

Jouni Kokkonen

OPTIMIZING THE VIRTUAL COMMISSIONING PROCESS FOR PANEL REPAIRING LINE

Master of Science Thesis
Faculty of Engineering
and Natural Sciences
Examiners:
Jose Luis Martinez Lastra
Luis Enrique Gonzalez Moctezuma
October 2025

ABSTRACT

Jouni Kokkonen : Optimizing the virtual commissioning process for panel repairing line
Master of Science Thesis
Tampere University
Master's Degree Programme in Automation Engineering
October 2025

The aim is to make deliveries of large production lines higher quality and more efficient with the help of Industry 4.0 innovations. One of these innovations is virtual commissioning, which enables testing of the production line in a virtual environment even before the actual commissioning. This process can be used to reduce delivery related risks and commissioning time and provide transparency to the customer right from the beginning of the project. However, virtual commissioning is usually carried out with manual monitoring, so it is not particularly comprehensive, and does not offer opportunities for automatically repeatable test cases.

The purpose of this thesis was to design an efficient and standardized workflow for virtual commissioning, and to create a systematic and efficient validation model for the PLC software. In addition, the thesis investigated what the biggest challenges of virtual commissioning are in terms of reliability, and what other potential benefits it offers to the customer and supplier.

The testbed in this work was the Raute R7 panel repairing line's infeed and outfeed end, which were customized to suit the customer's facilities and needs. A virtual environment consisting of a 3D model, a behaviour model and an emulated PLC was built in the Siemens environment. The work explores various testing methods and tools, and based on them, a model best suited to the use case is created.

The implementation phase included the creation of a behavioural model and integrating it with the PLC program and the kinematic model. In the behavioural model, the couplings to the emulated PLC and the kinematic model had to be verified. In addition, the functionality of the virtual sensors, actuators and drive telegrams were validated. These steps ensured that the virtual commissioning setup accurately represented the real system.

The result was a standard model for creating a virtual commissioning environment for Siemens environment, using Software-in-the-Loop (SiL) configuration. In addition, a systematic five-step PLC validation model was developed, which allows validation to be carried out from individual program blocks all the way to the validation of customer requirements and capacity runs.

The study also assessed the reliability of the virtual model in commissioning and pointed out shortcomings where the virtual environment does not fully correspond to the operation of the real production line. In the case of the testbed, for example, shortcomings were observed in the functionality of the suction cup and the modelling of flexible materials and components. In addition to reducing commission time, other benefits of virtual commissioning were identified, such as the possibility for the commissioning engineers to familiarize themselves with the line in advance and several transparencies that are essential for the customer.

Keywords: Virtual commissioning, PLC software validation, Panel repairing line, Virtualization

The originality of this thesis has been verified using the Turnitin Originality Check service.

TIIVISTELMÄ

Jouni Kokkonen : Virtuaalisen käyttöönoton optimointi paneelinkorjauslinjalle
Diplomityö
Tampereen yliopisto
Automaatiotekniikan diplomi-insinöörin tutkinto-ohjelma
Lokakuu 2025

Suurten tuotantolinjojen toimituksista pyritään tekemään laadukkaampia ja tehokkaampia Industry 4.0 -innovaatioiden avulla. Yksi näistä innovaatioista on virtuaalinen käyttöönotto, joka mahdollistaa tuotantolinjan testaamisen virtuaalisessa ympäristössä jo ennen varsinaista käyttöönottoa. Tämän prosessin avulla voidaan vähentää toimitukseen liittyviä riskejä ja käyttöönottoaikaa sekä tarjota tilaajalle läpinäkyvyyttä jo projektin alkumetreiltä lähtien. Tavanomaisesti virtuaalinen käyttöönotto toteutetaan kuitenkin manuaalisesti monitoroimalla, joten se ei sellaisenaan ole erityisen kattava, eikä tarjoa mahdollisuuksia automaattisesti toistettaville testitapauksille.

Tämän opinnäytetyön tarkoitus oli suunnitella tehokas ja standardisoitu työnkulku virtuaaliselle käyttöönotolle, sekä luoda systemaattinen ja tehokas validointimalli PLC-ohjelmalle. Lisäksi opinnäytetyössä selvitettiin, mitkä ovat virtuaalisen käyttöönoton suurimmat haasteet luotettavuuden kannalta, ja mitä muita mahdollisia hyötyjä se tarjoaa tilaajalle ja toimittajalle.

Testipenkinä tässä työssä toimi Raute R7 paneelinkorjauslinjan syöttö- ja purkupää, jotka ovat räätälöity asiakkaan tiloihin ja tarpeisiin sopiviksi. Virtuaalinen ympäristö, joka koostuu 3D-mallista, käyttäytymismallista ja emuloidusta PLC:stä rakennettiin Siemens ympäristöön. Työssä pureudutaan erilaisiin testausmenetelmiin ja -työkaluihin, ja niiden pohjalta laaditaan käyttötapukseen parhaiten soveltuva malli.

Toteutusvaiheeseen sisältyi käyttäytymismallin luominen sekä sen integroiminen PLC-ohjelmaan ja kinematiikkamalliin. Lisäksi varmistettiin kytkennät käyttäytymismallin, kinematiikkamallin ja emuloidun PLC:n välillä. Prosessiin kuului myös virtuaalisten antureiden, toimilaitteiden ja käyttötelegammien toiminnallisuuden validointi. Näillä toimenpiteillä varmistettiin, että virtuaalisen käyttöönoton ympäristö vastasi mahdollisimman tarkasti todellista järjestelmää.

Tuloksena laadittiin Siemens-ympäristöön soveltuva, Software-in-the-Loop (SiL) -kokoonpanolla toimiva standardimalli virtuaalisen käyttöönottoympäristön luomiseksi. Lisäksi kehitettiin systemaattinen viisivaiheinen PLC-validointimalli, jonka avulla validointi toteutetaan yksittäisistä ohjelmalohkoista aina asiakasvaatimusten ja kapasiteettiajojen validointiin asti.

Työssä arvioitiin myös virtuaalisen mallin luotettavuutta käyttöönotoissa sekä nostettiin esille epäkohtia, joissa virtuaalinen ympäristö ei täysin vastaa todellisen linjaston toimintaa. Testipenkin tapauksessa havaittiin esimerkiksi epäkohtia imutarttujan toiminnallisuudessa ja taipuisien materiaalien ja -komponenttien mallinnuksessa. Käyttöönottoajan lyhentämisen lisäksi tunnistettiin muitakin virtuaalisen käyttöönoton hyötyjä, kuten käyttöönottajän mahdollisuus perehtyä linjaan etukäteen sekä useita tilaajalle oleellisia läpinäkyvyystekijöitä.

Avainsanat: Virtuaalinen käyttöönotto, PLC-ohjelman validointi, Paneelinkorjauslinja, Virtualisointi

Tämän julkaisun alkuperäisyys on tarkastettu Turnitin Originality Check -ohjelmalla.

USE OF AI IN THESIS

I have utilised AI tools in my thesis:

- No
- Yes

The AI tools utilised in my thesis and their purposes are described below:

Names and versions of AI tools: [List all the AI applications and their versions used during your thesis process]

Purpose of using AI tools: [Provide a detailed explanation of the purpose and application of AI tools during your thesis process]

Sections where AI tools were used: [List all the steps and sections of your thesis where AI has been applied during your thesis process]

I acknowledge that I am fully responsible for the entire content of my thesis, including the parts generated by AI, and accept accountability for any violations of ethical standards in publications.

PREFACE

First, I would like to express my sincere gratitude to Professor Jose Martinez Lastra and University Instructor Luis Gonzalez Moctezuma for their support and guidance throughout the thesis process.

I also wish to thank Antti Pennanen, Heikki Korpilahti, Lassi Hyypiä, Tero Myöhänen and Toni Laine from Raute for their valuable assistance and expertise during the project. My thanks also to Pekka Pyhäluoto, Ville Torvinen and Samuli Metsälä from Siemens for their comprehensive support with the Siemens software and simulation tools.

Finally, I would like to thank my friends and family for their support throughout my master's degree.

Lahti, 12 October 2025

Jouni Kokkonen

CONTENTS

USE OF AI IN THESIS	III
INTRODUCTION.....	1
1.1 Background	1
1.2 Problem Statement.....	2
1.3 Research Questions & Objectives	3
1.4 Limitations of the Study	4
1.5 Thesis Outline.....	4
LITERATURE REVIEW.....	6
2.1 Industry 4.0.....	6
2.1.1 Simulation Modelling & Cyber-Physical systems in Industry 4.0....	6
2.2 Virtual Commissioning	7
2.3 PLC Programming and Testbed Environment.....	11
2.3.1 IEC 61131-3.....	11
2.3.2 Siemens Software and Tools	12
2.4 Error Detection in Virtual Commissioning.....	13
2.4.1 Test Case Design	16
2.4.2 Software Validation and Commissioning in Process Control Systems.....	17
2.5 State of the Art in Verifying and Validating PLC Programs.....	18
2.5.1 V-Model Testing.....	19
2.6 State of the Art in Virtual Commissioning.....	20
2.6.1 Virtual-Real Fusion Commissioning Using AR	21
2.6.2 Narrowing Reality Gap in VC	21
2.6.3 Integration Technologies in VC	22
DESIGN AND TESTBED DESCRIPTION	24
3.1 Research Design and Approach	24
3.1.1 Justification of Simulation Environment Selection	26
3.2 Case Study: Panel Repairing Line	26
3.3 Tools and Software.....	29
3.4 PLC Program Validation Plan	30
3.4.1 Justification of Validation Model	32
IMPLEMENTATION	34
4.1 Development of a Standardized Virtual Commission Workflow	34
4.2 Designing a Systematic PLC Program Validation Model.....	37
4.3 Analysis of Testbed Specific Simulation & Validation Tools	37
DISCUSSION AND RESULTS	40
5.1 Standardized Virtual Commissioning Workflow.....	40
5.1.1 3D Model Creation	40
5.1.2 Behaviour Model Creation.....	42

5.2	Systematic PLC Validation Model.....	44
5.3	Evaluation the Reliability of Virtual Commissioning.....	46
5.4	Additional Benefits for Suppliers and Customers	48
5.5	Interpretation of Results.....	49
5.6	Implications for the Studied Testbed.....	51
5.7	Advantages and Limitations of Siemens Environment	52
5.8	Impact on Industrial Practices.....	52
5.9	Practical Recommendations for Virtual Commissioning	53
CONCLUSIONS.....		55
6.1	Summary.....	55
6.2	Future Research.....	56
REFERENCES.....		58

LIST OF FIGURES

<i>Figure 1. The first documented case of "bug being found" [10]</i>	2
<i>Figure 2. Comparison of classical commissioning and VC, adapted from [21] [22]</i>	7
<i>Figure 3. Digitalization approaches [31]</i>	9
<i>Figure 4. Most typical VC configurations, adapted from [32]</i>	10
<i>Figure 5. Software validation steps, adapted from [46]</i>	13
<i>Figure 6. Example debugging status view in different programming languages [47]</i>	15
<i>Figure 7. V-Model design [55]</i>	19
<i>Figure 8. Research approach</i>	25
<i>Figure 9. Siemens virtual environment [62]</i>	25
<i>Figure 10. Raute R7 panel repairing line [63]</i>	27
<i>Figure 11. Loading end of the panel repairing line in NX MCD</i>	28
<i>Figure 12. Unloading end of the panel repairing line in NX MCD</i>	28
<i>Figure 13. Siemens virtual validation network architecture</i>	30
<i>Figure 14. Status- & control word conversion example</i>	35
<i>Figure 15. SIMIT drive telegram example with conversion</i>	36
<i>Figure 16. SIMIT sensor coupling example</i>	36
<i>Figure 17. Example test case implementation for validating a Pythagorean function block using structured text [64]</i>	39
<i>Figure 18. Example of evaluation of the test results of Figure 16 [64]</i>	39
<i>Figure 19. Flowchart of NX MCD model creation steps [65]</i>	41
<i>Figure 20. Behaviour model creation steps</i>	43
<i>Figure 21. PLC software validation model</i>	45
<i>Figure 22. Vacuum gripper correction in a behavior model</i>	47
<i>Figure 23. Veneer sheet in a 3D model</i>	48
<i>Figure 24. Drying rack poles in a 3D model</i>	48
<i>Figure 25. Qualitative comparison of the proposed VC workflow and traditional commissioning</i>	50
<i>Figure 26. Test management in SIMIT Rapid Tester [66]</i>	51
<i>Figure 27. SIMIT HMI chart example</i>	53
<i>Figure 28. ALM process. Applied from [70]</i>	57

LIST OF SYMBOLS AND ABBREVIATIONS

ALM	Application Lifecycle Management
AR	Augmented Reality
CAD	Computer-Aided Design
CPU	Central Processing Unit
DT	Digital Twin
FAT	Factory Acceptance Test
FBD	Function Block Diagram
HiL	Hardware-in-the-Loop
HMI	Human-Machine Interface
IEC	International Electrotechnical Commission
IL	Induction List
LD	Ladder Diagram
MiL	Model-in-the-Loop
NX MCD	Siemens NX Mechatronics Concept Designer Software
NC	Normally Closed
NO	Normally Open
OPC UA	Open Platform Communications Unified Architecture
POU	Program Organization Unit
PLC	Programmable Logic Controller
PLCSIM	Siemens PLC Simulation Software
QA	Quality Assurance
RiL	Reality-in-the-Loop
SAT	Site Acceptance Test
SCADA	Supervisory Control and Data Acquisition
SiL	Software-in-the-Loop
SIMIT	Siemens Simulation Platform
SMG	Siemens Simulation Modeling Generator Software
ST	Structured Text
TIA Portal	Siemens Totally Integrated Automation Portal
UAT	User Acceptance Test
VC	Virtual Commissioning

INTRODUCTION

1.1 Background

In the modern industrial landscape, efficient commissioning and troubleshooting of automated production lines have become critical to maintaining competitive advantage. This is particularly true for custom-tailored solutions, where every project introduces unique challenges. It is a great challenge in large and expensive wood processing lines, which require significant effort during commissioning due to the unique configuration and customer-specific requirements. [1] [2].

Traditional methods of commissioning often involve physical commissioning which might be risky since the large mechanical changes are costly. Also, manual error detection and on-site debugging, which are time-consuming, costly and prone to errors. Furthermore, unanticipated problems during commissioning can lead to delays, increased expenses, and potential risk to equipment damage [1] [2]. Traditional commissioning can consume as much as 15–20% of overall automation project timeline, with nearly two-thirds of this time spent correcting software defects [3]. To address these challenges, virtual commissioning (VC) has emerged as a promising approach, but it requires effort and expertise [4]. By simulating the behaviour of production lines in a virtual environment, it allows engineers to identify and resolve issues before the physical commissioning reducing time, cost and risks. Results of virtual the VC have been convincing. Rockwell Automation reported up to 40% reduction in commission time [5], while Schamp et al. observed a 73% reduction in debugging time when applying VC in their study [6].

Despite the advantages of VC, the adoption of this approach in custom-tailored solutions, such as wood processing lines, faces unique challenges. Selecting the correct tools and methods for implementing VC and validation of programmable logic controller (PLC) software is basis for the whole project [1]. Furthermore, the standardization of behavioural models and kinematic models could significantly improve the simulation efficiency, while also aligning with practices with Industry 4.0 standards [7].

System validation or so-called debugging is the process of identifying and resolving faults in a system. It has always been crucial part of verifying system reliability. Interestingly, the term debugging was popularized in 1947, when a moth was discovered inside the

Mark II Aiken Relay Calculator at Harvard University, between points at relay 70, disrupting its operation. The removal of the moth was logged by the engineering team, including Grace Hopper, as "First actual case of bug being found" (Figure 1). [8] [9].

