires seamless module integration and quick adaptation to fluctuating production demands, has encountered some difficulties. Even though the Asset Administration Shell (AAS) for the workpiece transfer unit has been finalized, this is due to the ongoing development of the digital factory. Despite this, the project continues progressing smoothly, and the verified and validated capabilities of the unit guarantee effective functioning within the digital factory ecosystem.

Furthermore, the system encountered irregular limitations associated with the watchdog loop and session renewal while validating the standalone operation requirement. These obstacles lead to the display of 'BadCommunicationError' notifications on the OPC UA server. Fortunately, these issues have been effectively addressed by implementing a program service file that automatically restarts the OPC UA server on the RevPi, resolving these concerns with minimal impact. Despite these challenges, the project remains on track, and the unit's capabilities, as confirmed and validated, promise efficient operation within the digital factory ecosystem.

Therefore, by confirming the unit's robustness and compliance with established requirements, the System Verification and Validation procedure ensures optimal functioning within the digital factory ecosystem. By conducting these comprehensive verifications and validations, the foundation for the unit's successful integration into the Industry 4.0 landscape has been laid.

# 7. Conclusion & Outlook

## 7.1. Conclusion

The Industry 4.0 workpiece transfer unit project has achieved significant milestones in alignment with Industry 4.0 principles. The project successfully developed and integrated a highly advanced and versatile system designed for the digital manufacturing landscape. However, it is essential to acknowledge that full requirements have not been met due to the ongoing development of the digital factory.

### 7.1.1. Digital Factory Enhancement

The project has made substantial progress in achieving seamless integration and adaptation capabilities through the implementation of the Asset Administration Shell (AAS) framework. This framework enables efficient communication and data exchange with other modules. Nevertheless, the project's ultimate potential in this regard is constrained by the evolving nature of the digital factory. As new modules and technologies are integrated, further adaptations will be required to maintain seamless operation.

### 7.1.2. Adaptability as a Continuous Process

Adaptability is not a one-time achievement but an ongoing process. As the digital factory matures, the workpiece transfer unit must continue to evolve to accommodate new production needs, technologies, and modules. This includes regular updates to the AAS and software to ensure compatibility with emerging systems. Therefore, rather than viewing this as a shortcoming, it should be seen as an inherent aspect of Industry 4.0, where adaptability is a continuous journey.

## 7.2. Outlook

While the project has made significant strides, there are areas for further exploration and enhancement:

### 7.2.1. Continued Digital Factory Development

The Industry 4.0 workpiece transfer unit has been designed with adaptability, allowing for seamless integration of new modules and quick adaptation to changing production needs. As the digital factory continues to evolve, further refinements will be essential to ensure the unit remains at the forefront of efficiency. This includes continuous evaluation and integration of new modules, ensuring that the unit can readily incorporate them into the production process. Moreover, ongoing collaboration with other units within the digital factory will be crucial to optimize overall operations continuously.

### 7.2.2. Integration with Other Modules

Expanding integration with other modules and systems within the digital factory is vital for creating a more interconnected and efficient production ecosystem. As the digital factory evolves and introduces new modules, it is crucial to ensure the Industry 4.0 workpiece transfer unit can seamlessly integrate with these additions. This entails both technical adaptability and active collaboration with other units. This enhanced integration fosters a more interconnected production environment, facilitates scalability, and enables cross-unit data exchange, ultimately optimizing the overall efficiency of the digital factory ecosystem.

### 7.2.3. Error Mitigation

Further refinement of error mitigation strategies, particularly addressing the occasional 'BadCommunicationError' encountered during standalone operation, is essential to enhance the system's reliability. While the project has achieved autonomous operation, it is vital to improve its robustness in handling unexpected situations. Investigating and mitigating the 'BadCommunicationError - Error in watchdog loop' may involve optimizing the monitor loop, enhancing session renewal mechanisms, or implementing more robust error recovery procedures. The overarching aim is to minimize disruptions in data communication, especially in industrial environments where uninterrupted operation is paramount. This focus on error mitigation ensures the workpiece transfer unit's continued dependability and resilience in real-world manufacturing scenarios, aligning with the core principles of Industry 4.0.

### 7.2.4. Enhanced Standalone Operation

While the project has achieved autonomous operation, future improvements can be explored. One promising avenue is to consider supplying power directly from the UR5e robot controller's 24VDC source. This modification aligns with industrial practices and enhances the unit's reliability, particularly when power supply interruptions are a concern. Such an upgrade ensures that the workpiece transfer unit is well-suited for deployment in demanding industrial environments, where uninterrupted operation is a critical requirement.

### 7.2.5. Advanced File Attachment Handling

The project leverages the OPC UA protocol for secure and standardized data exchange. To further enhance efficiency, future developments could focus on improving the OPC UA specification. One significant area of improvement is the handling of file attachments. Currently, files such as PDFs, STLs, and HTMLs are stored externally, often in services like Dropbox. Future iterations of the project could explore ways to handle these file attachments directly within the OPC UA server. This would reduce reliance on external services, simplifying file management and streamlining data exchange processes. The use of APIs or plugins may be developed to facilitate seamless file handling, retrieval, and updates directly within the OPC UA server. This approach reduces the complexity of managing external file repositories, enhances security by keeping critical documents within the industrial network, and improves overall data integrity and accessibility.

By addressing these outlook items, the Industry 4.0 workpiece transfer unit can evolve into a more advanced, efficient, and adaptable component within the digital factory ecosystem. These enhancements align with the core principles of Industry 4.0, emphasizing connectivity, data-driven decision-making, and the continuous pursuit of operational excellence in the rapidly evolving landscape of smart manufacturing.

# Bibliography


[Ber01] O. Bergmann, *Industrial data transport technologies*, HS Emden Leer, 2023-03-01.

[BML+20] B. Boss *et al.*, "Digital twin and asset administration shell concepts and application in the industrial internet and industrie 4.0: An industrial internet consortium and plattform industrie 4.0 joint whitepaper," p. 33, 2020. (visited on 08/02/2023).

[Col01] A. W. Colombo, *Digitalization of icps: The digitalization transformation*, HS Emden/Leer, 2023-03-01.

[Col21] A. W. Colombo, *Engineering industrial cyber-physical system: Life cycle engineering of industrial cyber-physical systems (icps)*, 2022-09-21.

[Col23] A. W. Colombo, *Industrial cyber-physical systems (icps)*, HS Emden Leer, Mar. 6, 2023.

[eVar] I. D. T. A. e. V. "Teilmodelle." Accessed on 2023-09-20. (Year), [Online]. Available: https://industrialdigitaltwin.org/content-hub/teilmodelle.

[HS 22] HS Emden Leer, *Projects for the digital factory: Winter semester 2022/23*, 2022.

[MLD09] W. Mahnke, S.-H. Leitner, and M. Damm, *OPC Unified Architecture*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, ISBN: 978-3-540-68898-3. DOI: 10.1007/978-3-540-68899-0.

[NM23] P. Noonurak and H. Meyer, *Workpiece transfer unit: Semester project report - ws22/23*, Mar. 26, 2023. (visited on 06/06/2023).

[PAB+16] Plattform Industrie 4.0 *et al.*, *Reference architecture model industrie 4.0 (rami4.0)*, DIN SPEC 91345, English translation of DIN SPEC 91345:2016-04, 2016. [Online]. Available: https://www.beuth.de/en/technical-rule/din-spec-91345/2482458 (visited on 06/06/2023).

[Pi20] R. Pi. "Revpi core base module - industrial raspberry pi." Accessed on: 02-08-2023. (2020), [Online]. Available: https://revolutionpi.com/revpi-core.

[Sch16] K. Schweichhart, *Reference architectural model industrie 4.0 (rami 4.0): An introduction*, 2016. [Online]. Available: https://ec.europa.eu/futurium/en/system/files/ged/a2-schweichhart-reference_architectural_model_industrie_4.0_rami_4.0.pdf.