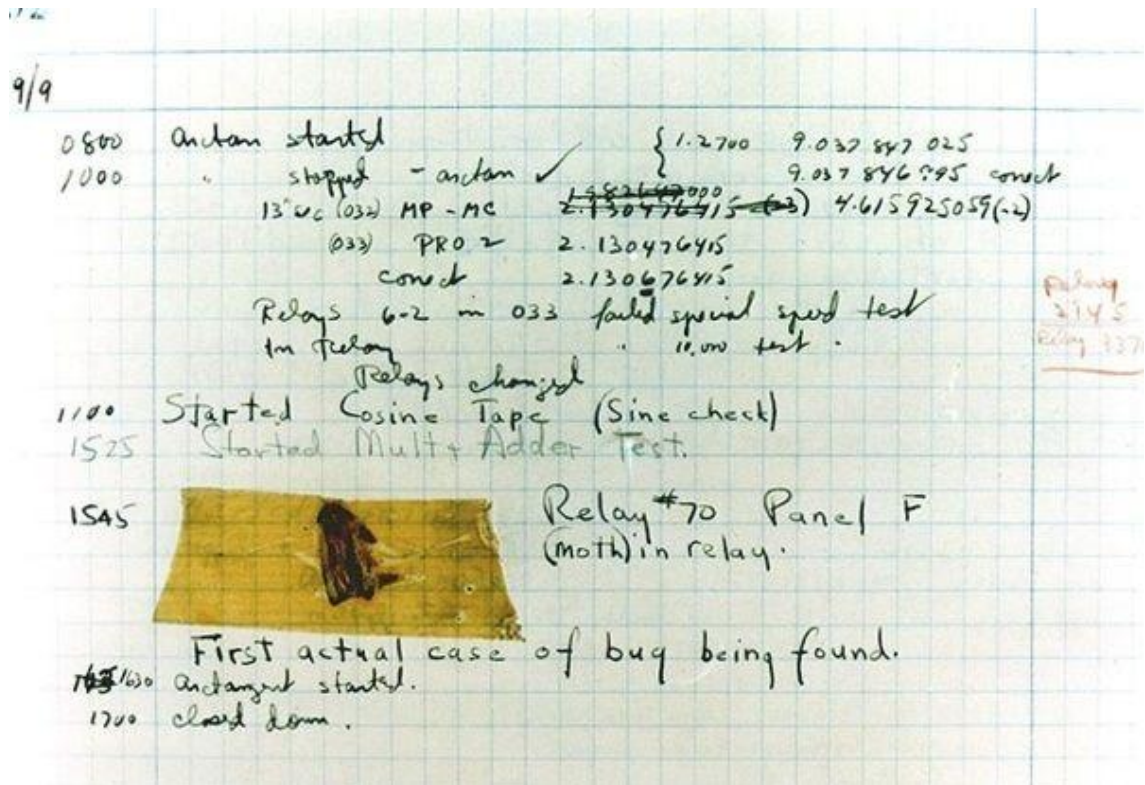


Figure 1. The first documented case of "bug being found" [10]

Since then, debugging has evolved significantly. Nowadays, the complexity of a systems has increased, and many different tools and methods have developed for validation and debugging such as VC.

1.2 Problem Statement

The commissioning and troubleshooting custom-tailored processing lines are time-intensive and risky due to the following key issues [1]:

- **High Complexity of Customization:** Most of the wood processing lines are uniquely tailored to customer requirements, leading to extensive on-site adjustments and debugging.
- **Risky Traditional Methods:** Physical commissioning as the last phase of the project may result delays with error detection and costly changes.

- **Limited Error Detection in PLC Programming:** Bugs in PLC programs are often discovered in the physical commissioning. Some features and for example capacity test can't be tested without physical or virtual environment.
- **Risk During On-Site Tests:** Errors during commissioning pose potential risk of delays and can lead to equipment damage or operational delays [11]. Fines for the delay can be substantial.

These challenges highlight the need for advanced methods to streamline commissioning and ensure the reliability of PLC programs before on-site testing. VC, combined with systematic PLC program validation, offers a promising solution to mitigate these challenges, reduce risks, and improve overall efficiency [1].

Validating the functionality of delivery, testing is a crucial part of ensuring system reliability and user satisfaction. Effective testing plays important role in identifying potential issues before commissioning, thereby minimizing operational risks and project delays. There are important testing phases in the validation process that bring certainty to the customer and supplier, such as factory acceptance testing (FAT) and site acceptance testing (SAT). [12] [13].

1.3 Research Questions & Objectives

The aim of this research is to explore the application of virtual commissioning in identifying, mitigating and preventing errors in PLC programming. By focusing on a panel repairing line in a Siemens virtual environment, the study seeks to identify common challenges and propose practical guidelines to enhance the commissioning process. Additionally, in this research, a standardized and efficient workflow for future software validations using VC will be developed and documented, aiming to establish best practices and guidelines to facilitate consistent and reliable automation software validation process. The research is guided by following questions and objectives:

- How can a standardized and efficient workflow for virtual commissioning be created?
- How can a software validation framework for virtual commissioning be structured to ensure maximum fault detection?
- What are the key challenges in virtual commissioning environment for reliable software validation?
- What additional benefits does virtual commissioning provide to both suppliers and customers beyond reducing commissioning time?

The first objective of this research is to develop a standardized workflow model for future VC projects. The proposed model specifically focuses on project creation, execution and particularly emphasizes reliable simulation of sensors and actuators. A second objective is to create and document a systematic software validation framework, enabling precise and effective validation of PLC programs using VC. In addition, the research aims to identify and analyse potential challenges affecting the reliability of virtual commissioning environment, ensuring these issues receive appropriate attention in future VC implementations. In last objective, the study aims to explore and evaluate the additional benefits VC provides both suppliers and customers in addition to reducing commission time.

1.4 Limitations of the Study

While this thesis develops a standardized VC workflow and systematic validation model for PLC, it is important to acknowledge limitations. The scope of this study is limited to creation of those models but does not include an evaluation of the effectiveness of the results. Originally, the plan included evaluation of the VC against traditional commissioning outcomes. However, due to unexpected project timeline changes the onsite commissioning was postponed beyond the research timeframe.

Second, the VC testbed used in this study is based on Siemens virtual environment and testing tools. These tools provided cross-compatibility between software tools, but also with the actuators selected for the project. However, this creates a vendor lock-in, meaning the proposed workflow is not directly transferable to VC environments built on other platforms. Although the methodology may offer conceptual guidance that could be applied outside the Siemens ecosystem.

1.5 Thesis Outline

This thesis is structured into six main chapters. Chapter I introduces the research topic, providing background of virtual commissioning in the context of Industry 4.0, the challenges it addresses PLC programming for automated systems, and the specific focus on the Siemens environment which is selected to be used for this testbed. It also presents the research problem, questions and objectives.

Chapter II reviews the existing literature, covering the role of simulation modelling in Industry 4.0. The chapter explores key methodologies of making virtual commissioning and PLC program validation. It also examines Siemens virtualization tools and state-of-the-art concepts which are relevant to this study.

Chapter III describes the research design and testbed. It outlines the case study of a panel repairing line, detailing the tools and software. Additionally, this chapter presents the problems need to be considered making the software validation plan.

Chapter IV focuses on the implementation phase, detailing the development of a standardized VC workflow. It outlines the design of a systematic PLC program validation model and analyses tools used for simulation and PLC program validation.

Chapter V presents the results and discussion. It begins by examining the proposed standardized virtual commissioning workflow and the design of a systematic PLC validation model. The chapter evaluates the reliability of VC used testbed as a reference and provides insights into its consistency and practical value.

Finally, Chapter VI concludes the study by summarizing the key findings, discussing limitations and suggesting directions for future research. This chapter highlights the broader impact on advancing virtual commissioning practices.

LITERATURE REVIEW

2.1 Industry 4.0

The Fourth Industrial Revolution, also referred to as Industry 4.0, represents a transformative industrial paradigm that merges digital and physical worlds [14]. Companies that use continuous development models and have high standards in research and development work make themselves more competitive by utilizing the industry 4.0 concept [15]. The key elements in the concept are following [16].

- Additive manufacturing
- Augmented reality
- Autonomous robots
- Big data & Analytics
- Cyber-Physical Systems
- Simulation
- System integration

2.1.1 Simulation Modelling & Cyber-Physical systems in Industry 4.0

Simulation modelling involves replicating a real or virtual process or system to predict or analyze the outcomes of the modeled system or process. By utilizing real-time data, simulation models can accurately represent the real world, encompassing elements such as humans, products, and machines. This allows operators to optimize machine settings in a virtual environment before applying them in the physical world, thereby reducing machine setup times and enhancing quality. Recent advancements in simulation modeling have facilitated the creation of virtual factories, enabling the modeling of manufacturing and other systems. Additionally, incorporating advanced artificial intelligence in process control through simulation allows for autonomous adjustment and self-organization of operational systems. [16].

Cyber-Physical Systems (CPS) are integration of computational algorithms with physical processes, where embedded systems monitor and control physical components via real-time feedback loops. Systems combine actuators, sensors communication networks and control software to enable intelligent, adaptive and autonomous behavior [17]. In a CPS,

sensors and actuators serve as the main components to allow data exchange between the physical- and cyber systems. Sensors collect real-time operational data from machines and environments, while actuators execute control commands generated in the cyber layer. This creates a feedback loop to support precise control, predictive maintenance, and optimization in manufacturing settings. [18].

CPS is foundational to smart manufacturing [19]. It makes a bridge between the physical and digital domains. That enables real-time monitoring, dynamic control and adaptive decision-making by integrating computing, communication and control capabilities tightly, also known as the 3C integration. [18].

2.2 Virtual Commissioning

Virtual commissioning (VC) of production lines is one of the innovations of Industry 4.0, designed to shift troubleshooting to an earlier stage in the commissioning process (Figure 2). In a traditional arrangement, the automation program and the mechanical production line are tested on site. When the mechanical implementation is ready, the automation program is tested with real sensors and actuators. [20].

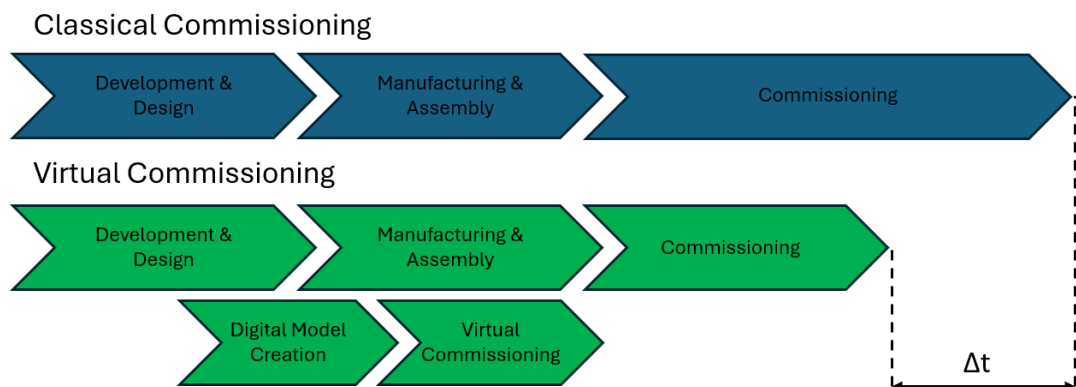


Figure 2. Comparison of classical commissioning and VC, adapted from [21] [22]

Auinger et al. [23] introduced early approach for VC which they referred as “soft-commissioning” in 1999. They tackled time consuming and expensive validation and testing process of pallet transfer system control software. During their experiments, they identified issues such as stopper malfunctions caused by overlooked inverse logic and problems in the material flow.

VC significantly reduces the time required for automation error correction compared to traditional methods [24] [25]. By simulating and testing systems in a virtual environment before physical commissioning, potential issues can be identified and resolved early in the development process [26]. This approach minimizes the costly and time-consuming

error correction and debugging during the physical commissioning. For example, Wipro Pari reported a 40-50% reduction in rework and a 70% decrease in on-site commissioning time by utilizing virtual commissioning. [27].

In the concept of VC, the process is carried out in a computer environment using either a real or soft PLC. The primary goal of this approach is to identify and resolve errors at the early stage. To achieve this, a simulation model of the production line is developed in simulation software, incorporating all relevant components (sensors, motors, valves, etc.). Through the interaction between the automation program and the simulation software, the operation of the machine or production line is virtually monitored, allowing program and design issues and bugs to be identified and corrected in this virtual setup. [20].

Shahim and Møller [28] identified benefits based on interviews of five Danish companies. Their questions concerned effects of applying VC on several automation parameters. A summary of their interviews can be seen in Table 1.

Table 1. Summary of VC benefits according to interviews by Shahim and Møller, adapted from [28]

Automation Parameter	Benefits Reported by Interviewees
On-site work	Significant reduction 50% in required on-site manpower.
Risk of Delays and Interruptions	Less interruptions and smoother project flow.
Software Quality	Improved stability and fewer errors due extensive testing before on-site implementation.
System Debugging Control	Enhanced control and efficiency in debugging, resulting higher software quality and lower rework costs.
Deadline Management	Better software quality increases the change of staying on schedule.
Reusability of Virtual Models	Models developed for VC can be reused, saving resources and time for future challenges.

Often when discussing about virtualization and digitalization, the term digital twin comes across, digital twin, digital shadow and digital model are often used as synonyms. As shown in Figure 3, there is a different digital integration based on the degree of data connectivity between the physical and digital representations. [29] [30].

- A digital model refers to a digital representation of a physical object, whether existing or planned, that operates without any automated data exchange between the physical and digital versions. These models can represent various objects, like factory simulation or product prototypes, and may utilize digital data from physical systems, though all data exchange is manual. Changes in the physical object do not automatically affect the digital model, maintaining a clear separation.
- A digital shadow concept has automated one way data flow from physical objects to digital representation. This means that changes in the physical object's state automatically update the physical object, but changes in the digital model do not impact the physical object.
- A digital twin represents a fully integrated, bidirectional data flow between a physical object and its digital counterpart, enabling real-time synchronization and mutual influence. In this model, the digital representation can even control or adjust the physical object. Changes in either the digital or physical object directly influence the other, supporting dynamic interactions and responsiveness between the two.

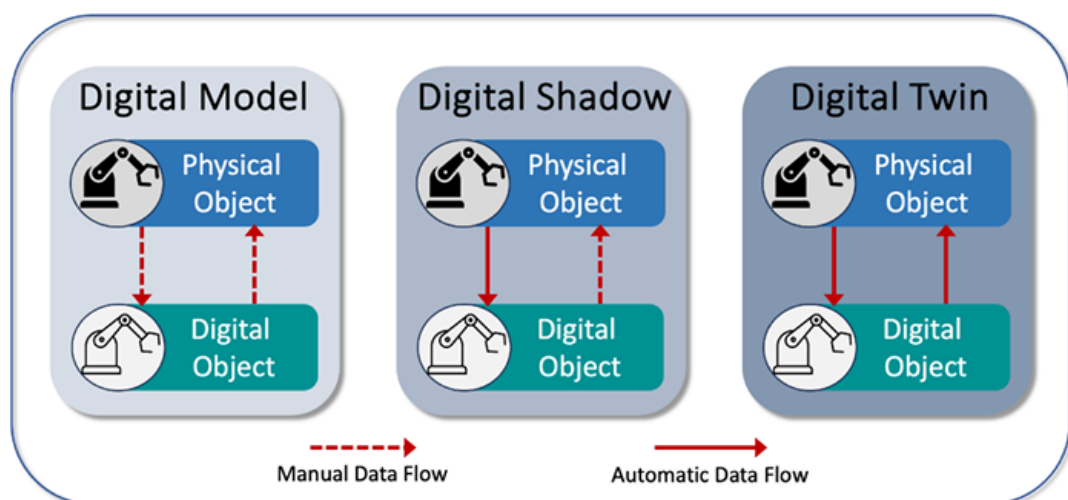


Figure 3. Digitalization approaches [31]

When building a setup for VC, there are different configurations to separate the physical and simulated partitions. Striffler et al. [31] presented the most typical configurations (Figure 4) in their journal article.