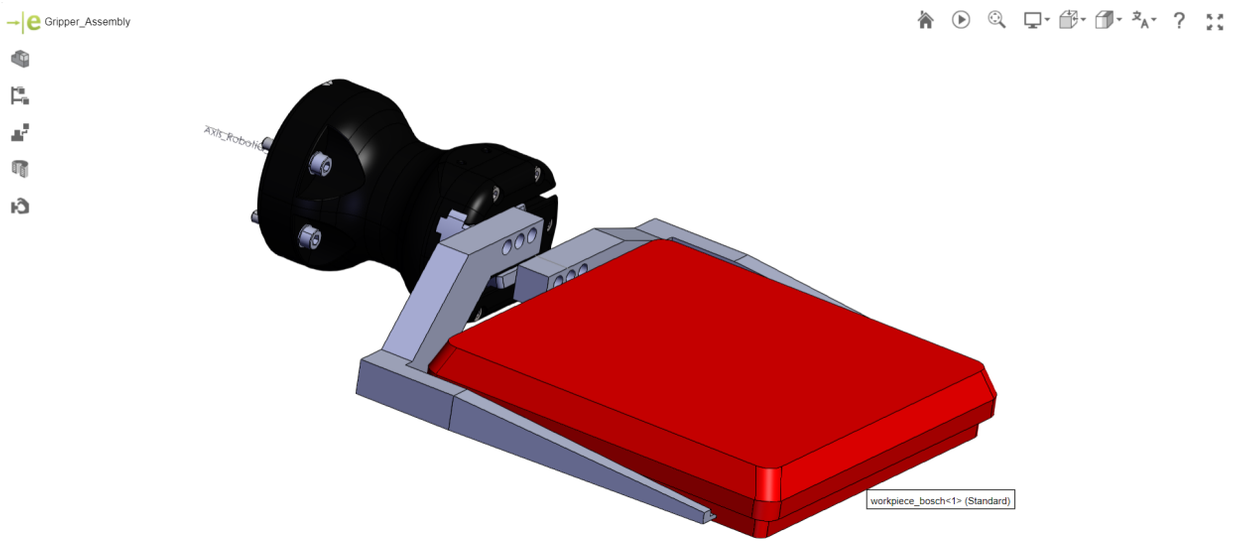
[Uni18] Universal Robots, *Universal robots e-series user manual: Ur5e: Orginal instrcution (en)*, 2018.

# A. Detailed Design

## A.1. SMC of Gripper Assembly

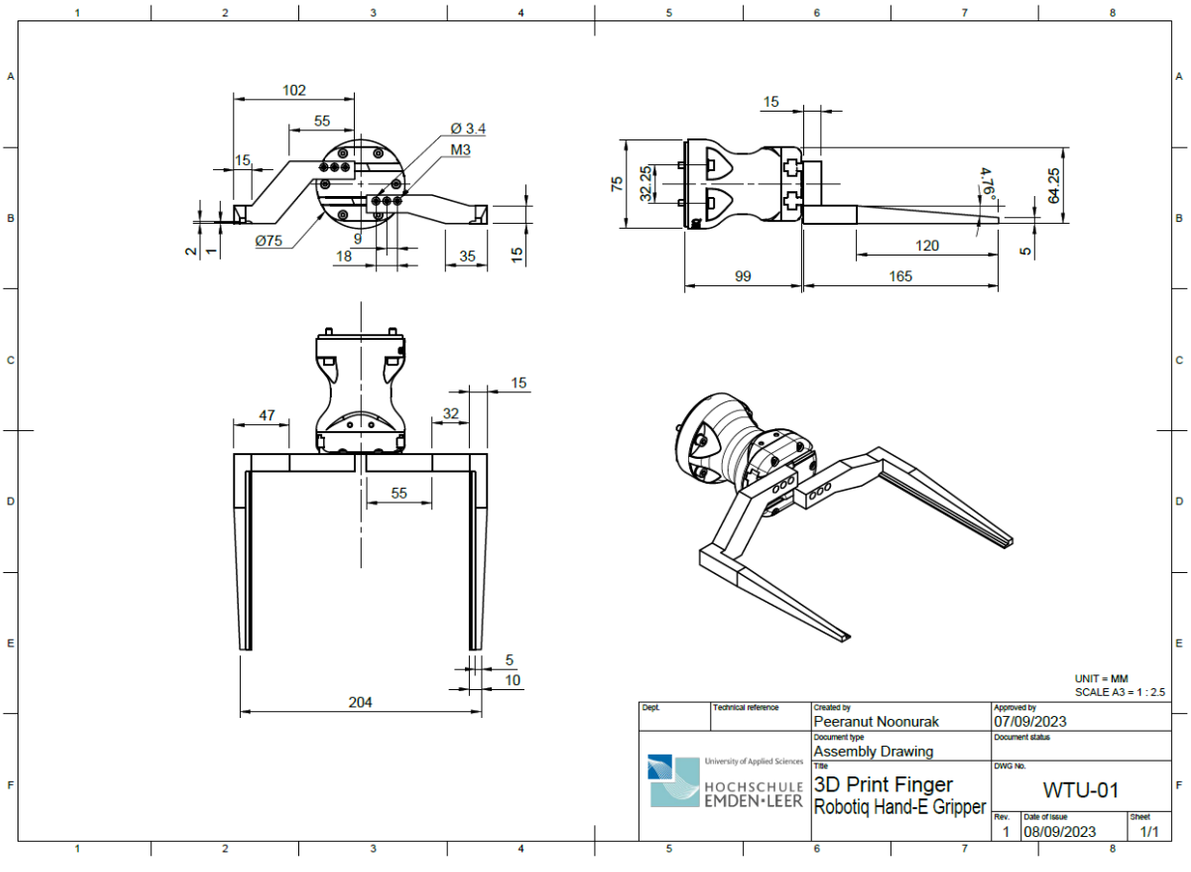
### A.1.1. CAD 3D Model of Gripper Assembly

This model provides a detailed representation of the Gripper Assembly's design, allowing for a visual understanding of its structure and components. It is a reference for comprehending the intricacies of this assembly with the Robotiq Hand-E gripper. Own illustration.



## A.1.2. CAD Drawing of Gripper Assembly

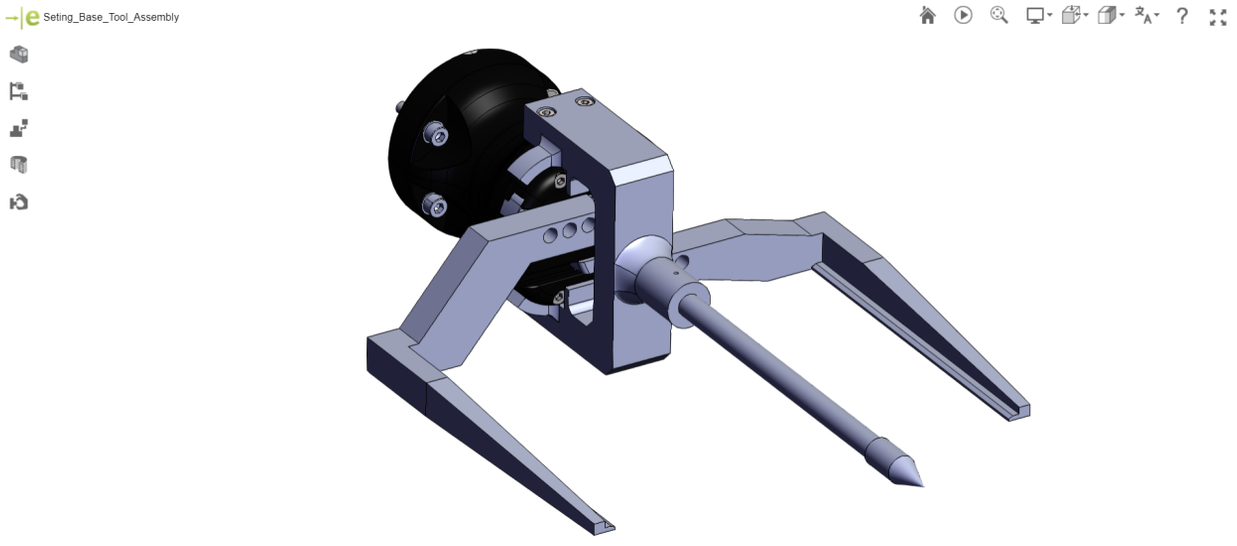
The CAD drawing provides precise engineering documentation of the Gripper Assembly's components and dimensions. It serves as a reference for understanding the assembly's design, helping to visualize its structure and configuration with the Robotiq Hand-E gripper. Own illustration.



# A.2. SMC of Setting Base Tool

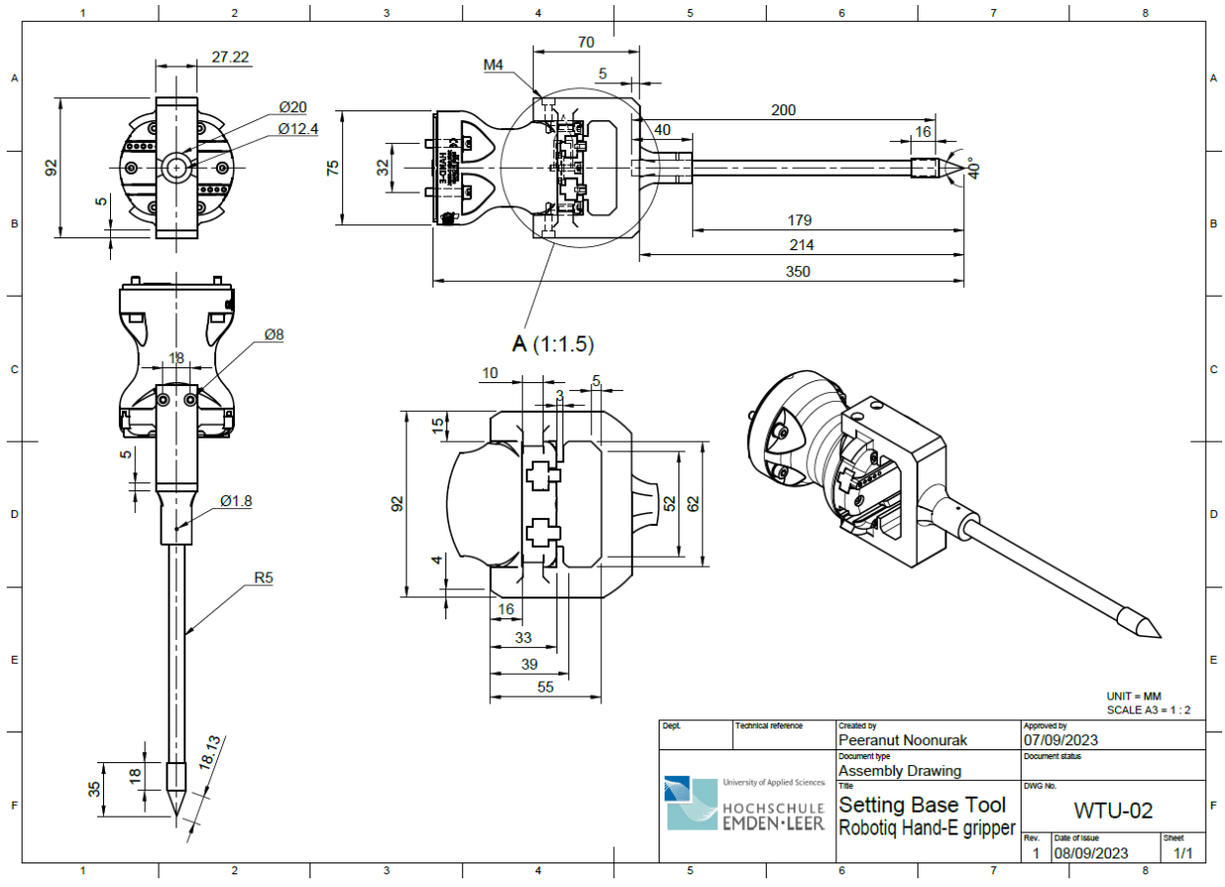
## A.2.1. CAD 3D Model of Setting Base Tool

This model represents the Setting Base Tool's physical structure, including its components and overall design, in great detail. It is a visual reference for understanding the complex details of the tool's construction. This model represents the Setting Base Tool's physical structure, including its components and overall design, in great detail. It is a visual reference for understanding the complex details of the tool's construction. Own illustration.



## A.2.2. CAD Drawing of Setting Base Tool

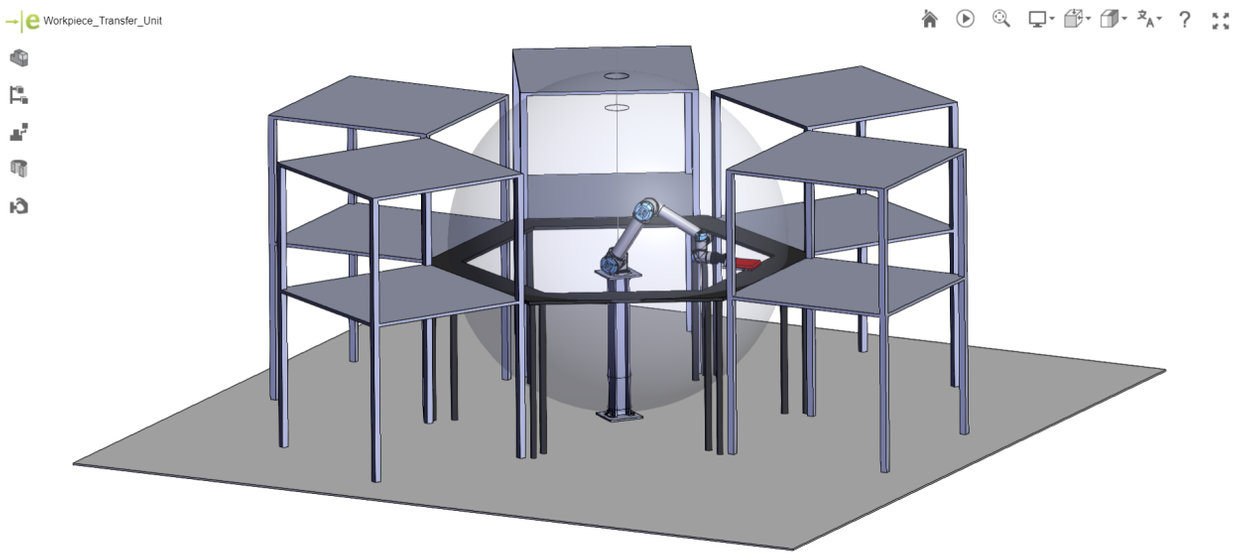
This drawing provides a precise representation of the tool's design, highlighting key features and dimensions. It is a reference for understanding the tool's structure and how it fits into the Robotiq Hand-E gripper. Own illustration.



## A.3. SMC of Workpiece Transfer Unit with Digital Factory

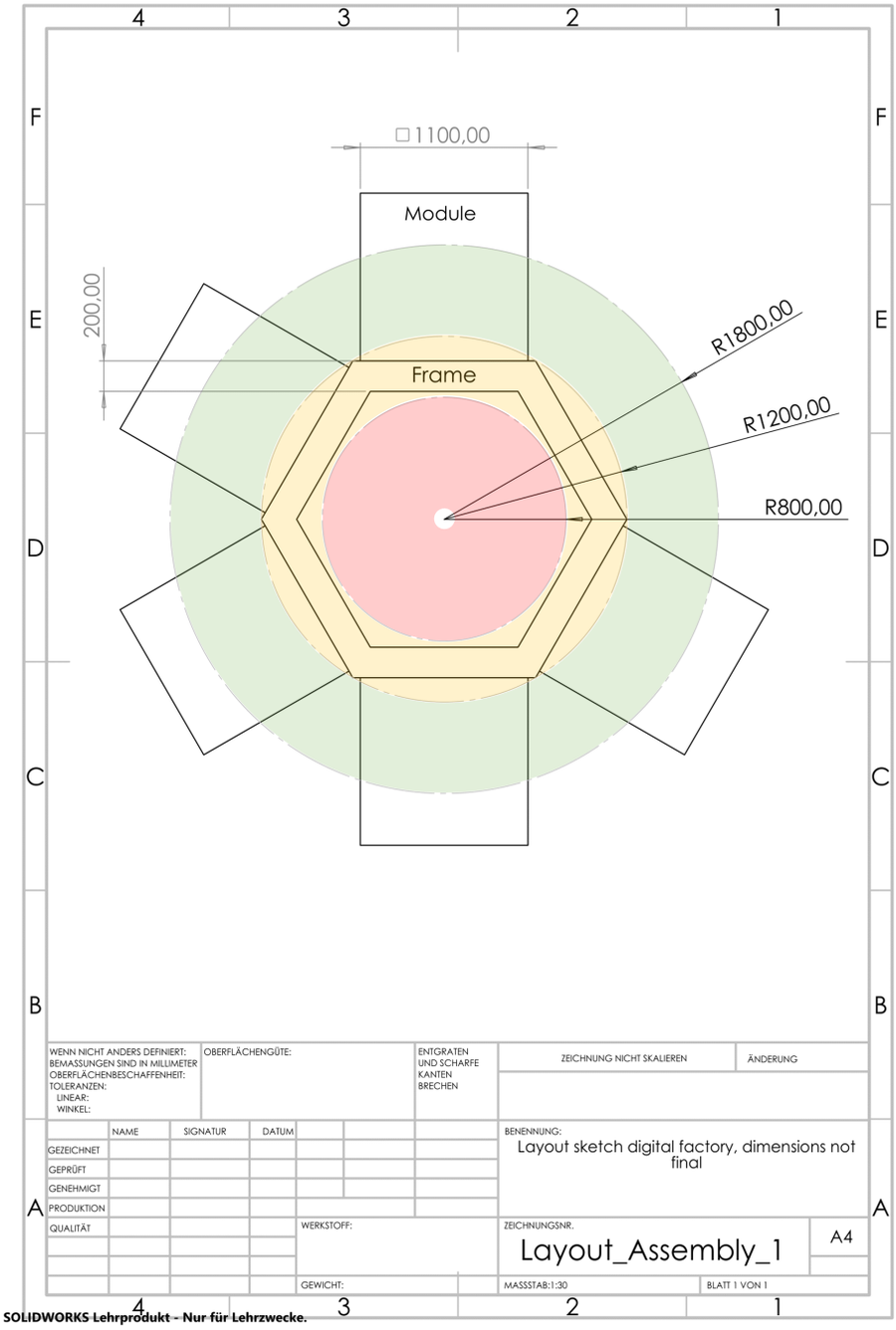
### A.3.1. CAD 3D Model of Workpiece Transfer Unit with Digital Factory