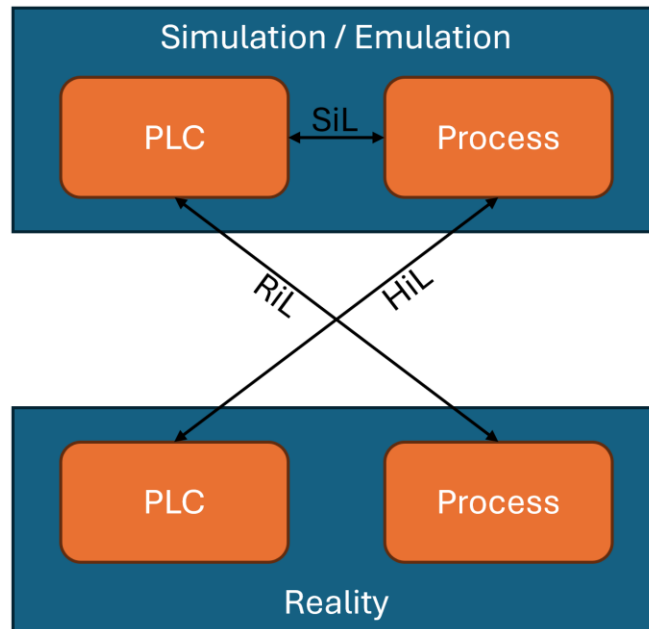


Figure 4. Most typical VC configurations, adapted from [32]

- Model-in-the-Loop (MiL) is the initial stage of testing control algorithms that are developed in a modelling language rather than PLC-specific programming languages like IEC 61131-3. Typically, these algorithms are built using graphical development tools like Matlab or Simulink, with simple simulation models that represent plant behaviour. This approach allows early validation of control strategies within the same simulation environment. [33] [32] [34].
- Software-in-the-Loop (SiL) involves testing the automation program by coupling an emulated control system with a simulation model. The control software is written in standard PLC-specific programming language and executed within the emulated control environment. Communication between those systems is usually implemented with protocols like TCP/IP, OPC or shared memory. SiL facilitates early logic program testing, independent of the hardware and without real-time constraints, allowing for time manipulation and complex simulations. [32] [34].
- Hardware-in-the-Loop (HiL) advances the testing process by loading the logic code onto actual hardware and connecting it to the simulated plant via a real communication interface like Profinet or EtherCAT. HiL is commonly used in VC

to evaluate the interface behaviour of the control system, including bus utilization and PLC cycle times, with high real-time processing demands. Successful HiL test allow seamless transition to the physical configuration. [32] [34].

- Reality-in-the-Loop (RiL) or hybrid commissioning integrates the real plant with a virtual control system or combines parts of the actual process with simulation. This hybrid approach aims to protect the plant from damage and minimize the production waste during commissioning by gradually transitioning from a simulated to a real operational environment. [32].

2.3 PLC Programming and Testbed Environment

PLCs are modular industrial computers used to automate processes by collecting and acting on real time data [34]. PLCs read input signals from sensors and user interfaces and execute logic based on programmed instructions. Based on those instructions PLCs control outputs such as relays, valves and motors. [36] [37]. A typical PLC system consists of three key components [37]:

- CPU module: The central processing unit executes the application program based on input conditions. It performs logic operations, data handling arithmetic and communicates with the memory and the I/O system.
- I/O module: This interface with field devices such as sensors and actuators. Input modules receive signals and output modules send control signals.
- Programming device: This is external unit, usually a PC or other handheld device used to develop and upload the control logic. Once programmed, the PLC can function independently without needing continuous connection to the programming device.

2.3.1 IEC 61131-3

PLCs are generally not cross-compatible, as most programming environments are vendor-specific, requiring developers to learn new systems when switching between brands. The IEC 61131-3 standard addresses issue by providing a vendor-neutral, hardware-independent programming standard. It allows developers to transfer knowledge between compliant PLCs, reducing time and cost of learning new programming system. Since IEC 61131-3 is basically set of rules, the standard does not guarantee direct code compatibility between devices due to differences in hardware architecture and compilation. Instead, it facilitates easier migration by allowing code adaptation in a new environment. [37].

The standard, developed through an examination of programming languages from various major manufacturers, defines essential aspects for industrial automation, including addressing, execution, data formats, symbol usage, sequential control, and connection between languages. In addition, the standard specifies five PLC-specific programming languages, Structured Text (ST), Function Block Diagram (FBD), Ladder Diagram (LD), Instruction List (IL), and Sequential Flow Chart (SFC). Of these LD, FBD and SFC are graphical languages, while IL and ST are text-based. [39].

2.3.2 Siemens Software and Tools

To improve the efficiency of engineering processes, modern practices are about to move from traditional serial approach towards parallel engineering processes. Siemens has developed several software tools specifically tailored for virtual commissioning. These tools are cross-compatible and forms simulation environments for VC.

The Totally Integrated Automation Portal (TIA Portal) is Siemens engineering framework developed for all automation related tasks. It integrates control programming, visualization configuration, parameterization of drives and networks and programming failsafe applications in single environment. [40].

In order to emulate PLC without a physical device, PLCSIM Advanced is a simulation software designed to virtually emulate Siemens S7-1500 and ET 200SP PLCs. It allows extensive functionality testing and validation withing the context of a machine and system. Key features include the ability to configure network interfaces, manage virtual time scaling, and use an extensive API for integration with other plant or machine simulations. [41] [42].

Accurate behavioural models are key to reliable VC [43]. To simulate actuators such as valves, drives and different parameters such as air pressure and temperature Siemens has introduced SIMIT software. It enables testing of the fault conditions and the analysis of device performance within a simulated environment. Additionally, SIMIT supports process simulations and be integrated with other simulation tools for comprehensive testing. [44].

In the NX Mechatronics Concept Designer (MCD) Siemens provides an integrated systems engineering tool for designing and validating mechatronic systems. NX MCD allows mechanical, electrical and automation engineers to concurrently work on system designs, thus reducing integration issues. It features open interface enabling smooth interoperability among engineering disciplines, avoiding redundant data handling. NX

MCD also offers physics-based simulation capabilities, allowing early detection and correction of potential design errors digitally prior to physical implementation. [45].

2.4 Error Detection in Virtual Commissioning

To validate a software, it must be tested. The primary objective in software testing is to identify and detect errors and faults that affect the behaviour of the software application, ensuring that it functions as intended. Effective testing verifies that the software meets all the specified requirements, produces correct outputs, operates within acceptable time constraints and functions in different environments [46].

Sneha et al. [46] introduced testing approach that can be defined as steps towards fully validated software (Figure 5). It contains four levels of testing:

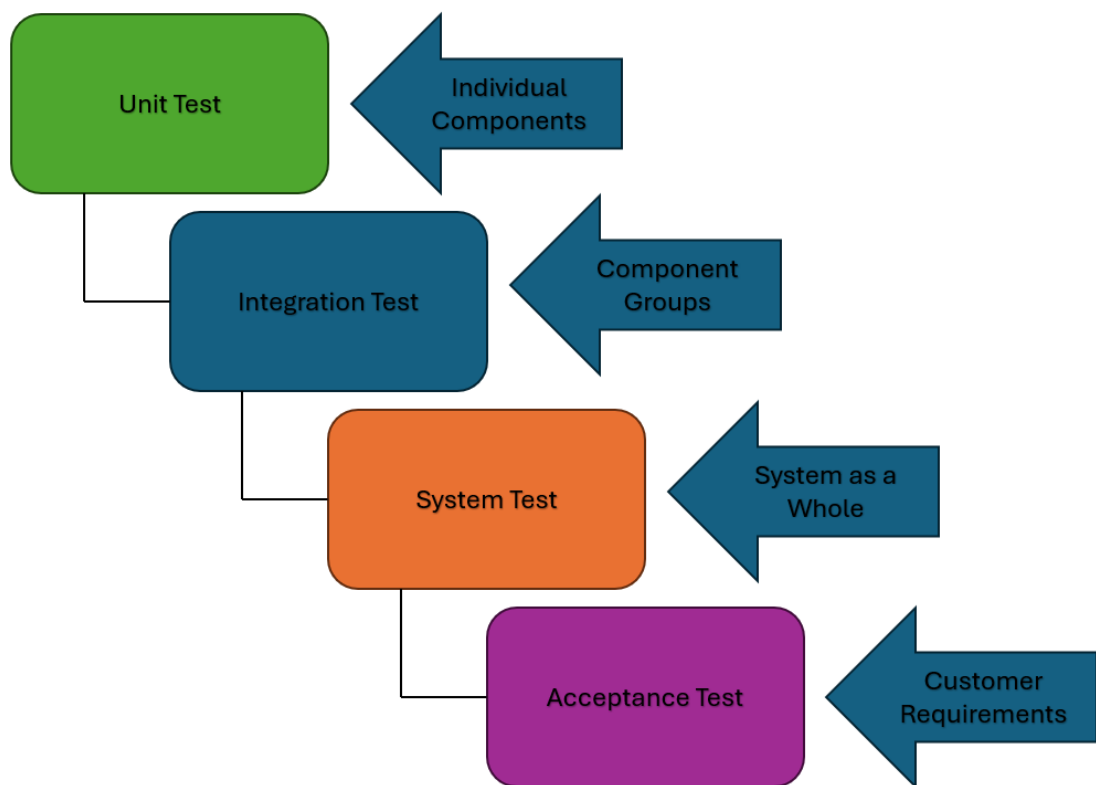


Figure 5. Software validation steps, adapted from [46].

A. Unit Testing

Unit testing is the lowest level of software validation. It involves evaluating the smallest components of a software, such as modules, functions or methods. The primary objective

is to ensure that each individual unit performs as intended. This approach supports continuous testing and iterative refinements, where the developers create unit test and modify the application code until all the tests are successfully passed.

B. Integration Testing

Integration testing is performed after unit testing and involves combining two or more software modules or components to evaluate them as a complete system. This testing can be approached using top-down or bottom-up methodologies. The primary objective of integration testing is to verify that all the integrated modules interact correctly and functions seamlessly together. It focuses on large scale structures and interfaces.

C. System Testing

System testing involves evaluating the complete and integrated software system to assess its overall quality and that it meets all specified requirements. This testing phase examines both functional and non-functional attributes, including:

- Reliability: Ensuring the system consistently performs as expected over time.
- Maintainability: Testing how easily the system can be updated and modified.
- Security: Verifying that the system protects against unauthorized access and vulnerabilities.

The primary objective of this phase is to validate all the system requirements are fulfilled and it works properly within its intended environment.

D. Acceptance Testing

This phase is carried out when the software is delivered to end-users or customers by the development team. The primary objective of acceptance testing is to confirm that the system operates as intended and meets all the user requirements in the real-world scenarios, rather than primarily focusing on identifying bugs or errors. This testing phase ensures that the system is ready for deployment and satisfies the acceptance criteria establish by the stakeholders.

According to John et al. [46] the most important part of validating a PLC program is the monitoring of the status variables or so called “flags” and external I/Os. For ease of use,

it is beneficial if these status variables can be displayed in a user-selectable form as demonstrated in Figure 6.

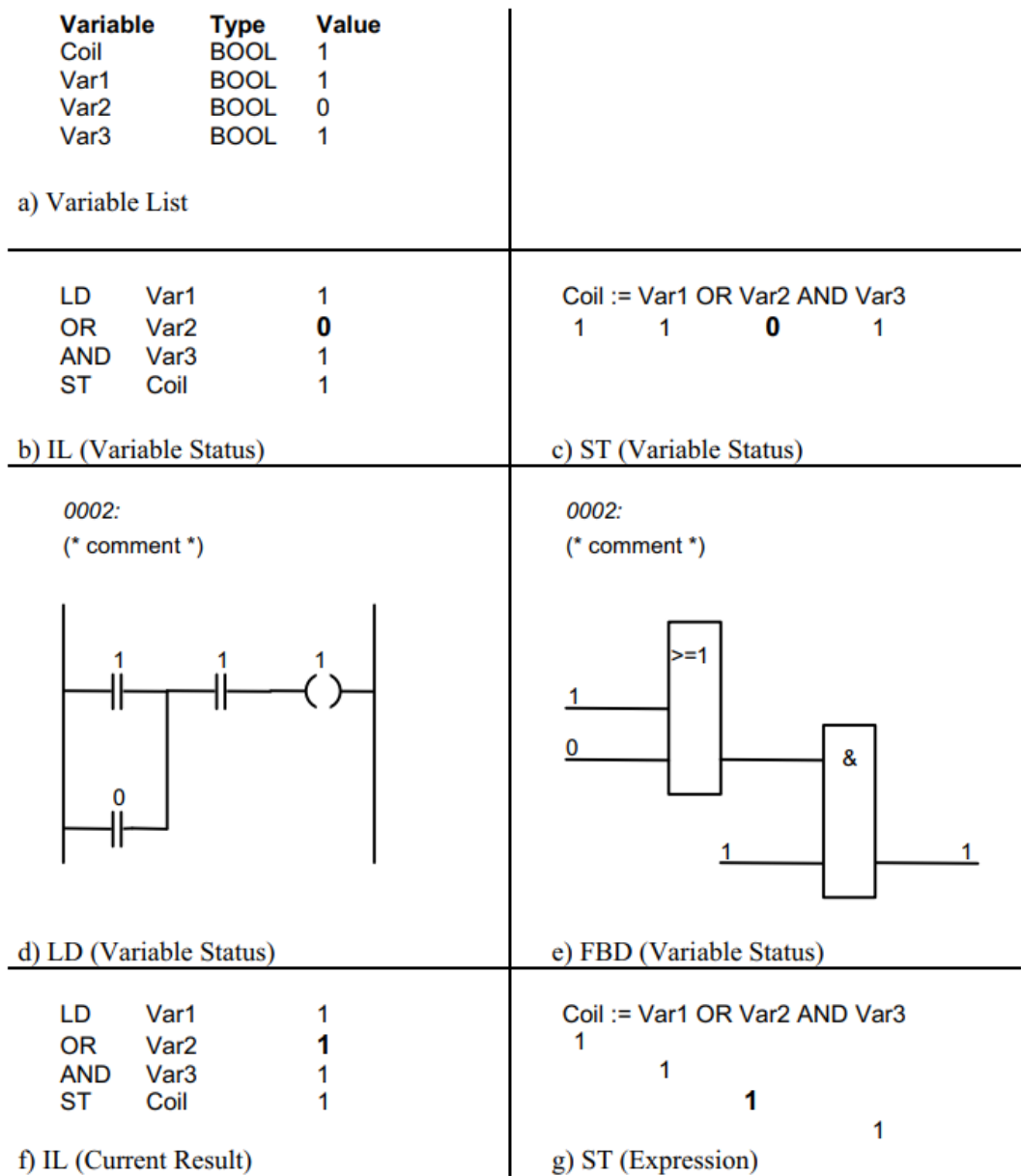


Figure 6. Example debugging status view in different programming languages [47]

Koo et al. [48] used virtual plant framework for validating PLC logic software of automated car body assembly line. With the validation they were able to identify a total of 18 programming errors. The validation process had four steps.