This model offers a visual representation of the Workpiece Transfer Unit's placement within the ideal digital factory layout. It provides insights into how the unit interacts with other components and workspaces in the digital manufacturing environment. Own illustration.



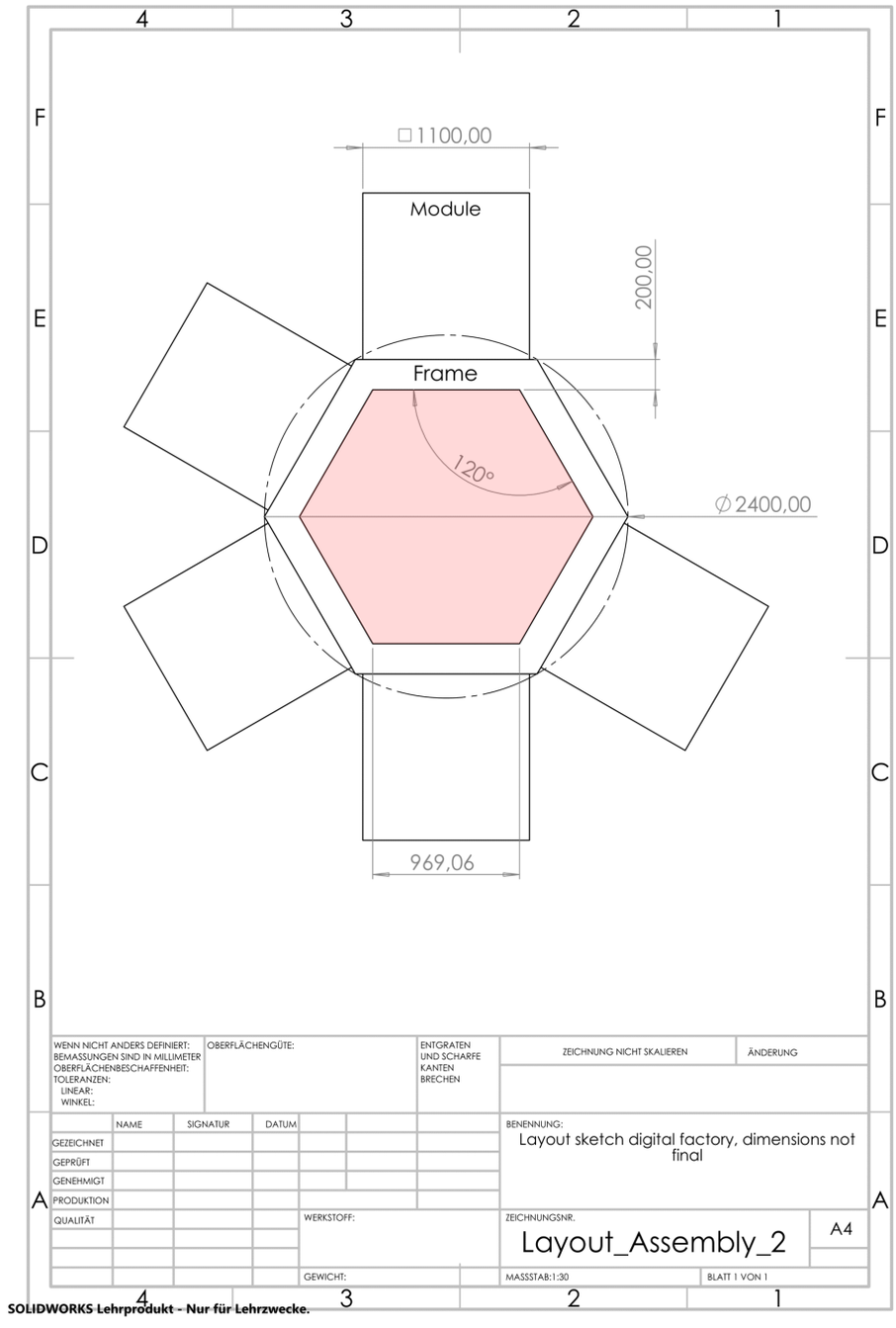
### A.3.2. CAD Drawing of Workspace for Workpiece Transfer Unit

Workspace requirements for the Workpiece Transfer Unit (WTU) in the context of the digital factory architecture, as depicted by a CAD drawing. The illustration highlights the red-outlined mandatory workspace area required for optimal WTU functionality. In addition, it investigates two alternative configurations that provide advantageous improvements to the workspace configuration. Own illustration.



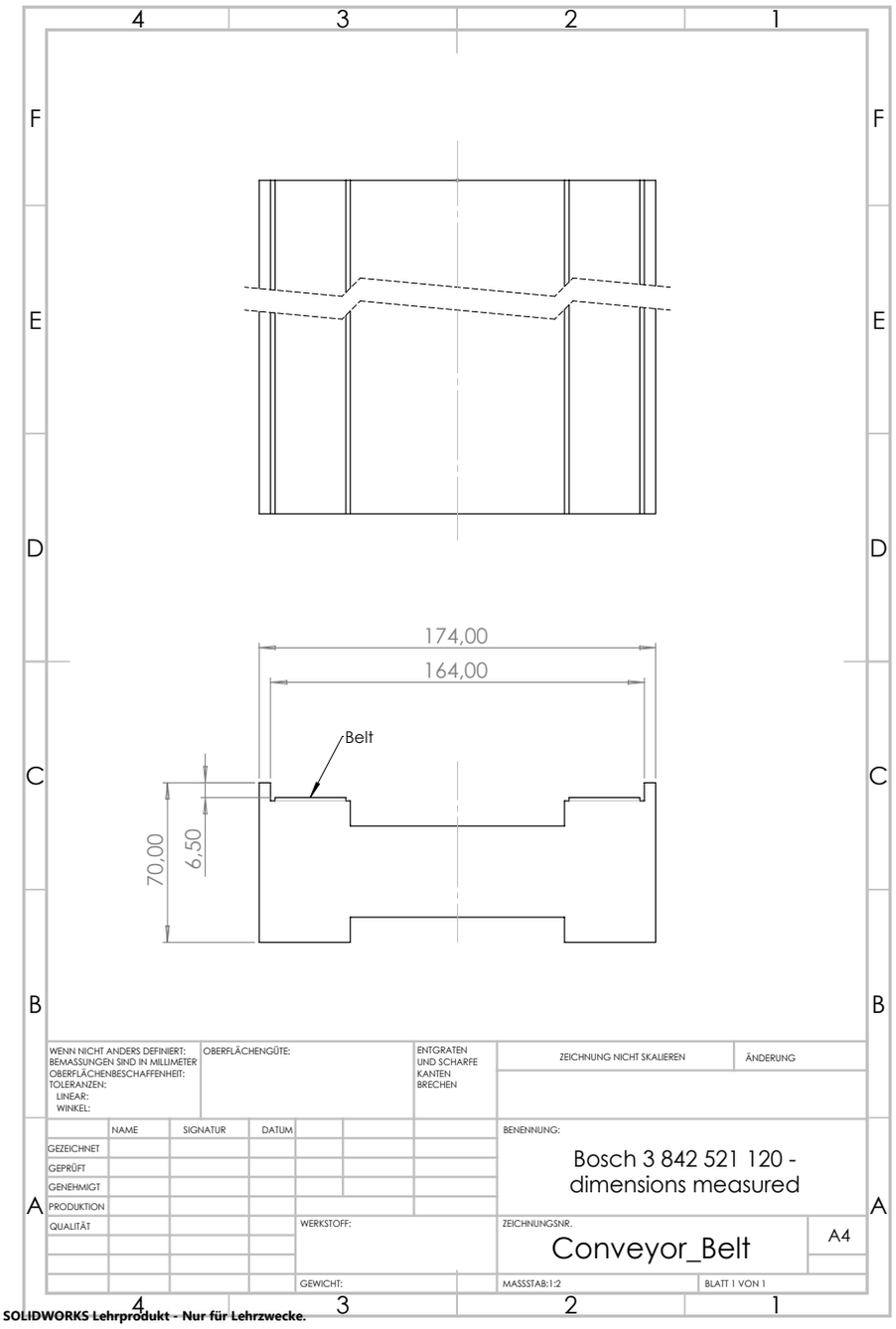
### A.3.3. CAD Drawing of Maximum Area for Workpiece Transfer Unit