1. **Manual Operation Testing:** The first step involves running the virtual plant in a manual mode, which is controlled through the HMI (Human-Machine Interface). This stage helps to identify initial errors in the PLC by allowing operators manually to test and observe the system's response.
2. **Automatic Execution with Error Corrections:** After errors from the first step are identified and corrected, the virtual plant is then set to run automatically. This phase verifies whether the corrections have resolved the issues and ensures the system functions as intended without manual intervention.
3. **Signal Chart Generation:** During the automated run, a signal chart is generated. This chart captures the behavior of signals over time, providing a record of the system's operation that can be used for further analysis.
4. **Comparison and Correction of Logical Errors:** In the final step, the generated signal chart is compared with a reference time chart. By analyzing any discrepancies between the two charts, logical errors in the PLC program, such as issues with sequencing or interlocking can be identified and corrected to ensure accurate and reliable system operation.

2.4.1 Test Case Design

Software testing involves more than identifying faults, it also aims to uncover opportunities for improvement in areas such as efficiency, usability, and accuracy. To facilitate this software testing, software testing utilizes predefined scenarios known as test cases. Effective test case design is critical for robust and reliable system validation [49]. A well-designed test case consists following parameters [49] [50]:

- **Module Name:** A descriptive title that defines functionality of the test.
- **Test case ID:** A unique identifier.
- **Tester Name:** The name of the person conducting the test.
- **Test Scenario:** The test scenario offers the tester a short description that highlights the purpose, key components and actions required.
- **Test Case Description:** The required condition to be tested.
- **Test Steps:** Steps to verify the condition.
- **Prerequisite:** Conditions that need to be satisfied before the test begins.
- **Test Priority:** Priority indicator.

- Test Data: Inputs required during condition verification.
- Test Expected Result: The output required for passed test.
- Test Parameters: Assigned parameters.
- Actual Result: The output of failed / passed test at the end.
- Environment Information: The test environment, including details such as the operating system, software name, version and other relevant information.
- Status: Test result status (Pass, Fail, N/A).
- Comments: Test-related notes.

Test cases are important part of software engineering, as they indicate how testing was performed. They are used to verify whether the software functions as intended. Writing test cases offers several advantages. First, they help to ensure that the software meets customer expectations by validating whether a module or application aligns with the specified requirements. They also check the consistency of software behaviour under predefined conditions. In addition, test cases assist to narrow down software needs and identify updates. Test cases contribute to better test coverage by ensuring all possible scenarios are considered and documented. [49].

2.4.2 Software Validation and Commissioning in Process Control Systems

Before commissioning, the correct function of the systems needed must be verified. Usually, the test is separated into hardware- and software check-out. The software check-out process, as defined by IEC norm 62381, consists of three key stages. [12].

- **Factory Acceptance Test (FAT):** Conducted at the process control system implementation site, this test ensures that the software meets its specified requirements. FAT is performed prior to shipping the system to the client's site, aiming to verify that the vendor has fulfilled the requirements and client expectations. Key objectives of FAT include evaluating functionality and quality, ensuring that the vendor follows design specifications, and proactively addressing potential issues at the manufacturer's site. FAT typically involves visual inspections, functional testing, verification of contractual obligations and detailed documentation such as technical drawings, maintenance manuals, and calibration certificates. Identifying issues during FAT reduces the potential costly corrections at later stages.

- **Site Acceptance Test (SAT):** Performed at the final production site after delivery, this stage includes the same test spectrum as the FAT to verify the software's performance in its actual environment. Unlike FAT, SAT evaluates the installed equipment under real environmental conditions, incorporating actual utilities and peripheral systems. It includes visual inspection, functionality and integration test, interlock and safety device verifications and operator training. The test validates that equipment is undamaged after transportation and installation, it also validates interfaces with existing systems, ensuring the readiness for operation. SAT serves as a last validation before equipment becomes operational, thus playing a critical role guaranteeing specific operational requirements.
- **Site Integration Test (SIT):** This testing phase focuses on the correct functioning and interaction of multiple integrated systems.

The final stage of testing is generally called User Acceptance Test (UAT). This is where the end user validates the application being used. Generally, the end user also defines the UAT test cases. The acceptance criteria for this phase are agreed upon between the developer and the end user. [51].

2.5 State of the Art in Verifying and Validating PLC Programs

Program Organization Unit (POU) -oriented unit testing is a way to validate components of control software in IEC 61131-3, like functions, function blocks and programs individually by testing each POU separately. This approach is performed by first isolating each POU to be tested individually to ensure that performs as expected without interference from other parts of the program. Test typically utilizes global variables like TEST_RESULT to indicate pass or fail and TEST_READY for monitoring the status of each test cycle. The testing framework often includes assertion functions to check whether specific conditions within each POU hold true under test, ensuring that each unit meet predefined requirements. In testing mock objects are often used to simulate external dependencies or interactions with other POUs. This is helpful when testing POUs that interact with hardware or other software units that are not developed yet. By focusing on each POU separately ensures that each component of the software is verified for accuracy and reliability. [52].

In addition to POU-oriented testing, simulation-based PLC software validation tool SIMIT Rapid Tester is a software tool from Siemens used for automated testing and validation in automated systems. It is an external add-on to the SIMIT simulation platform and must

be connected via OPC UA, explained in section 2.6.3. The Rapid Tester enables continuous validation of automation code by running predefined test cases, helping to identify potential errors early in the development process. The software also allows test results to be exported to a .csv file, facilitating test documentation and enabling efficient tracking and reporting the validation outcomes. According to Siemens this approach ensures the functionality and reliability of automation systems while reducing the need for physical testing environments and minimizing downtime during commissioning. [53] [54].

2.5.1 V-Model Testing

The V-Model validation model (Figure 7) is a structured software development methodology that sets software development and testing as parallel activities. Each software development phase is directly associated with a corresponding testing phase, ensuring that quality assurance is integrated throughout the full development lifecycle. This model is most suitable for well-defined and stable requirements, as it facilitates early detection of defects and promotes cooperation between developers and testers. [55] [56].

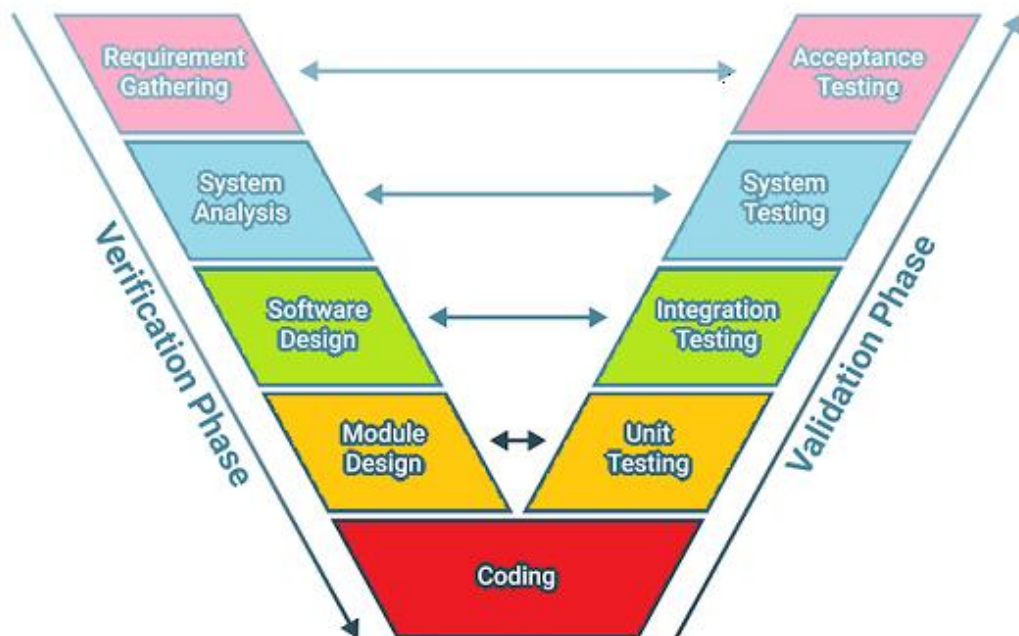


Figure 7. V-Model design [55]

The model focuses on testing and validating the software at the earliest possible stage, reducing the cost and effort of the later phases. V-models structured approach and clear documentation requirements make it suitable for projects where regulatory compliance and detailed reporting are critical. On the other hand, its rigidity and assumption of stable requirements may limit its applicability to projects where requirements are expected to evolve over time. [55] [57].

Overall, the V-Model provides a disciplined framework that unifies development and testing processes, improving product reliability and quality assurance. Its applicability is most useful in environments where requirements are well understood and unlikely to change, enabling a systematic and predictable process. However, this structured approach may not be suitable for dynamic and evolving context such as a panel repairing line, which is the testbed of this study, since the requirements of the project may change during the project.

2.6 State of the Art in Virtual Commissioning

Lechler et al. [57] introduced three exploratory use cases for Virtual Commissioning of advanced production systems. In the human-robot collaboration (HRC) or “Human-in-the-Loop” use case, one of the main challenges is ensuring safety and worker acceptance in environments where humans and robot interacts closely. To address this, Lechler et al. propose extending VC simulations with tools like motion capture systems and virtual reality interfaces. These technologies enable immersive and interactive simulations that consider both expected human behaviour and potential misuse scenarios. By integrating realistic robot controller simulations and safety features, VC supports thorough risk assessment and virtual training, minimizing costly iterative design changes.

Another use case introduced by Lechler et al. [58] focuses on the energy management of smart grids, particularly in integrating renewable energy sources into production systems. The main challenge is to manage the volatility of renewable energy sources and the complex interactions among energy consumers, produces and storage systems. The use case is to use VC to virtually test and optimize energy management system before deployment. In this use case advanced tools like Siemens Plant Simulation and SIMIT are used. With those tools the integration of energy and process management is handled by utilizing communication standards like OPC UA. This approach ensures compatibility and adaptability while reducing risk of operational inefficiencies.

Third use case introduced by Lechler et al. [58] examines VC in the context of special machine construction, which involves custom, high-quality projects. For special machine

construction (SMC), a key issue is the need for precise, error-free engineering. The issue is handled by integrating VC into the early stages of engineering to validate PLC code and system kinematics well before real-world commissioning. They emphasize the importance of collaborative, cross-domain modelling to detect and resolve discrepancies early. Additionally, they advocate for the creation of parametric and reusable simulation models, where possible, to reduce redundancy and improve efficiency in custom projects.

2.6.1 Virtual-Real Fusion Commissioning Using AR

Recent development steps in VC have also expanded into new hybrid methods that approach closing the gap between virtual simulation and physical systems. One of those is the integration between augmented reality (AR) and DT technology to create virtual-real fusion commissioning environments. Xu et al. [59] proposed a novel framework that enables commissioning even when some parts of the physical system are incomplete or unavailable.

The approach utilizes AR systems to overlay digital equipment onto real-world environments, allowing real-time interactions, control and visualization of both virtual and physical components. In their implementation, the authors modelled a motor rotor embedded wire production unit, demonstrating how AR-fused digital models can simulate missing machines and components, enabling early system level testing and validation before the assembly is complete. The AR interface supports viewpoint tracking and occlusion handling, ensuring realistic alignment and interaction between real and virtual elements.

According to Xu et al. [59] the system showed notable benefits. Compared to traditional commissioning methods, the AR-enhanced approach reduced commissioning cycle by 43.6% and saved 21.8% in spatial costs. Additionally, the method enabled manual operation, real-time error detection, and interference visualization, offering a level of interactivity not possible with conventional VR-only approaches. Importantly, it allowed process optimization and commissioning in the absence of physical assets, making it especially suitable for early phase testing and digital prototyping.

2.6.2 Narrowing Reality Gap in VC

When implementing VC, it has been noted that there are differences between virtual and physical models. These differences which potentially lead to incorrect operations are commonly referred to as the reality gap. Kuhn et al. [60] address this concern by examining

how imperfections in physical components, installation faults, and real-world disturbances can cause deviations from simulation results. Their study highlights that while current VC platforms accurately reflect control logic and sequences, they often fail to account for operational anomalies such as sensor drift, timing irregularities, and transient communication delays, all of which can occur in actual production environments.

Kuhn et al. [60] argue that these overlooked differences can lead to overconfidence in PLC logic that appears robust in simulation but fails under physical commissioning. To address this, they suggest embedding error-including conditions into virtual models to approximate real world failure scenarios. For example, simulation blocks could include randomized signal noise, component latency, or environmental variation which helps to assess how control system behave under stress or degraded conditions. Rather than treating VC as a purely functional testbed Kuhn et al. propose using it as a testbed for improving the software robustness before hardware deployment.

Integrating such error model into VC could have positive practical implications. It allows identifying edge-case failures, predict system degradation patterns, and design better error handling strategies within the PLC code. [59].

2.6.3 Integration Technologies in VC

OPC UA is an interoperability standard developed by OPC Foundation. It was founded in 2008, and it offers a platform independent, service-oriented architecture (SOA) designed for secure, scalable and reliable communication between industrial devices and systems. [61].

Key features of OPC UA include [61]:

- **Functional equivalence:** It combines functionalities, such as data access, alarms, while offering enhance capabilities.
- **Platform independent:** OPC UA can operate across various hardware platforms, from embedded micro controllers to cloud-based infrastructure.
- **Security:** The architecture incorporates robust security measures, including encryption, authentication and auditing.
- **Extensibility:** Desing allows to add new features without disrupting existing applications.
- **Comprehensive information modelling.** OPC UA supports the definition of complex information models, enabling detailed and organized representation of data.

These features make OPC UA suitable for integrating external validation software. Such as automated test tools or simulation platforms. For example, testing framework like SIMIT Rapid Tester can use OPC UA for real-time data exchange with the PLC, enabling automated test execution and result logging without requiring hardware connections.

DESIGN AND TESTBED DESCRIPTION

3.1 Research Design and Approach

The project commences from the following scenario: the panel repairing line has been mechanically completed and successfully FAT tested by the supplier at the supplier's facilities. Prior to delivery to the customer before the SAT phase, the line is to be virtually commissioned. This commission process involves validating the automation software, conducting a capacity run, and eliminating any potential bottlenecks that may arise, especially pre-identified curved panel removal part. The overall research approach is illustrated in Figure 8.

VC tests follow SiL approach. The initial phase of the project involves a comprehensive review of the automation program. This systematic examination enables the identification of appropriate testing methodologies for each segment of the program. At first focusing on the program's standard components, which encompass the HMI, alarm system, diagnostics and capacity runs. Following this, the project advances to a stage where the edge cases are systematically identified.

A comparative analysis of VC environments is considered outside of the scope of this study. The primary focus is on finding a systematic and efficient workflow for VC rather than evaluating the performance or capabilities of different platforms. Siemens environments have been selected for the study and justified in the section 3.1.1.

The Siemens virtual environment (Figure 9) is designed to enable the development of and testing of automation program in a simulated operating environment. The key components of the test environment are TIA Portal, PLCSIM Advanced, SIMIT and NX Mechatronic Concept Designer Player (NX MCD).

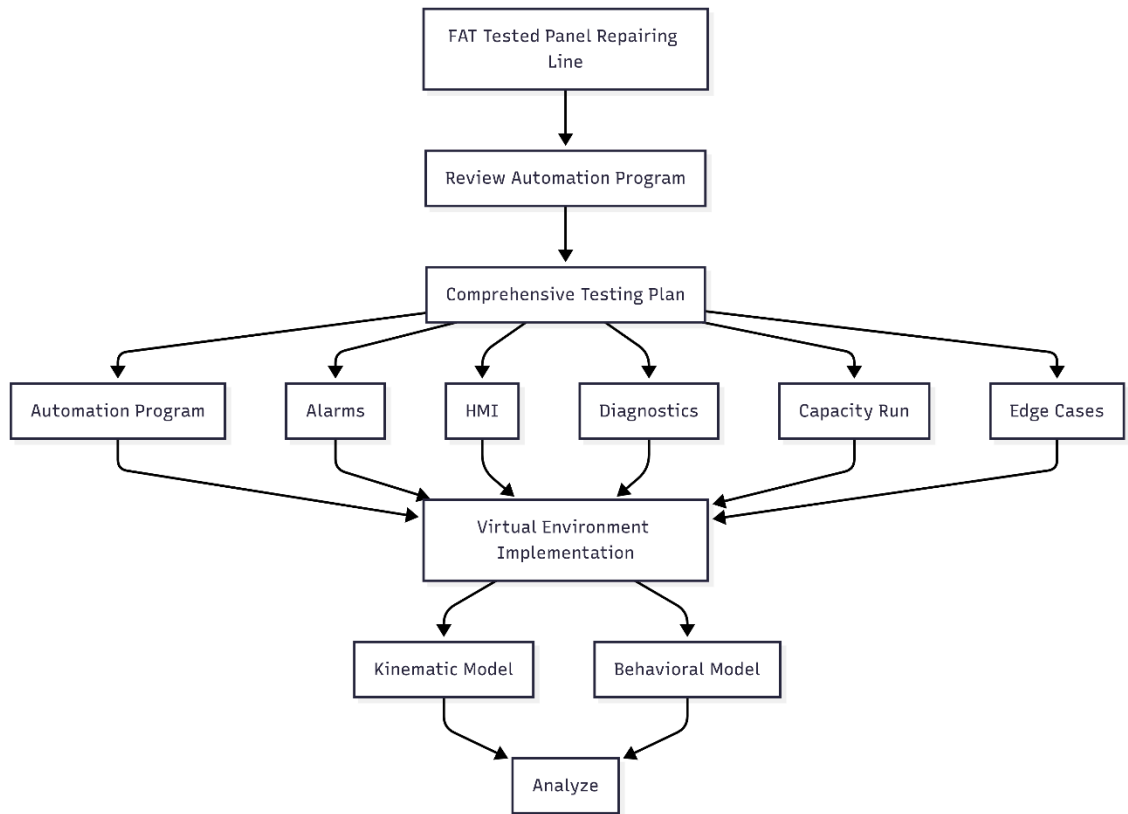


Figure 8. Research approach

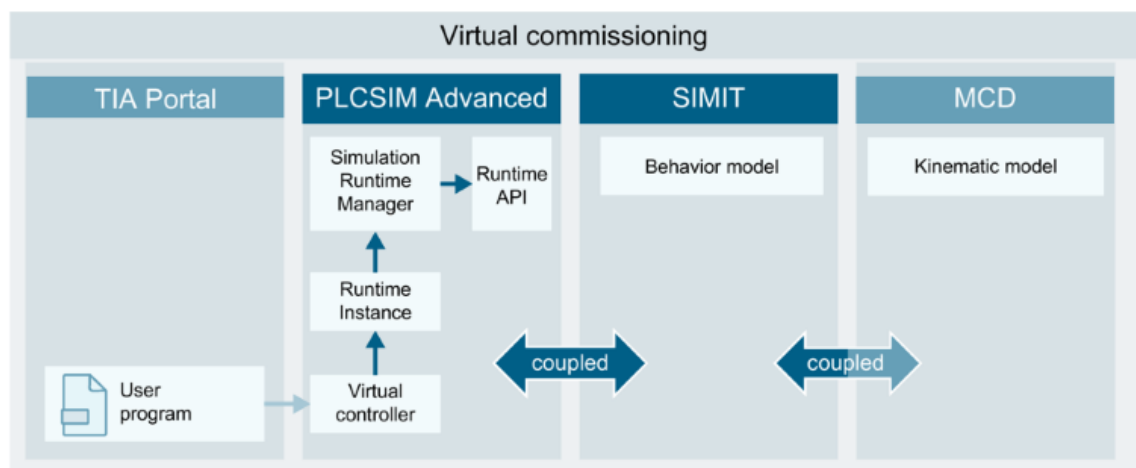


Figure 9. Siemens virtual environment [62]

Program monitoring is performed using the TIA Portal software, which serves as the interface to the simulated PLC program. The automation program is loaded into the PLCSIM Advanced, which simulates the PLC. PLCSIM communicates with the SIMIT software, enabling the simulation of sensors and actuators. This provides a realistic behaviour without need for physical devices.

Visual monitoring is carried out using the NX MCD Player software. This allows visual monitoring of the simulated system and helps to visualize the automated behaviour in a dynamic 3D environment.

Additionally, the PLC program will be tested with SIMIT Rapid Tester to validate its functionality under various conditions. New program blocks which are developed for this particular project will be tested using Test Suited Advanced in a POU-oriented manner ensuring the correct behaviour of each component. The test cases for VC are to be structured as described section 2.4.1 and developed in cooperation by the automation engineers involved in the project, including the program designers and commissioning engineers.

3.1.1 Justification of Simulation Environment Selection

The selection of Siemens virtual environment and tools of the testbed was guided by technical compatibility and practical efficiency. Since the PLC program for the panel repairing line was developed using TIA Portal, it was logical to continue within the Siemens ecosystem to maintain consistency in programming, monitoring and testing environments. Using the same platform minimizes configuration mismatches and allows seamless access to diagnostics, program variables, and integrated testing tools such as Test Suite Advanced.

A major factor in the effectiveness of the VC approach was that the panel repairing line is using Siemens drives in the physical system. Thus, the ready-made behavioural models were directly available in the SIMIT library, reducing the time and complexity needed to build and validate realistic behavioural models. These prebuilt models accurately replicate real behaviour and are easy to debug.

The PLCSIM Advanced enables efficient soft-PLC emulation with SiL method. Working is efficient when the PLC program can be easily uploaded to the emulated PLC and the PLCSIM advanced starts automatically when the simulation is started in the SIMIT software. Combined, the environment offers a scalable, modular and efficient testbed for VC which allows low-level FB testing and high-level functional validation.

3.2 Case Study: Panel Repairing Line

For this project, the testbed utilized a custom-tailored Raute R7 panel repairing line (Figure 10). The line comprises three main sections: infeed end (Figure 11), the repairing section and the outfeed end (Figure 12). While the repairing section is a standardized component that has been pretested, simulated and operates with Beckhoff logic, the

loading and unloading ends are custom tailored. These custom segments necessitate VC to validate their functionality prior to physical commissioning. Additionally, the loading and unloading sections are controlled using Siemens PLCs, and thus have interfaces with Beckhoff controlled repairing part.



Figure 10. Raute R7 panel repairing line [63]

The operation of panel repairing line is as follows: First, a stack of panels brought to the unloading section is transferred to a non-stop elevator. The purpose of the non-stop system is to increase capacity by picking up the last panels of the stack while simultaneously allowing a new stack to move to the elevator. From the top of the elevator, warped panels are transferred to a warped-panel removal station using a suction-cup conveyor. At this station, the panels are inspected to determine whether they are too warped to be repaired. If a panel is deemed unsuitable for repair, it is removed and placed in a reject stack using a vacuum gripper.

The panels then pass to repair conveyor and pass through the analyzers. The analyzer identifies defects such as holes and splits on the surface of the panels, and based on this defect map, and based on this defect map, repair heads in the line fill the defective areas with putty. The repaired panels are transferred to a continuous drying rack. Each panel is identified by its ID, and the automation logic ensures that each panel remains in the drying rack for the required amount of time. Finally, the panels are transferred to the finished stack for unloading.

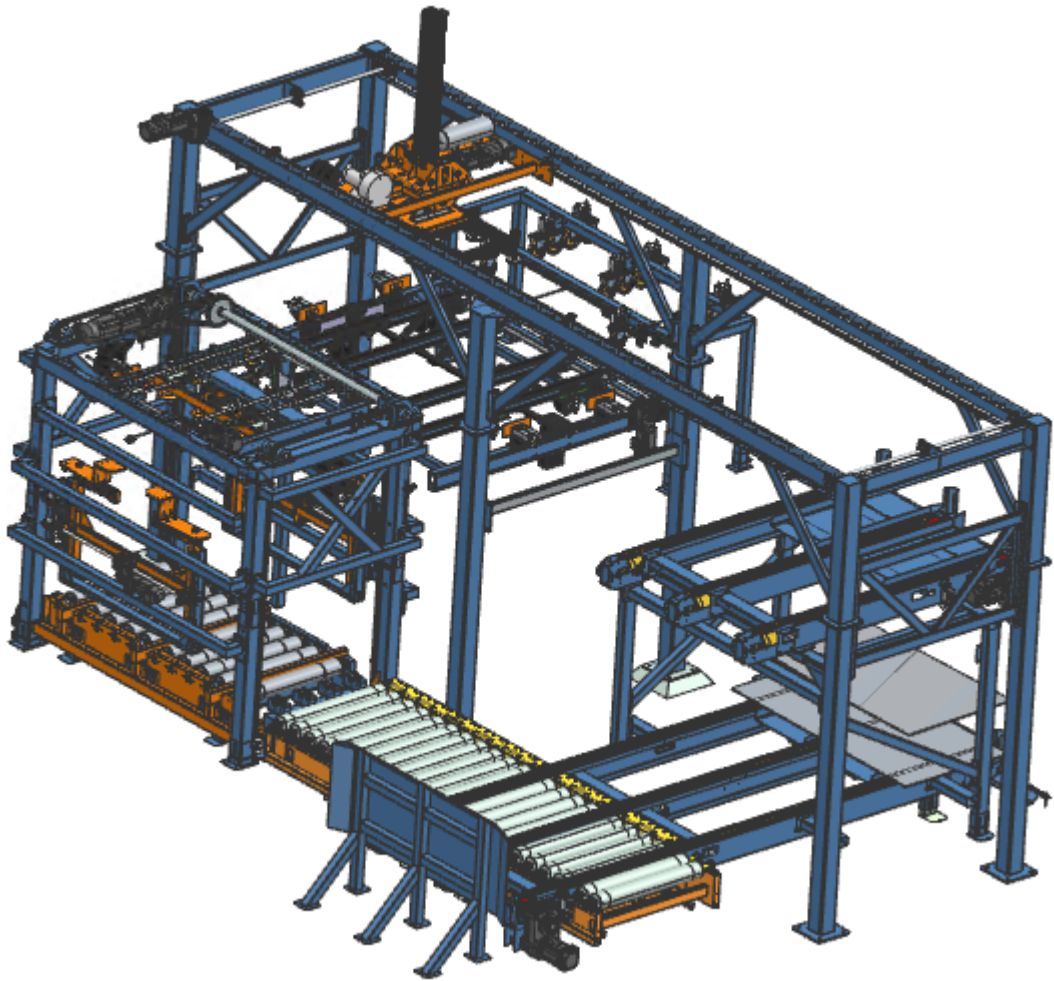


Figure 11. Loading end of the panel repairing line in NX MCD

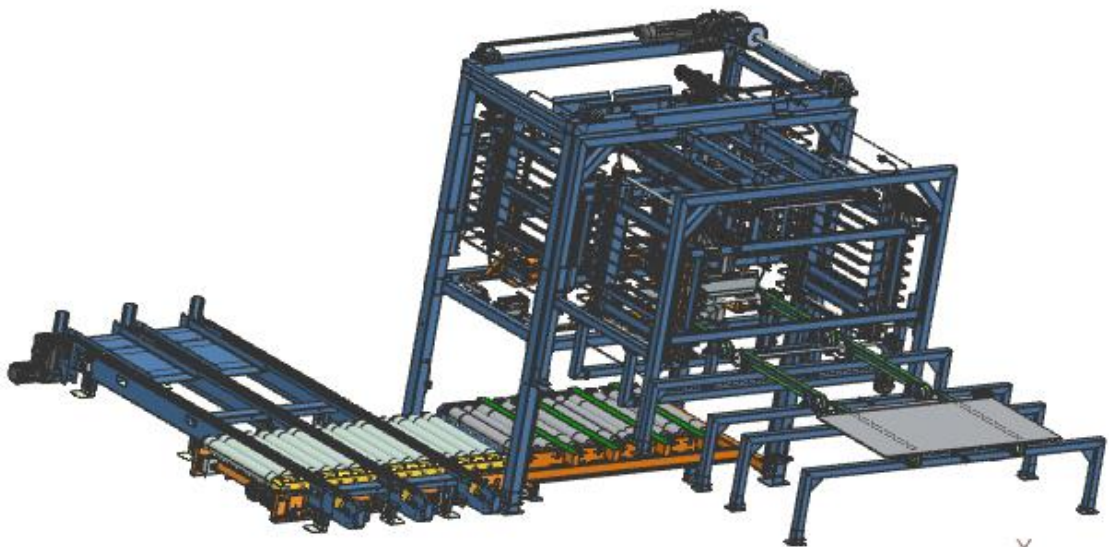


Figure 12. Unloading end of the panel repairing line in NX MCD

The project requires comprehensive testing across several critical areas, including the overall line functionality, capacity, human-machine interfaces (HMIs), alarms and diagnostic systems. The potential bottleneck in the panel repairing line that requires special attention regarding capacity during virtual commissioning is the curved panel removal part.

3.3 Tools and Software

VC leverages digital technologies to simulate, validate and optimize automation systems before their physical implementation. To perform selected validation methods for the automation software for complex processes the right kind of software plays a big role. This section outlines the primary tools used in the project.

In this project, the Siemens platform was selected for the partitions where VC is implemented. To support this decision, a comprehensive set of Siemens PLC programming, testing and simulating tools has been employed. These selected tools (Figure 13) offer environment provides a platform for the desired testing methods and seamless integration.

The project utilizes the following tools, previously introduced in section 2.3, along their designated roles:

- TIA Portal V18 (Siemens): PLC programming, configuration and monitoring.
- TIA Portal Test Suite Advanced V19: POU-oriented testing and reporting of selected non-standard code blocks.
- S7-PLCSIM Advanced V5.0 Update 1 (Siemens): Replicates the execution of physical PLC.
- SIMIT V11.2 SP1 (Siemens): Simulates sensors and actuators of the project.
- SIMIT Rapid Tester 2407 (Siemens): Automated test case execution and reporting.
- NX MCD V2406 (Siemens): Platform for the digital model to be monitored.
- Simulation Model Generator (Siemens): Enables the user to automatically generate behavior model telegrams for SIMIT.
- HWCN Exporter B11.03.00.00_10.01.00.01 (Siemens): Allows users to export hardware configurations to SIMIT.

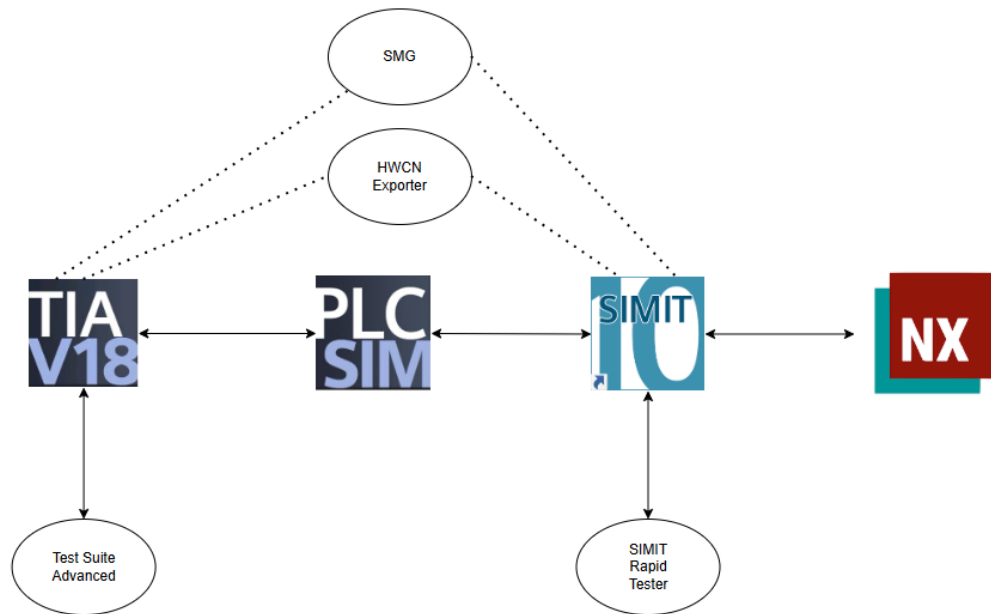


Figure 13. Siemens virtual validation network architecture

In VC, the role of PC hardware is important due to the high computational demands of modern simulation environments. VC typically involves the use of multiple sophisticated software tools involving 3D physic engines, physics solver, control system emulator and real-time data exchange frameworks. While all operate simultaneously, to achieve accurate and reliable simulations that run in real time, powerful hardware is necessary.