CAD drawing showcasing the maximum available area allocated for installing the Workpiece Transfer Unit (WTU) within the planned digital factory. The marked red area signifies the designated space for the WTU. Own Illustration.



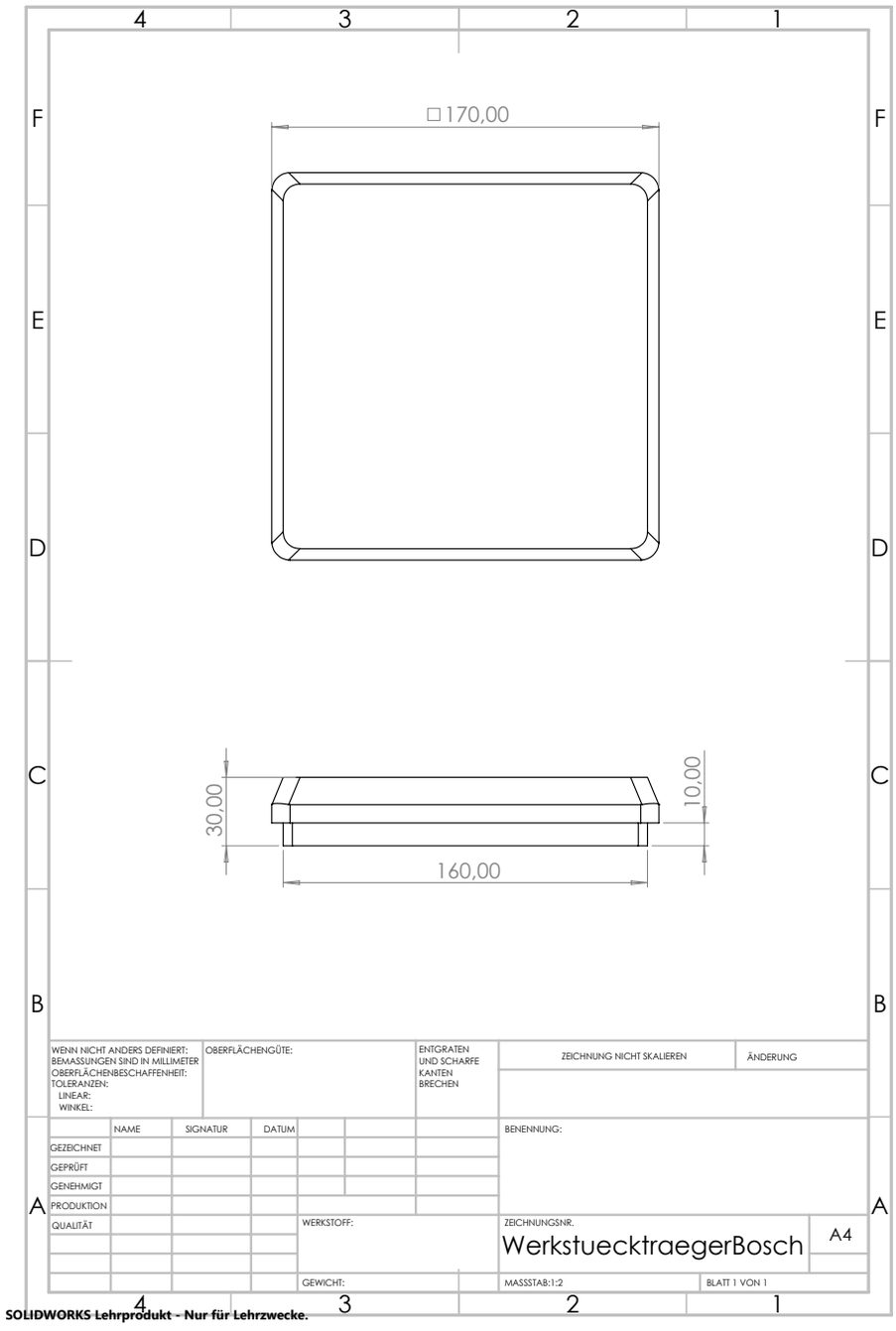
### A.3.4. CAD Drawing of Conveyor Belt

A technical drawing of the conveyor belt utilized in the digital factory is shown. The drawing includes precise measurements, allowing for a comprehensive understanding of the conveyor belt's specifications. Own Illustration.



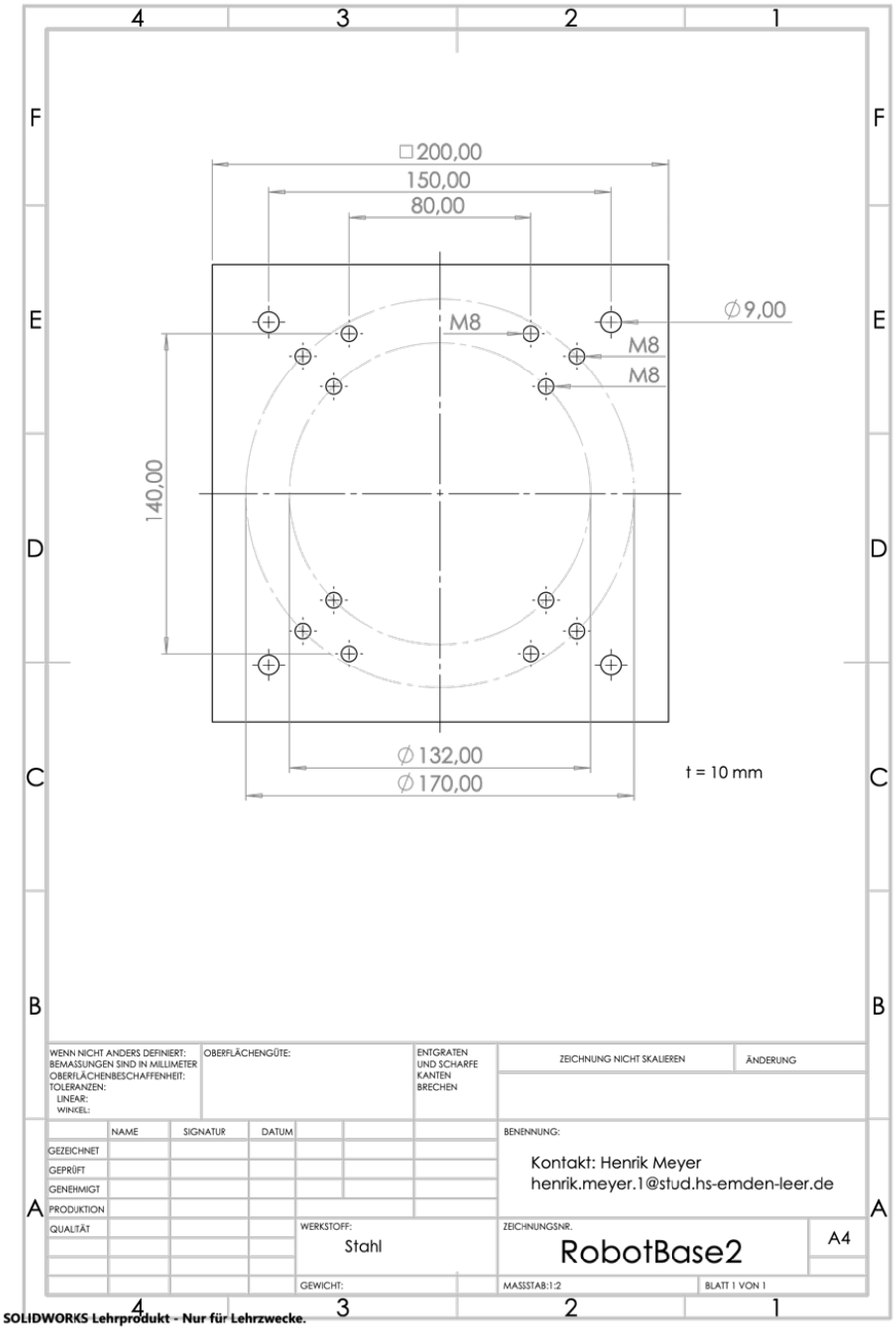
## A.3.5. CAD Drawing of Workpiece

A technical drawing of the workpiece slated for transportation within the digital factory is presented. This drawing includes precise measurements, offering essential insights into the workpiece's physical attributes. Own Illustration.