Simulation PC hardware:

CPU: AMD 7800x3d

GPU: AMD Radeon PRO W7700 16GB

RAM: 64GB 6000Mhz

NVMe: Corsair MP700 PRO 1TB

3.4 PLC Program Validation Plan

Collecting and categorizing test data is essential in order to find optimal software validation techniques and understanding various error types in VC processes. This section delineates the approaches and techniques employed to collect and categorize data within the testbed environment. Test results from the following categories are reported either using the reporting tools provided by the program or manually.

- Monitoring status values: Fast way to validate PLC's correct interpret and respond to various status values. By simulating for motor for example in different states such as (start, stop, fault). This verification ensures that the motor control logic operates as intended under different conditions.
- POU-Oriented testing: POU-oriented testing focuses on individual testing of individual function blocks especially tailored for this project. Test cases for the blocks are prepared in cooperation multiple automation engineers and performed with TIA Portal Test Suite Advanced software.
- Simulation-based monitoring: This is the main point of virtual commissioning; the functionality of panel repairing line can be tested and monitored from the NX MCD software.
- SIMIT Rapid Tester integration: Simulation-based monitoring test cases, which are considered necessary to be repeatable, reportable and to allow progress tracking, are managed with this tool.
- Capacity run analysis: Monitoring capacity run under various load conditions. The test evaluates whether the promised operational capacity can be achieved using panel assemblies of varying quality, with an emphasis on potential bottlenecks.
- HMI interaction: All interactions with the HMI, including button presses, screen navigation and input entries will be tested. Additionally, alarms and notification for their correct functioning and that they provide meaningful information for operators.
- Diagnostics tests: To ensure PLC programs reliability, efficiency and maintainability, diagnostics testing should be performed properly. This section test alarms and warnings.
- Edge cases and stress testing: Edge cases often involve scenarios that push the system beyond its normal operating parameters. Testing ensures that the automation system can handle unexpected scenarios. Stress testing validates the system for its stability and robustness.

To facilitate comprehensive testing and to allow all stakeholders possibility to monitor validation process, the development of precise test cases is important. Each test case is created collaboratively by automation- and commissioning engineers involved in the project. Using a standardized template, as described earlier in the section 2.4.1, every test case includes detailed parameters such as the test scenario description, precondition,

execution steps, expected outcomes, priority levels and environmental settings. This method ensures consistency, traceability and clarity. Test cases address normal operational scenarios, capacity evaluations, and critical edge case scenarios. Test cases created for SIMIT Rapid Tester and Test Suite Advanced, automated test scripts are created based on predefined scenarios. By documenting these test cases, stakeholders from different disciplines can easily monitor test process.

In order to validate PLC software comprehensively, this research integrates the systematic testing methodology described by Sneha et al. [46] alongside specific validation approaches tailored for VC. Sneha et al.'s testing framework divides testing into four levels, unit testing, integration testing, system testing and acceptance testing. Each of these testing levels has been adapted for use in testbed environment. At the lowest level, validation process employs POU-oriented testing, where individual and custom-developed program blocks are tested in Siemens Test Suite Advanced. This stage corresponds directly to unit testing, focusing on validating individual software units.

Integration testing is carried out by testing components groups starting from workpiece input. The functionalities of individual devices are tested in this section in VC environment. Following this, system-level testing is conducted by testing the system as a whole also in VC environment. This approach combines Sneha et al.'s testing framework and VC testing methodologies.

Finally, the acceptance testing phase corresponds to validating scenarios designed to validate customer requirements, including capacity runs, HMI interactions and diagnostics. By combining these testing methodologies, the study ensures comprehensive software validation before traditional commissioning.

3.4.1 Justification of Validation Model

The validation model used in this project is built on the staged testing framework introduced by Sneha et al. [46] which divides the software validation process into four distinct levels: unit testing, integration testing, system testing and acceptance testing. This approach was selected because it provides a systematic and traceable method for verifying complex automation systems. It is also suitable for projects where changes may occur in the later phases of the project, unlike the V-Model.

The unit testing level is carried out using POU-oriented testing. POU-oriented testing takes place in Test Suite Advanced since it is add-on to original PLC programming environments of this project. This was specifically chosen for custom-tailored FBs developed for panel repairing line. These blocks did not exist in previous solutions, thus making

them more prone to logical errors or edge-case failures. By isolating each function block and testing it independently with assertions and controlled inputs, potential issues could be identified early, without interference from other program elements.

This bottom-up approach is suitable for automation projects, where the reuse of generic code is limited, and logic is often tightly coupled to project-specific mechanics. The POU-oriented method also supports long-term maintainability by enabling reusability of test cases and making possible to move fully tested FBs to standard library.

Combining testing state-of-the-art testing methodologies ensures coverage and depth of the validation process. It allows early detection, clear documentation and thus builds confidence for physical on-site commissioning.

IMPLEMENTATION

4.1 Development of a Standardized Virtual Commission Workflow

The standardized VC workflow developed in this study is divided into two main phases. The first phase focuses on the creation of virtual environment suitable for VC, while the second phase covers the validation model creation.

The creation of the virtual environment is carried out in three steps. First 3D model of the panel repairing line is conducted in NX MCD, using imported CAD data, by R&D engineer at Raute. Second, behaviour model is created to simulate the behaviour of sensors and actuators. This step also includes the creation of an HMI-type control model within the behaviour model to ease VC. Finally, the signal couplings are implemented between PLC emulation, behaviour model and 3D model. When the environment is fully prepared and validated, the VC process itself can begin. The systematic VC validation process is described in section 4.2.

The implementation of the VC starts with the phase where the PLC code of the panel repairing line is ready and first lightweight testing has been carried out at the supplier's premises. A slightly simplified version of the 3D model has been imported to NX MCD software and the functionality necessary for VC have been implemented. Bolts and nuts have been removed from the 3D model to make it easier to run the software. Trajectories and collision surfaces have been implemented for actuators, panels and panel stacks. In addition, sensors and object sources and sinks have been created for the model.

When setting up SIMIT for VC, it is important to ensure that certain settings are correctly configured. In the SIMIT project manager settings, the time slice 2 parameter should correspond to the cycle time of Servo OB in TIA Portal, and the operating mode should be set to isochronous. Also, the isochronous / bus synchronous mode must be enabled in both PLCSIM Advanced and NX MCD couplings.

Connecting the PLC program to the SIMIT simulation environment is done using the SMG software that can be imported as an add-on to the TIA Portal. The software can be used to import the hardware configuration of the software to SIMIT and behavioural models of the Siemens drives, if the drives in the project have been implemented by another manufacturer, their behavioural models must be implemented manually, and the status and control words must be changed to the correct form at the bit level as demonstrated in Figure 14. The components for the conversion can be found in SIMIT.

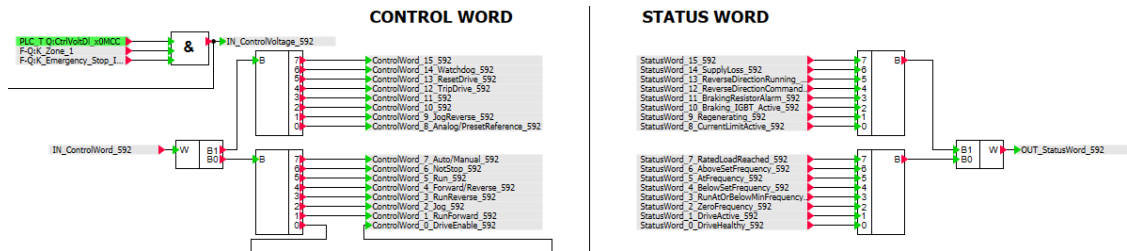


Figure 14. Status- & control word conversion example

In the simulation model development phase, attention was paid to ensuring the correct functional behaviour of sensors and actuators. Normally open and normally closed sensors were considered while making sensor chart. For Siemens drives used in the project, behaviour models also known as drive telegrams were created and imported into SIMIT using the SMG software.

When creating a simulation model for actuators and sensors, attention was paid to the correctness of the functions. While every project is unique, the testbed panel repairing line provided valuable insights. If Siemens drives have been used for the project, the behavioural models of the drives, also called drive telegrams (Figure 15) was created and imported using the SMG software. However, attention was paid to this drive telegram when importing signals from MCD coupling.

In feedback-connected drive simulation model, the MCD coupling must include speed, set speed and position signals. To ensure compatibility with the drive telegram, MCD coupling needs to be converted accordingly. The velocity data (speed signal) is usually in millimetres per second (mm/sec) and must be transformed into rotational speed (1/min). This conversion can be performed using the LinToRot function block, which requires the leadscrew circumference as a parameter. Also, the motor gear ratio must be taken into account, as it determines how many motor revolutions correspond to single shaft revolution. This conversion can be achieved by multiplying the rotational speed by the gear ratio using the MUL function block in SIMIT. By applying the same process in reverse, velocity data from the drive telegram can be converted to set speed data for use in the MCD coupling.

In feedback-connected drives, position data must be converted into pulse data for the encoder, similar to conventional feedback systems utilizing pulse sensors. This conversion can be achieved by multiplying the position data by the motor's gear ratio. In this case, LinToRot_sensor function block is used with leadscrew circumference as a parameter. Once the conversion is complete the SensorProcessRotatory function block of the encoder must be configured with parameters, steps per revolution and determinable revolutions.

Sensor couplings between SIMIT and NX MCD (Figure 16) can be implemented in two ways. In small projects, by establishing individual sensor coupling connection, and in the larger projects, by defining the connections in Excel and importing them into SIMIT. When establishing a coupling connection between two signals, the source and target names must be specified, followed by the names of the signals. Additionally, it is important to consider whether the sensors are normally open (NO) or normally closed (NC) when making the connections.

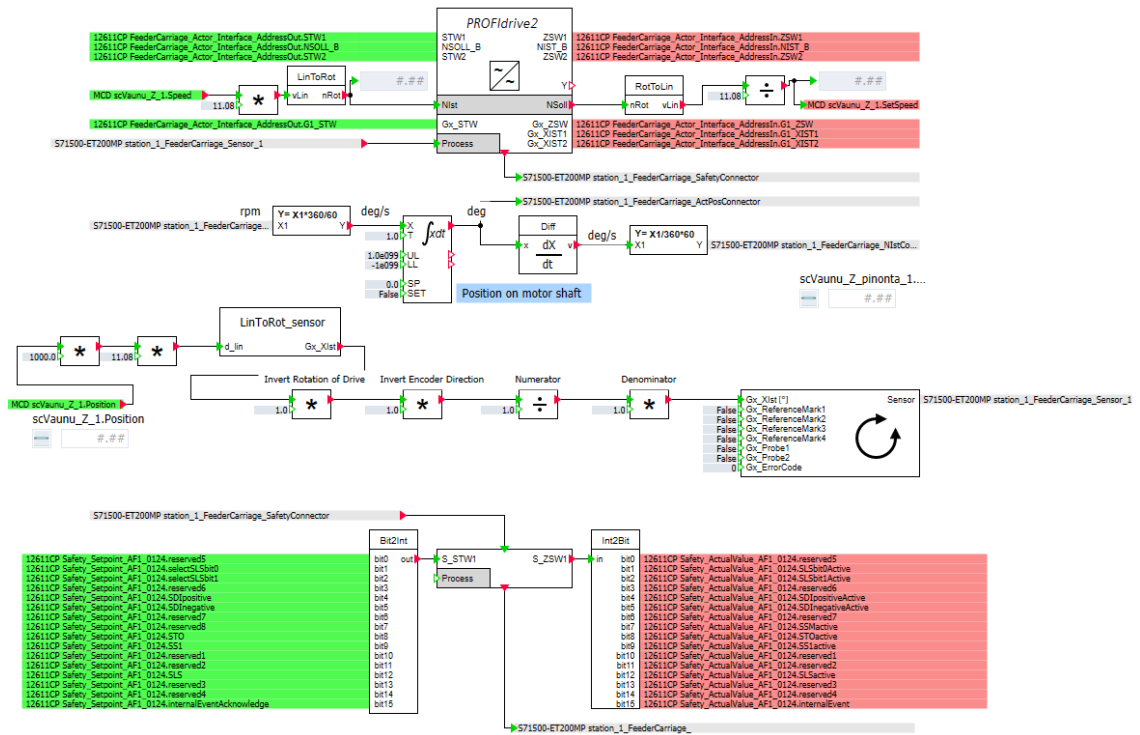


Figure 15. SIMIT drive telegram example with conversion



Figure 16. SIMIT sensor coupling example

The operation of actuators, such as pneumatic valves, can be implemented by SIMIT by bringing the control bit into the Selection control block, where parameters like actuator velocity can be defined. This type of simple modelling is sufficient for most VC applications. The actuator’s trajectories and positions are defined in the NX MCD.

4.2 Designing a Systematic PLC Program Validation Model

In this study, a systematic PLC validation model was planned and prepared. When designing the model, special attention was paid on ensuring comprehensive and efficient validation for entire program to support VC. The goal of the model was to be effective, repeatable and it would increase transparency between stakeholders.

The foundation of the validation model was based on the step-by-step model validation approach presented by Sneha et al., where the steps start from individual components and end with customer requirements. However, the model required some adaptation to ensure suitability for the testbed and to achieve more comprehensive validation. The main modifications were as follows:

- An additional initial phase was introduced to validate the program blocks specifically developed for the project. This validation is conducted using POU-oriented approach.
- The testing phases for individual components and component groups were merged. In this approach, individual components are tested from the workpiece's input direction one at a time, after which testing proceeds directly to the "system as a whole" phase.
- A separate phase was added after system-level testing to validate supplementary features. This includes for example alarms and HMI functionalities.

Testing is performed with the following tools:

- POU-oriented testing is performed using the Test Suite Advanced software. If the test is simple and do not require repetition, it can be conducted through alone.
- Test cases in the virtual model requiring repetitions are performed using SIMIT Rapid Tester software. Otherwise, simple monitoring-based validation is sufficient.

4.3 Analysis of Testbed Specific Simulation & Validation Tools

In the testbed utilized for this study, the Siemens simulation tools demonstrated strengths and some minor limitations that directly impacted to VC process, mainly in terms of work hours. The integration between TIA Portal, SIMIT, NX MCD and PLCSIM Advanced provided a cross compatible platform that provided a smooth workflow from initial setup to final software validation.

PLCSIM Advanced provided an accurate and robust platform for PLC behaviour emulation. It has the ability to simulate network communications such as Profinet, which enables realistic modelling of bus connections. SIMIT software gave reliable simulation platform for sensors and actuators, with extensive library of prebuilt behaviour models, connectors, control and conversion blocks. In addition, SIMIT have practices to facilitate coupling, such as a chart that could be imported using Excel template. However, some of the manual creation and adaptation to behaviour models added complexity and potential risk sources of error. That fact highlights the importance of careful validation of custom-built simulation components.

NX MCD provided possibility for visual validation, which is the key feature of the VC. It provided a good tool to observe events from different angles and with different settings and thus provided clear insight into mechanical behaviours and potential conflicts. A noted limitation was a need to simplify the large models to ensure real-time simulation performance. It was beneficial also to simplify the behaviour of the sensors.

Test Suite Advanced which is used to carry out the first stage of the PLC Software validation model offers benefits such as reusability of test cases, ease of use, and clear result monitoring. Once created test cases can be reused across different projects or modified for similar program blocks, saving the time in long term. The tools integration into TIA Portal also allows seamless switching between testing / test creation and code development, which also enhances the workflow efficiency. An example of structured test case implementation using Test Suite Advanced is shown in Figure 18, where individual steps validate function block behaviour through assertions and execution cycles and times.

The graphical interface simplifies the monitoring of test progress and outcomes, with pass/fail indicators and detailed result logs providing clear traceability and helps with debugging processes. This visibility is useful when multiple stakeholders are involved in the project, as it allows everyone to easily track the status of the individual test steps. As shown in Figure 18, the results of each step or test cases are displayed in the inspector window with either “Pass” or “Fail” message. Additionally, the “Go to” function enables easy navigation to the specific test step corresponding to that specific result.

However, the initial creation of high-quality test cases can be time consuming. Each test case requires careful of input- and output conditions, possible fail scenarios and edge cases.

```

1  VAR                                     //The area to instantiate variables starts with 'VAR'
2                                          //To reduce characters, placeholders can be used as shown below
3  a: InstPythagoras.a := 0.0;           //Instantiate a and set it 0.0
4  b: InstPythagoras.b := 0.0;           //Instantiate b and set it 0.0
5  c: InstPythagoras.c;                 //Instantiate c
6  valid: InstPythagoras.valid;         //Instantiate 'valid'
7  END_VAR                               //The area to instantiate variables ends with 'END_VAR'
8
9  STEP: Check_for_negative_a_and_b      //A test case starts with 'STEP:' following the name of the test case
10 a := -1.0;                            //Set a and b < 0
11 b := -1.0;
12 RUN(CYCLES := 1);                      //Run the FB once
13 ASSERT.Equal(c, -1.0);                 //Check if c = -1.0
14 ASSERT.Equal(valid, false);           //Check if 'valid' = false
15 END_STEP                               //A test case ends with 'END_STEP'
16
17 STEP: Check_for_a_and_b_equal_to_zero //Set a and b = 0.0
18 a := 0.0;
19 b := 0.0;
20 RUN(CYCLES := 1);                      //Run the FB once
21 ASSERT.Equal(c, 0.0);                 //Check if c = 0.0
22 ASSERT.Equal(valid, true);            //Check if 'valid' = true
23 END_STEP
24
25 STEP: Check_a_equal_to_3_and_b_equal_to_4
26 a := 3.0;                             //Set a = 3.0
27 b := 4.0;                             //Set b = 4.0
28 RUN(CYCLES := 1);                      //Run the FB once
29 ASSERT.Equal(c, 5.0);                 //Check if c = 5.0
30 ASSERT.Equal(valid, true);            //Check if 'valid' = true
31 END_STEP
32
33 STEP: Check_a_equal_to_0_and_b_equal_to_1
34 a := 0.0;                             //Set a = 0.0;
35 b := 1.0;                             //Set b = 1.0;
36 RUN(TIME := t#5s);                    //Run the FB for 5 seconds
37 ASSERT.GreaterThanOrEqual(c, 1.0);    //Check if c >= 1.0;
38 ASSERT.NotEqual(valid, true);         //Check if 'valid' != true
39 END_STEP
40

```

Figure 17. Example test case implementation for validating a Pythagorean function block using structured text [64]

	Path	Description	Go to	Errors	Warnings
✖	Application test		↗	3	0
✖	testCasePythagoras: PLC_1		↗	3	0
✔	Check_for_negative_a_and_b	Pass	↗		
✖	Check_for_a_and_b_equal_to_zero	Fail	↗	2	0
✖	InstPythagoras.c	Actual: -1.000000E+000, Expected: 0.000000E+000	↗		
✖	InstPythagoras.valid	Actual: False, Expected: True	↗		
✔	Check_a_equal_to_3_and_b_equal_to_4	Pass	↗		
✖	Check_a_equal_to_0_and_b_equal_to_1	Fail	↗	1	0
✖	InstPythagoras.c	Actual: -1.000000E+000, Expected: 1.000000E+000	↗		
ℹ	Test case(s) execution completed.				

Figure 18. Example of evaluation of the test results of Figure 17 [64]

DISCUSSION AND RESULTS

5.1 Standardized Virtual Commissioning Workflow

In the standardized VC workflow implemented using the Siemens test bed environment, the process begins by transferring the 3D model into NX MCD. At this stage, it is important to clean the model of unnecessary details, such as bolts and nuts. Those details do not offer any additional value for the simulation environment but possibly slows down the real time simulation.

After cleaning the 3D environment, motion trajectories, signals and sensors are defined in the MCD model. It is essential that signal naming for actuators and sensors match those used in the PLC program, as this simplifies the coupling between NX MCD and SIMIT.

After 3D phase, appropriate simulation settings are configured in simulation platform (SIMIT) to enable real-time simulation. After this, the TIA Portal add-on (SMG) is used to generate XML files of hardware configuration, plc tags and drive telegrams. These files are then imported into SIMIT, and the PLC Advanced coupling is established. Input and output signals must also be connected between NX MCD and the automation program within the SIMIT environment. A signal mapping table in Excel can be created for and imported into SIMIT to simplify the coupling process.

Special attention must be paid to the modelling of actuators, particularly ensuring that the drive gear ratios and encoded pulse sensor configurations are precise. Finally, it is important to recognize that the creation and functional testing of the VC simulation environment itself is already part of the overall testing and validation process.

5.1.1 3D Model Creation

Figure 19 illustrates the process to prepare NX MCD 3D model used in this study. The model was created by an R&D engineer on the project team and is based on geometry imported from Creo software. The process includes geometry preparation, motion and sensor definition, and signal preparation for SIMIT coupling. Each step shown in the flowchart diagram represents a stage of transforming static mechanical model into a functional 3D model suitable for VC.

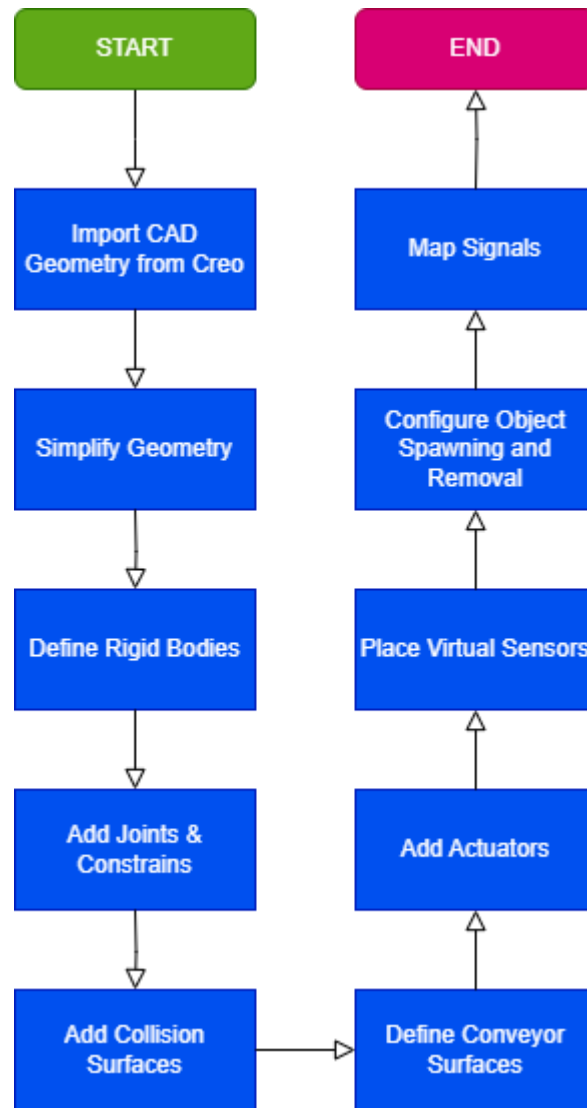


Figure 19. Flowchart of NX MCD model creation steps [65]

The steps in Figure 19 is detailed as follows [65]:

1. Import CAD geometry from Creo: The base geometry of panel repairing line is exported from Creo as a STEP and imported into NX MCD. Only essential mechanical parts are included.
2. Simplify geometry: Non-functional elements such as bolts, washers and fillets are removed. Simplification helps improve simulation performance, especially in large systems.
3. Define rigid bodies: Components that move together into single rigid bodies. This simplifies the motion logic and reduces processing overhead.

4. Add joints & constrains: Each rigid body is assigned degrees of freedom based on its movement. For example, vertical lifts are limited with minimum and maximum travel positions.
5. Add collision surfaces: Surfaces are defined parts from overlapping or falling through each other's.
6. Define conveyor surfaces: Conveyor paths are defined with direction, surface area and speed.
7. Add actuators: Grippers and moving parts are animated with control parameters such as position, force or speed, based on the PLC signals.
8. Place virtual sensors: Virtual sensors simulate for example object detection, distance measurement and actuator position feedback for the PLC.
9. Configure object spawning and removal: Dynamic components such as pallets or panels are created and removed during the simulation. Entry and exit points are defined.
10. Map signals: Simulation parameters and signals are linked between NX MCD and SIMIT. Custom mappings are used for motions like pneumatic actuation.

5.1.2 Behaviour Model Creation

After creating the 3D model, the next phase is to create a behaviour model. The model contains 7 steps described in Figure 20, and the configuration starts with defining the project settings and ends with validating the model. These steps ensure full synchronization between PLC program, behaviour model and the 3D model.

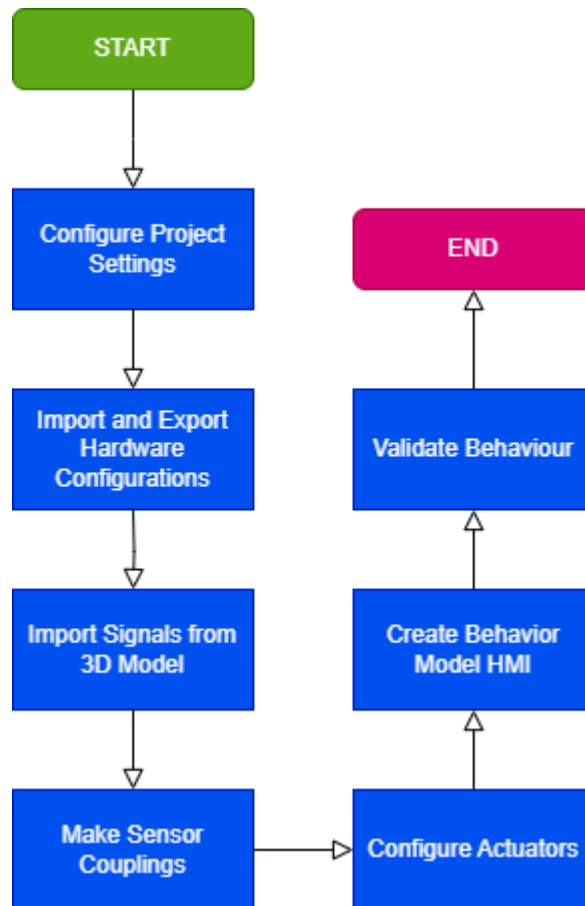


Figure 20. Behaviour model creation steps

The steps in Figure 20 with description:

1. Configure project settings

At this initial step, key simulation parameters in behaviour model are configured to ensure synchronized simulation timing and real time monitoring. Time slice 2 need to match TIA Portal Servo OB cycle time and isochronous mode should be activated.

2. Import and export project settings

The PLC hardware configuration (HWCN file) and actuator telegrams (SMG file) are imported from the TIA Portal to SIMIT. This step ensures the couplings between emulated PLC and behaviour model.

3. Import signals from 3D model

At this step, signal names are imported from the 3D model to behaviour model. These signals correspond to virtual sensors and actuators modelled in 3D model. Also signals for example to create and destroy workpiece should be imported.

4. Make sensor couplings

The sensor signals between 3D model and emulated PLC need to be coupled. Couplings can be configured manually or imported using Excel template. During this step, the sensors logical behaviour for example normally open (NO) and normally closed (NC) must be specified.

5. Configure actuators

The actuators in the behaviour model are configured to match PLC settings. For Siemens devices with drive telegram, conversions such as velocity setpoint, velocity and position must be determined.

6. Create behaviour model HMI

A simple operational interface is developed within behaviour model to manually control actuators, generate and destroy workpieces and monitor critical system variables. This HMI makes manual testing more rapid.

7. Validate behaviour:

System validation is conducted by verifying correct behaviour, for example actuator movements and sensor responses. After validation the virtual model is ready for VC.

5.2 Systematic PLC Validation Model

As a result of this study, a PLC validation model was developed, based on the staged testing approach presented by Sneha et al. [46] In this research, the method has been adapted to be as precise, efficient and practical as possible specifically for the VC of large-scale production lines.

The foundation of the validation model is a bottom-up testing approach. It begins with low-level, POU-oriented testing where program blocks specifically implemented for the project are tested. After program blocks, individual components of the production line are systematically tested, beginning from the workpiece entry point. After this, the entire production line is tested as a whole in order to validate the complete production line logic.

After the production line logic is validated, supplementary features are verified, including alarms and HMI functionality. The final phase of the validation model involves confirming that all defined customer requirements are fully met. The validation model also emphasizes the creation and documentation of case-specific and high-quality test cases. A test case form allows also non-engineering stakeholders to follow and understand the progress of the validation.

In the systematic automation program validation model (Figure 21), testing is conducted in a staged manner, progressing from lower to higher levels. Initially, the new program blocks should be tested at the low level. In the case of testbed, this is carried out with the Siemens Test Suite Advanced software. The testing procedure involves the creation of a script that encompasses all possible failure scenarios. Upon execution, Test Suite Advanced integrated to TIA Portal generates a report detailing the test results, highlighting both the successfully passed testcases and the testcases that have failed.

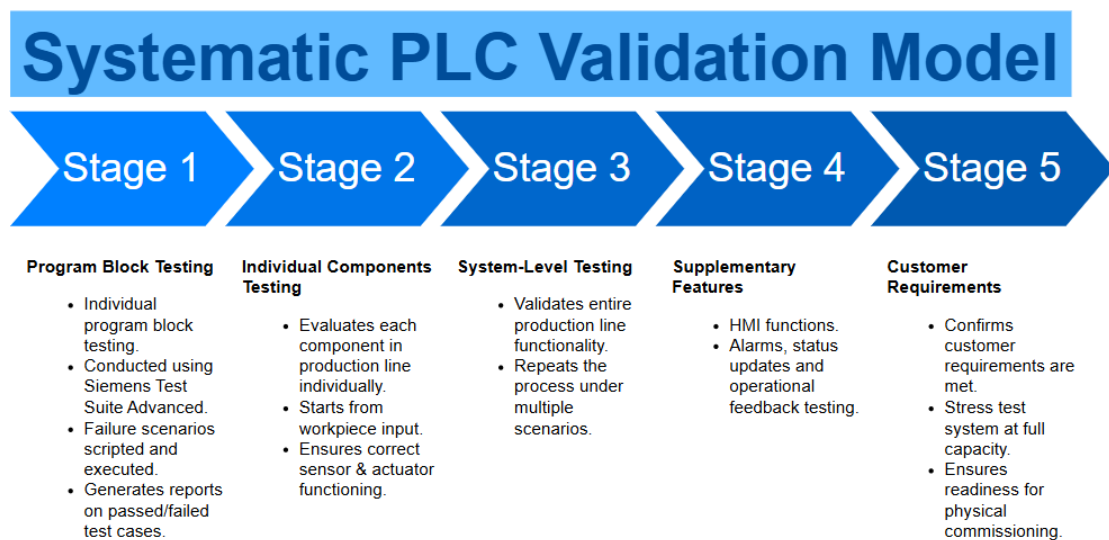


Figure 21. PLC software validation model

In the stage 2, the individual components of the production line are evaluated. The testing process of individual components can be systematically initiated from the input direction of the workpiece, with each device being tested sequentially. This can be done by monitoring behaviour and parameters, which allows verification of the operation and the setting of the sensors and actuators, such as velocities and gear ratios of the drives. By carefully assessing these parameters, the system's correct functioning of each individual device can be ensured.