## A.3.6. CAD Drawing of Adapter Plate

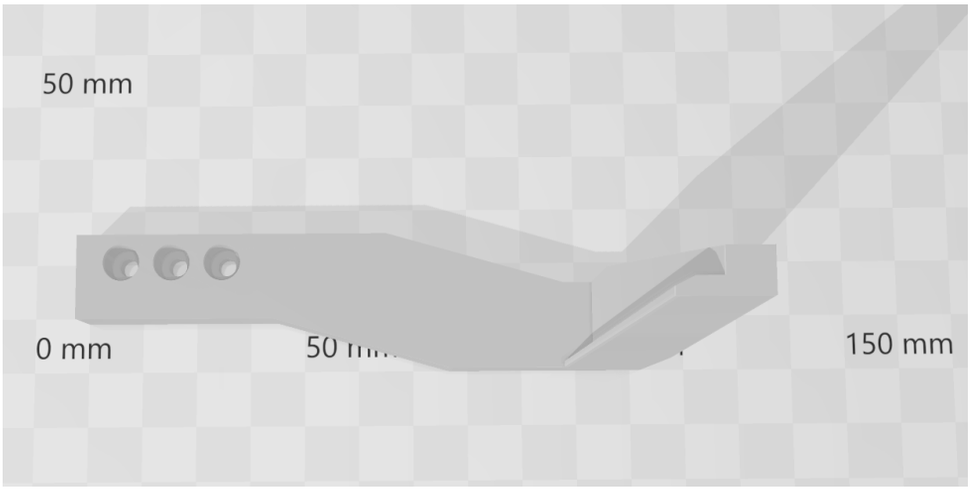
Technical drawing of the adapter plate utilized for mounting the UR5e robotic arm is presented. This drawing offers comprehensive details about the adapter plate's design and specifications. It serves as a valuable reference for the assembly and integration of the adapter plate with the robotic system. Own Illustration.



# A.4. SMC of 3D Print Model for Gripper

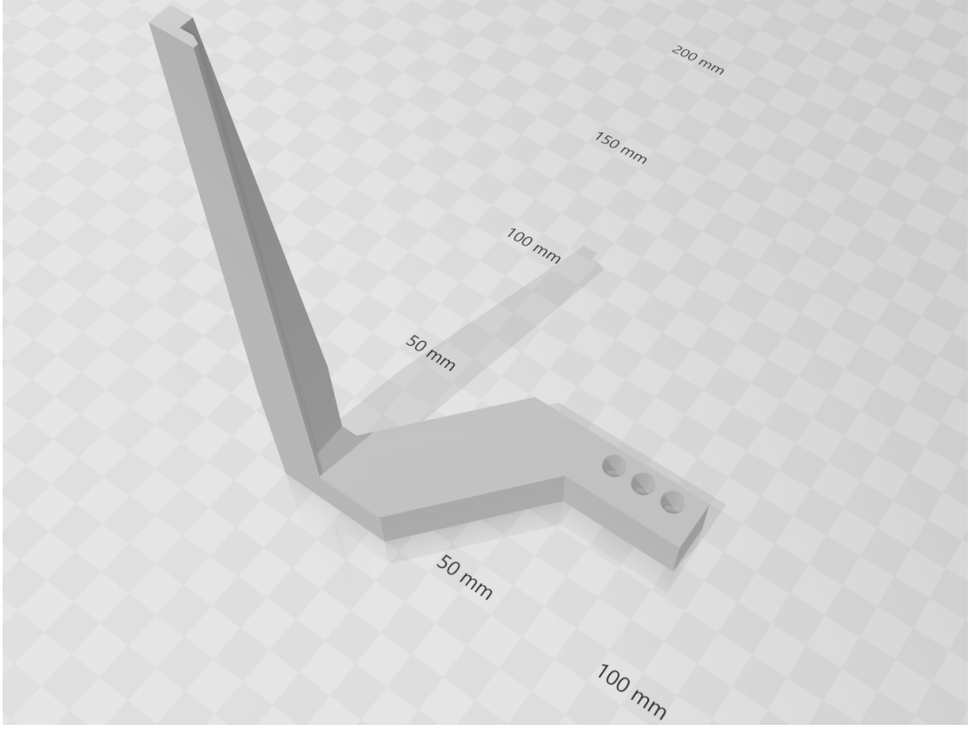
## A.4.1. Gripper Left

The "Gripper Left" component is an integral part of the 3D-printed model for the gripper assembly. It plays a crucial role in the gripping mechanism, contributing to the assembly's overall functionality. Own Illustration.




## A.4.2. Gripper Right

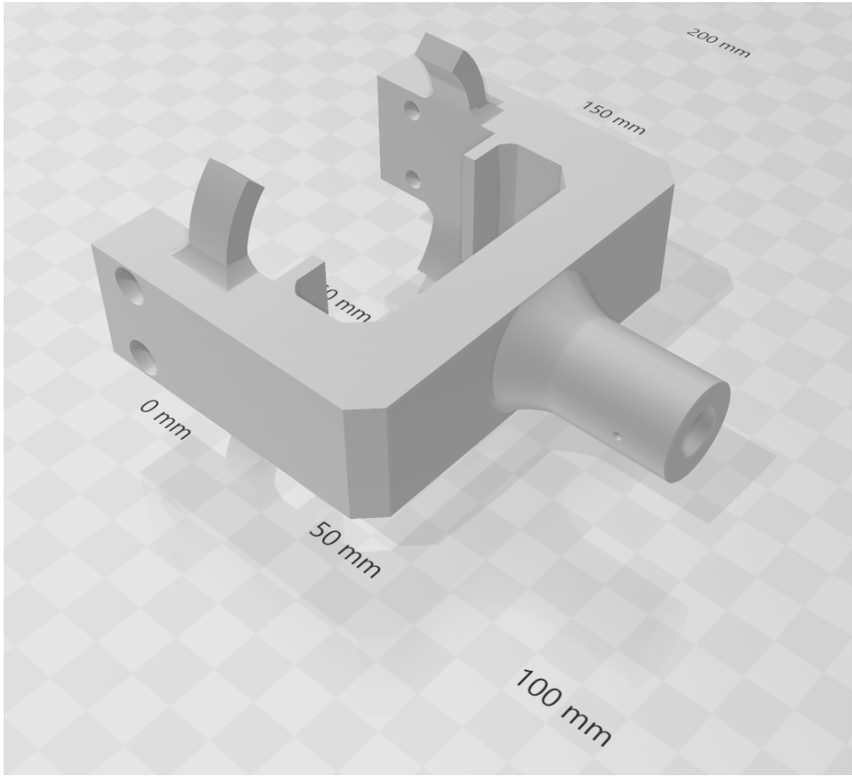
The "Gripper Right" component, another key element of the 3D printed gripper model, complements the "Gripper Left" to form a functional gripper assembly. Together, they enable precise and controlled gripping actions. Own Illustration.



# A.5. SMC of 3D Print Model for Setting Base Tool

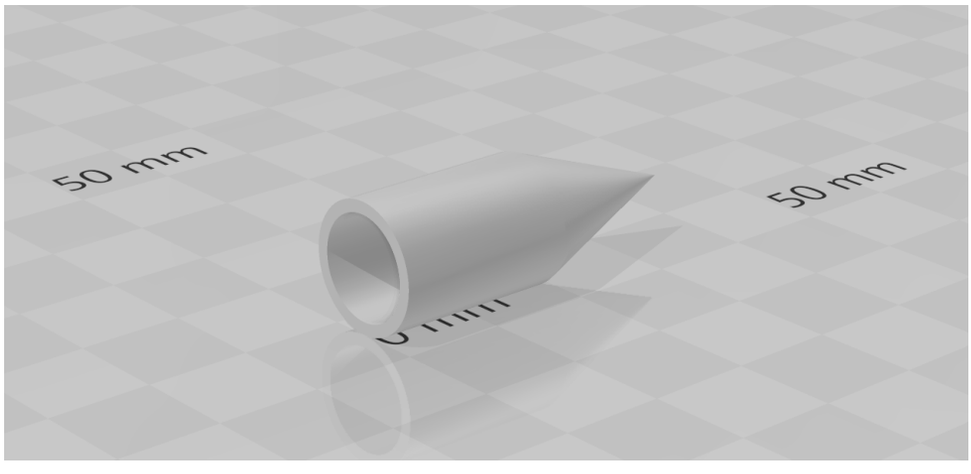
## A.5.1. Holder

The holder is the main component for the Setting Base Tool; it will be attached to the Robotiq Hand-E gripper when the setting new plane is required. As well as the rod, which required a firm placement into the holder. Own Illustration.




## A.5.2. Tip

The "Holder" is a main component of the Setting Base Tool, serving as the primary interface for connecting with the Robotiq Hand-E gripper when adjustments to a new plane are necessary. This component also plays a crucial role in securely anchoring the rod, ensuring its stable placement during operations. Own Illustration.



# B. Implementation

## B.1. Configuration of UR5e Robot Programming

```
BeforeStart
  Gripper Activate
  Base_varBase_1_const
  Call Homee
  Read OPC UA server: start
  Read OPC UA server: isBusy
  Read OPC UA server: service
  Read OPC UA server: isUnderService
  Read OPC UA server: pick_id
  Read OPC UA server: pick_dir
  Read OPC UA server: place_id
  Read OPC UA server: place_dir
  Read OPC UA server: atServicePosition
Robot Program
  Call Homee
  Call openGripper
  Call Service
  Call wait_for_start
  Call read_pos
  Switch pick_dir
    Case 1
      Base_varBase_1_const
    Case 2
      Base_varBase_2_const
    Case 3
      Base_varBase_3_const
    Case 4
      Base_varBase_4_const
    Case 5
      Base_varBase_5_const
    Case 6
      Base_varBase_6_const
    Default Case
      Halt
  Call Homee
  Switch pick_id
    Case 1
      Call goPick_1
    Case 2
      Call goPick_2
    Case 101
      'test'
    Default Case
      Halt
  Call Homee
```

```
  Switch place_dir
    Case 1
      Base_varBase_1_const
    Case 2
      Base_varBase_2_const
    Case 3
      Base_varBase_3_const
    Case 4
      Base_varBase_4_const
    Case 5
      Base_varBase_5_const
    Case 6
      Base_varBase_6_const
    Default Case
      Halt
  Call Homee
  Switch place_id
    Case 1
      Call goPlace_1
    Case 2
      Call goPlace_2
    Case 101
      'test'
    Default Case
      Halt
Service
  ServicePose0
  Write OPC UA server: atServicePosition
  Read OPC UA server: service
  isUnderService False
  Write OPC UA server: isUnderService
  Switch service
    Case 0
      'no service needed'
    Case 1
      ServicePose1
      Write OPC UA server: atServicePosition
      service0
      Write OPC UA server: service
      isUnderService True
      Write OPC UA server: isUnderService
      Base_varBase_1_const
      MoveJ
        Waypoint_1
      Halt
    Case 2
      ServicePose2
      Write OPC UA server: atServicePosition
      service0
      Write OPC UA server: service
      isUnderService True
      Write OPC UA server: isUnderService
      Base_varBase_2_const
      MoveJ
```

```
        Waypoint_1
      Halt
    Case 3
      ServicePose3
      Write OPC UA server: atServicePosition
      service0
      Write OPC UA server: service
      isUnderService True
      Write OPC UA server: isUnderService
      Base_varBase_3_const
      MoveJ
        Waypoint_1
      Halt
    Case 4
      ServicePose4
      Write OPC UA server: atServicePosition
      service0
      Write OPC UA server: service
      isUnderService True
      Write OPC UA server: isUnderService
      Base_varBase_4_const
      MoveJ
        Waypoint_1
      Halt
    Case 5
      ServicePose5
      Write OPC UA server: atServicePosition
      service0
      Write OPC UA server: service
      isUnderService True
      Write OPC UA server: isUnderService
      Base_varBase_5_const
      MoveJ
        Waypoint_1
      Halt
    Case 6
      ServicePose6
      Write OPC UA server: atServicePosition
      service0
      Write OPC UA server: service
      isUnderService True
      Write OPC UA server: isUnderService
      Base_varBase_6_const
      MoveJ
        Waypoint_1
      Halt
Homee
  MoveJ
    posHome
openGripper
  Gripper Open (1)
wait_for_start
  isBusy False
  Write OPC UA server: isBusy
```

```
  Read OPC UA server: start
  Loop  True start
    Wait: 0.1
    Read OPC UA server: start
  start False
  Write OPC UA server: start
  isBusy True
  Write OPC UA server: isBusy
read_pos
  pick_id0
  pick_dir0
  place_id0
  place_dir0
  Read OPC UA server: pick_id
  Read OPC UA server: pick_dir
  Read OPC UA server: place_id
  Read OPC UA server: place_dir
goPick_1
  MoveP
    posTo_1_down
  MoveL
    pos_1_down
  Gripper Move45% (1)
  MoveL
    pos_1_up
  MoveL
    posTo_1_up
goPlace_1
  MoveL
    posTo_1_up
  MoveL
    pos_1_up
  MoveL
    pos_1_down
  Gripper Open (1)
  MoveL
    posTo_1_down
goPick_2
  MoveP
    posTo_2_down
  MoveL
    pos_2_down
  Gripper Move45% (1)
  MoveL
    pos_2_up
  MoveL
    posTo_2_up
goPlace_2
  MoveP
    posTo_2_up
  MoveL
    pos_2_up
  MoveL
    pos_2_down
```

```
Gripper Open (1)
MoveP
  posTo_2_down
```

## B.2. OPC UA Server - Based on AAS

```
# This is OPC UA Server - Based on AAS
import asyncio
import logging
import queue

from asyncua import Server, ua
from asyncua.common.methods import uamethod

# Import of Client functions. This Server is connected to Server of robot control only via
                                                    the Client
from OPCUA_Client_Communication_with_OPCUA_Server_UR5e import read_var, read_pos,
                                                   write_start, write_service, write_pos

# Declaring nodeID of UR5e robot control's OPC UA server for communication
nodeID_start = "ns=2;s=start"
nodeID_isBusy = "ns=2;s=isBusy"
nodeID_service_id = "ns=2;s=service"
nodeID_isUnderService = "ns=2;s=isUnderService"
nodeID_pick_id = "ns=2;s=pick_id"
nodeID_pick_dir = "ns=2;s=pick_dir"
nodeID_place_id = "ns=2;s=place_id"
nodeID_place_dir = "ns=2;s=place_dir"

# Fifo queue for storing pick and place requests, pap_quqeue_length defines the max amount
                                                    of requests stored
pap_queue_length = 3
pap_queue = queue.Queue(pap_queue_length)

# Functions basically just write variables that are used in the robot program to start
                                                   certain processes
# and/or read variables to monitor its state. Connection realized using Client functions.

# Declare fucntion of service operation
# Put robot into one of its service positions/programs, return True when program runs
                                                   through
@uamethod
async def service(nodeID, service_id):
    await asyncio.create_task(write_service(nodeID_service_id, service_id))
    return True

# Declare function of checking number of queue
# Queueing pick and place requests, return False when queue already full, True when it's
                                                   not
@uamethod
async def pick_and_place(nodeID, pick_id, pick_dir, place_id, place_dir):
    if pap_queue.full():
        return False
    else:
        pap_queue.put([pick_id, pick_dir, place_id, place_dir])
        return True

# Declare function of pick and place operation
# Start certain robot process
async def pap_action(pick_id, pick_dir, place_id, place_dir):

    # Set id of module and its direction, start Process
    # Pick
    await asyncio.create_task(write_pos(nodeID_pick_id, nodeID_pick_dir, pick_id, pick_dir
                                                    ))
    await asyncio.create_task(read_pos(nodeID_pick_id, nodeID_pick_dir))
    # Place
    await asyncio.create_task(write_pos(nodeID_place_id, nodeID_place_dir, place_id,
                                                   place_dir))
    await asyncio.create_task(read_pos(nodeID_place_id, nodeID_place_dir))
    # Start
    await asyncio.create_task(write_start(nodeID_start, True))
```

```
async def main():
    logger = logging.getLogger(__name__)

    # Setup server
    server = Server()
    await server.init()
    server.set_endpoint("opc.tcp://192.168.158.62:4840/AAS")
    server.set_server_name("I40 Workpiece Transfer Unit")

    # Import the AAS of the Workpiece Transfer Unit in XML format
    aas_nodes = await server.import_xml("AAS_Workpiece_Transfer_Unit.xml")
    nodes = [server.get_node(node) for node in aas_nodes]

    # Get node from AAS of Workpiece Trasnfer Unit in XML format and link method for
                                                function of checking number of queue
    pick_and_place_method_from_xml = server.get_node("ns=3;i=1606")
    server.link_method(pick_and_place_method_from_xml, pick_and_place)

    # Get node from AAS of Workpiece Trasnfer Unit in XML format and link method for
                                                function of service
    service_method_from_xml = server.get_node("ns=3;i=1630")
    server.link_method(service_method_from_xml, service)

    # Get nodes from AAS of the Workpiece Transfer Unit in XML format for various
                                                attributes
    is_busy_node_from_xml = server.get_node("ns=3;i=1639")
    is_under_service_node_from_xml = server.get_node("ns=3;i=1646")
    is_at_service_position_node_from_xml = server.get_node("ns=3;i=1653")
    number_of_queue_node_from_xml = server.get_node("ns=3;i=1660")
    current_picking_direction_node_from_xml = server.get_node("ns=3;i=1667")
    current_placing_direction_node_from_xml = server.get_node("ns=3;i=1674")
    current_picking_module_id_node_from_xml = server.get_node("ns=3;i=1681")
    current_placing_module_id_node_from_xml = server.get_node("ns=3;i=1688")

    # Setup namespace
    uri = "https://github.com/loukjeab"
    idx = await server.register_namespace(uri)

    # Create root node for upcoming functions, variables
    objects = server.nodes.objects

    # Prepare arguments for methods
    # explanation for structure will be depicted only one time for first argument

    # Pick and place operation

    pick_id = ua.Argument()  # Implementation as a argument
    pick_id.Name = "pick_id"  # Display name
    pick_id.DataType = ua.NodeId(ua.ObjectIds.Int32)  # Data type
    pick_id.ValueRank = -1  # Amount of array dimensions (-1 equals scalar value)
    pick_id.ArrayDimensions = []  # amount of values in each array dimension
    pick_id.Description = ua.LocalizedText("ID of the module for picking")  # Display
                                                explanation

    pick_dir = ua.Argument()
    pick_dir.Name = "pick_dir"
    pick_dir.DataType = ua.NodeId(ua.ObjectIds.Int32)
    pick_dir.ValueRank = -1
    pick_dir.ArrayDimensions = []
    pick_dir.Description = ua.LocalizedText("Direction of the direction for picking")

    place_id = ua.Argument()
    place_id.Name = "place_id"
    place_id.DataType = ua.NodeId(ua.ObjectIds.Int32)
    place_id.ValueRank = -1
    place_id.ArrayDimensions = []
    place_id.Description = ua.LocalizedText("ID of the module for placing")
```

```
place_dir = ua.Argument()
place_dir.Name = "place_dir"
place_dir.DataType = ua.NodeId(ua.ObjectIds.Int32)
place_dir.ValueRank = -1
place_dir.ArrayDimensions = []
place_dir.Description = ua.LocalizedText("Direction of the direction for placing")

result_pap = ua.Argument()
result_pap.Name = "result_pap"
result_pap.DataType = ua.NodeId(ua.ObjectIds.Boolean)
result_pap.ValueRank = -1
result_pap.ArrayDimensions = []
result_pap.Description = ua.LocalizedText("Call successfull")

# service operation

service_id = ua.Argument()
service_id.Name = "service_id"
service_id.DataType = ua.DataType = ua.NodeId(ua.ObjectIds.Int32)
service_id.ValueRank = -1
service_id.ArrayDimensions = []
service_id.Description = ua.LocalizedText("Service Positions: 0 = none, 1-6 =
                                          Maintanance Positions")

result_s = ua.Argument()
result_s.Name = "result_s"
result_s.DataType = ua.NodeId(ua.ObjectIds.Boolean)
result_s.ValueRank = -1
result_s.ArrayDimensions = []
result_s.Description = ua.LocalizedText("Call successfull")

# Populating address space
await objects.add_method(idx, "pick_and_place", pick_and_place, [pick_id, pick_dir,
                                          place_id, place_dir],[result_pap])
await objects.add_method(idx, "service", service, [service_id], [result_s])


# Running Server
logger.info("Starting Server!")
async with server:
    while True:

# Read/update variables from UR5e OPC UA server
robot_busy = await read_var(nodeID_isBusy)
is_under_service = await read_var(nodeID_isUnderService)
service_position = await read_var(nodeID_atServicePosition)
current_pick_direction = await read_var(nodeID_pick_dir)
current_place_direction = await read_var(nodeID_place_dir)
current_pick_module_id = await read_var(nodeID_pick_id)
current_place_module_id = await read_var(nodeID_place_id)

# Send pick and place instruction if one is in the queue
if pap_queue.qsize() > 0 and not robot_busy:
    instruction = pap_queue.get()
    await asyncio.create_task(pap_action(instruction[0], instruction[1], instruction[2
                                          ], instruction[3]))

# Write values back to AAS-based OPC UA server
await is_busy_node_from_xml.write_value(robot_busy)
await number_of_queue_node_from_xml.write_value(pap_queue.qsize())
await is_at_service_position_node_from_xml.write_value(service_position)
await is_under_service_node_from_xml.write_value(is_under_service)
await current_picking_direction_node_from_xml.write_value(current_pick_direction)
await current_placing_direction_node_from_xml.write_value(current_place_direction)
await current_picking_module_id_node_from_xml.write_value(current_pick_module_id)
await current_placing_module_id_node_from_xml.write_value(current_place_module_id)

        await asyncio.sleep(0.5)
```

```
if __name__ == "__main__":
    asyncio.run(main())
```

## B.3. OPC UA Client - Communication with UR5e OPC UA Server

```
# This is OPC UA Client - Communication with UR5e OPC UA Server
import asyncio
import logging
import concurrent.futures
from asyncua import Client, ua

#logging.basicConfig(level=logging.INFO)
_logger = logging.getLogger(__name__)

# Url for connection to UR5 with username and password (both have to be replaced with
                                                    actual access data)
url_ur5 = "opc.tcp://192.168.158.34:4840"


# Client functions used to read and write variables from/to OPC UA Server in robot control

# Read single variable of nodeID, return this value
async def read_var(nodeID):
    async with Client(url=url_ur5) as client:
        node = client.get_node(nodeID)
        value = await node.read_value()
        return value


# Write single value to variable with nodeID
async def write_service(nodeID, value):
    async with Client(url=url_ur5) as client:
        node = client.get_node(nodeID)
        await node.write_value(value, ua.VariantType.Int32)


# Write single value to variable with nodeID
async def write_start(nodeID, value):
    async with Client(url=url_ur5) as client:
        node = client.get_node(nodeID)
        await node.write_value(value, ua.VariantType.Boolean)


# Write two values, here position represented by current id and dir value
async def write_pos(nodeID_id, nodeID_dir, id, dir):
    async with Client(url=url_ur5) as client:
        node = client.get_node(nodeID_id)
        await node.write_value(id, ua.VariantType.Int32)

        node = client.get_node(nodeID_dir)
        await node.write_value(dir, ua.VariantType.Int32)


# Read two values, here position represented by current id and dir value, return these
                                                    values
async def read_pos(nodeID_id, nodeID_dir):
    async with Client(url=url_ur5) as client:
        node = client.get_node(nodeID_id)
        value_1 = await node.read_value()

        node = client.get_node(nodeID_dir)
        value_2 = await node.read_value()

        return [value_1, value_2]


async def main():
    while True:
        client = Client(url=url_ur5, timeout=60, watchdog_interval=60)
        try:
            # Running Client
```

```
            async with client:
                logger.info("Starting Client!")

            while True:
                await asyncio.sleep(0.5)
                await client.check_connection()
        except:
            _logger.warning("Reconnecting Client in 1 second")
            await asyncio.sleep(1)


if __name__ == "__main__":
    logging.basicConfig(level=logging.WARNING)
    asyncio.run(main())
```