Once the operation of the individual components has been validated, testing can proceed to the system level (stage 3), where the entire production line is evaluated as a whole. Initially, the operation of the system is tested by processing a single workpiece through all the functionalities and sequences of the production line. Once this step is validated, the same process is repeated in serial production, using multiple scenarios to simulate different conditions of the production.

After the validation of the system operation, an additional step to verify the control mechanism and supplementary features, such as HMI communication and alarms is performed (stage 4). Purpose of this phase is to ensure that the operator interface control commands work as planned and correctly displays the status updates, operational feedback and alerts in real time. System should also react appropriately to mitigate potential risks.

The final phase of validation (stage 5) ensures that all the customer requirements regarding capacity and functionality met. This includes stress testing the system under promised production capacity without failures. In this phase all the functionalities and requirements are identified, analyzed and resolved to guarantee that the system performs optimally and aligns with the customer's expectations. This final evaluation phase provides the final assurance that the automation system is fully prepared for the physical commissioning.

Test cases are developed in collaboration with the engineers involved in the project. The operational description and customer requirements of the panel repairing line are systematically reviewed to support the creation of relevant test cases. The test case template described in section 2.4.1 is applied for documentation. Based on this template, an Excel file is generated to capture all necessary information in a clear and structured format, making it easy for all the participants to track the status of each test case.

5.3 Evaluation the Reliability of Virtual Commissioning

The reliability of VC and reality gap in this project was evaluated by comparing the behaviour of the virtual environment to the expected real-world behaviour. Additionally, the virtual model was compared against the functions and anomalies observed during the FAT tests of the panel repairing line. In general, the virtual environment proved to be relatively reliable for VC. However, in the studied testbed, it was found that not all the faults can be detected through VC alone. The non-conformities in this study were as follows:

1. Vacuum gripper functionality

In the real-world panel repairing line, the vacuum gripper suction is activated as soon as the gripper approaches the panel surface. However, in 3D model, this does not cause successful gripping. In the 3D model, gripping is only successful when the suction and thus also the gripping signal is activated when it is in con-

tact with the panel surface. However, this problem was solved by adding a separate activation level (Figure 22) to the behaviour model, which is activated when the panel surface height level is reached.

2. Bending of flexible materials and components

Modelling flexible materials and components is challenging. In Figure 23 is a veneer sheet, which in real life is very flexible. In the 3D model of the veneer sheet bending is implemented with parallel joints. A veneer modelled in this way will not behave the same way as in real life and some validation issues may go unnoticed.

In VC, components are often modelled as “rigid bodies”, but this creates reality gap. For instance, the steel rods of drying rack (Figure 24) bend under load in real life, but in virtual model this bending does not happen. Because of situation like this, some bending related issues can remain undetected.

3. Physics solver limitations

In the testbed scenario, if for example a stack of panels becomes trapped between two collision surfaces, the compression should normally stop. However, in the virtual model the compression continues unabated, causing the physics solver to catapult the trapped panels in all directions.

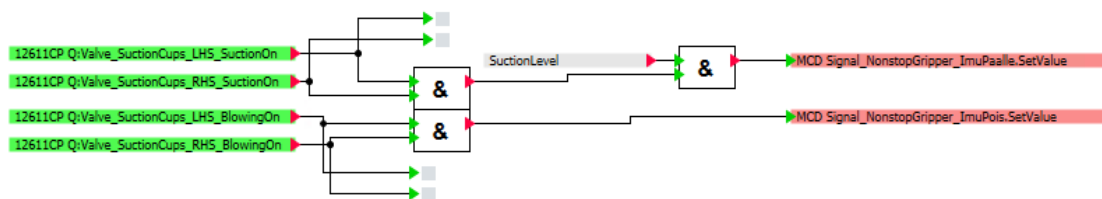


Figure 22. Vacuum gripper correction in a behavior model

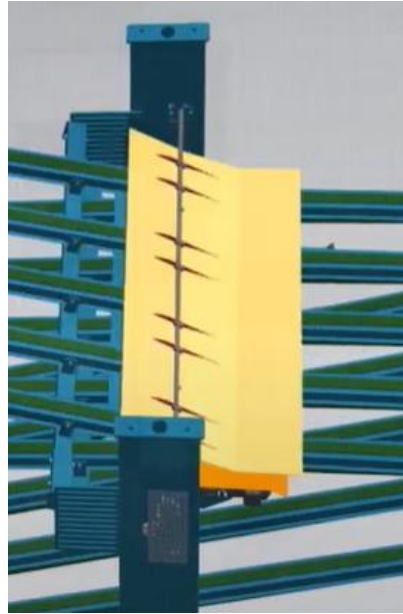


Figure 23. Veneer sheet in a 3D model

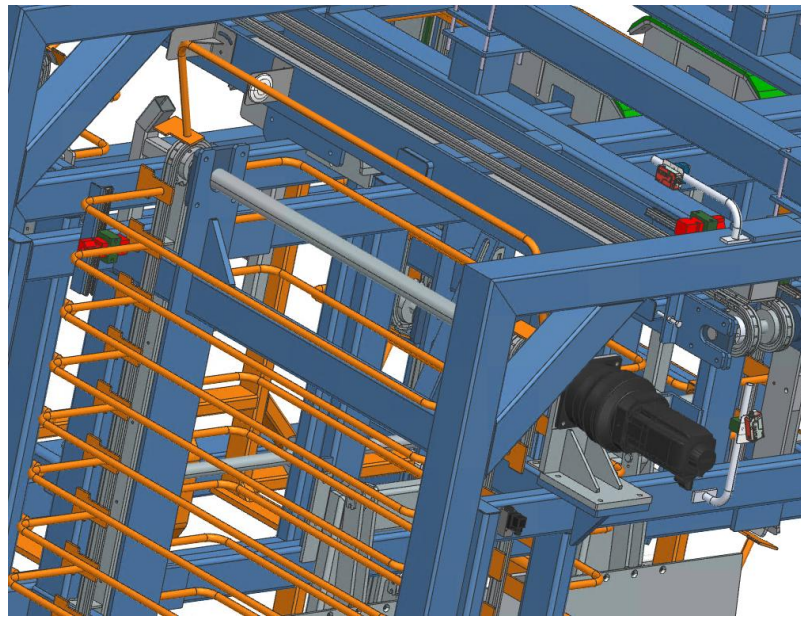


Figure 24. Drying rack poles in a 3D model

5.4 Additional Benefits for Suppliers and Customers

The additional benefits of VC can be divided into two areas: transparency and early expertise. Transparency refers to the customer's access to virtual representation of the system before assembly. Early expertise denotes the supplier's advantage in acquiring detailed knowledge of the product line in advance.

In many cases, the commission engineer is a different from the automation designer of the production line and may have never worked with the line before. In such cases, VC provides a comprehensive understanding of the system to be commissioned even before arriving on-site. For this reason alone, virtual commissioning reduces commissioning time and gives the customer a more professional impression of the commissioning engineer from the very beginning.

Customers gain early access to a virtual model of the production line, unlocking four transparency benefits:

1. Early feedback and design validation: Stakeholders can review virtual model and flag unwanted features and provide potential improvement suggestions.
2. Operator training: Operators can practice on a virtual model. This feature provides smoother production line ramp-up.
3. Virtual capacity run: Simulated capacity runs enable bottleneck identification and allow preliminary capacity run virtually.
4. Virtual user acceptance test (UAT): End users and management can perform preliminary UAT on digital system, ensuring functional requirements are met.

5.5 Interpretation of Results

The qualitative comparison presented in Figure 25 illustrates the relative performance of the key benefits of the proposed VC workflow compared against traditional commissioning. The radar diagram highlights several critical dimensions: transparency, efficiency, reliability, cost-effectiveness and risk reduction. The scoring is qualitative, based on observations and insights gathered during the Siemens testbed implementation.

The standardized VC workflow in areas such as transparency by allowing all stakeholders to track the progress of the project during the VC. Stakeholders can either participate in the test sessions themselves or review test case reports. In addition, the solution offers possibility to perform virtual UAT.

Efficiency and reliability are markedly enhanced by shifting most of debugging and parameter tuning to virtual environment reducing the on-site commissioning time and providing higher quality product to the SAT phase. A systematically tested automation program provides the best possible level of testing before physical commissioning. However, minor limitations in reliability remain, as discussed in 5.3.

Moving work phases to a virtual environment also has a positive impact on cost efficiency, as expensive on-site working time can be reduced. Project risks can also be significantly reduced when potential incorrect solutions become apparent before the line is assembled.

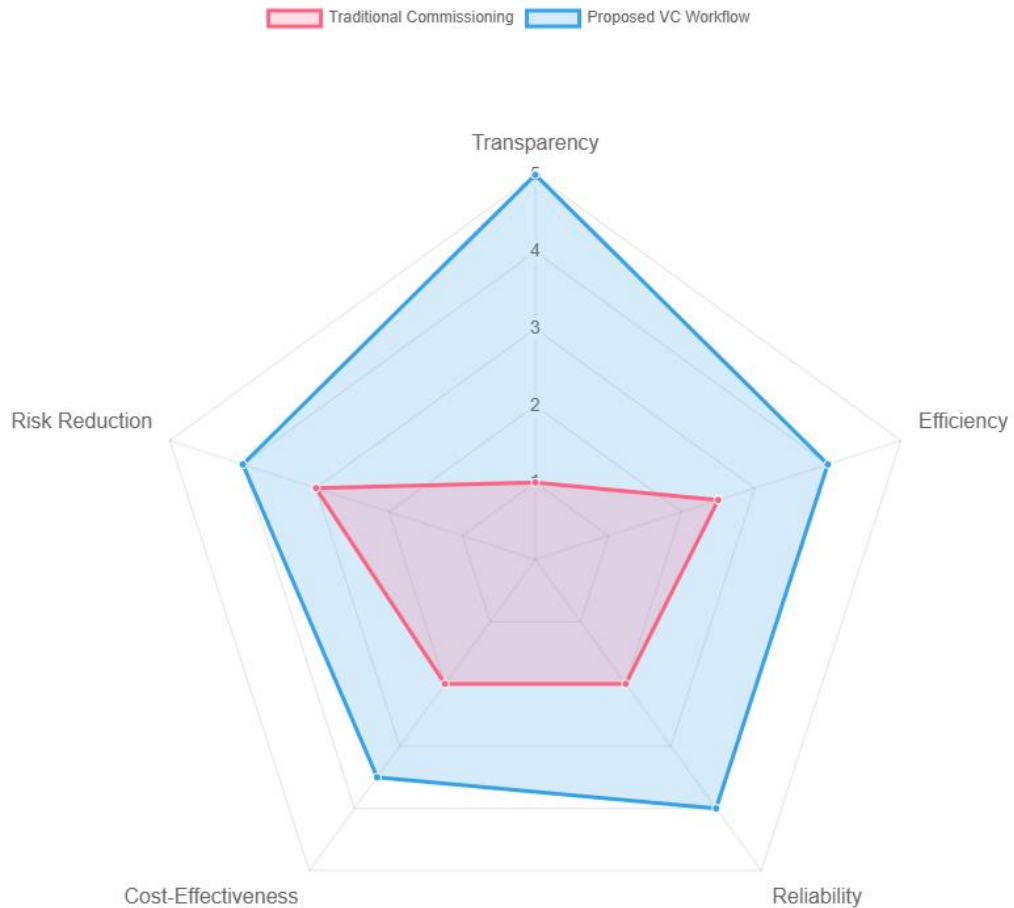


Figure 25. Qualitative comparison of the proposed VC workflow and traditional commissioning

The result of the VC implementation demonstrates how standardized workflow, combined with systematic PLC validation practices, can improve the quality and efficiency of PLC software development and commissioning and thus gives a better impression of the product quality to the customer. The systematic PLC validation model developed during this study allowed step-by-step testing, ensuring that the automation software is fully tested from the low level to high level. The approach covers all the testing from detecting logical errors to validating customer requirements.

The use of software tools like SIMIT Rapid Tester in VC creates desired features for VC, such as traceability and transparency. The software makes it easier to track test statuses and reduces manual testing. By automating test execution and systematically logging results testers do not need to maintain manual accounting of the results and all the project stakeholders can see the progress effortlessly. Example of the test management view in Simit Rapid Tester in Figure 26.

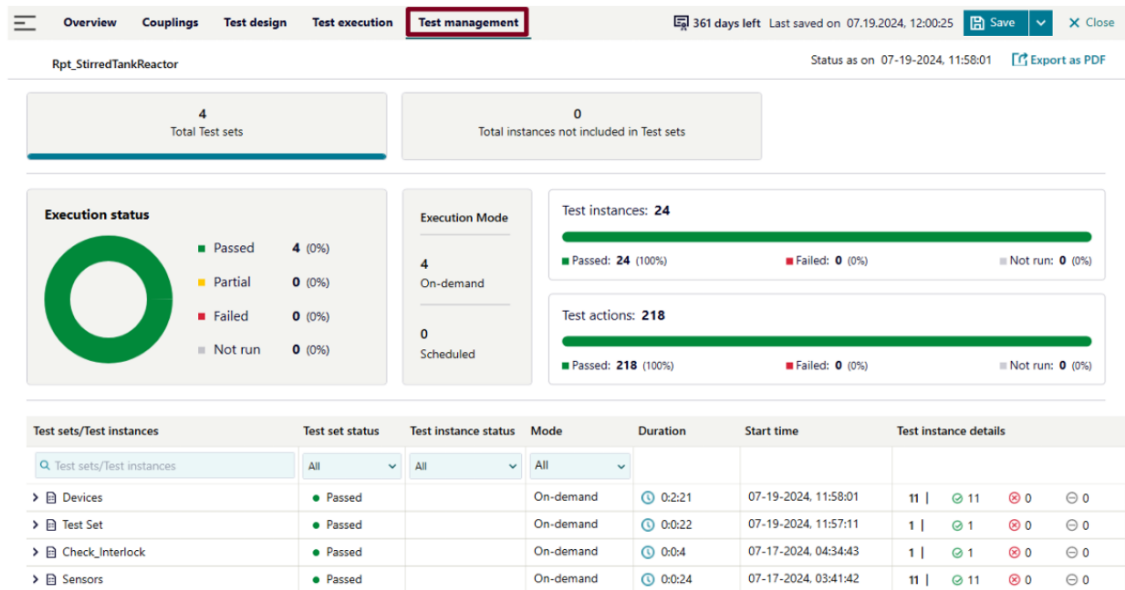


Figure 26. Test management in SIMIT Rapid Tester [66]

5.6 Implications for the Studied Testbed

The Siemens testbed virtual environment used in this study proved to be an effective platform for developing and testing standardized VC practices and debugging methodologies. The environment enabled iterative testing and integration of PLC logic, virtual sensors & actuators. Learned practices from the testbed can directly applied to future project with similar system architecture or easily adapted to slightly different systems.

In the studied testbed, a variety of actuators, such as pneumatic valves, servo drives, and vacuum gripper units were transformed into functional digital models. The virtual models created for these devices offered comprehensive perspectives on different implementations. Behaviour models implemented for these devices are easy to transfer and adapt to future projects.

5.7 Advantages and Limitations of Siemens Environment

The primary advantage of the Siemens test bed environment was the ease of the modeling the drives and servos. Since Siemens drives were also used in the real production line, ready-made simulation models were available in the SIMIT library. That reduced simulation environment development time and ensured that the behavior of virtual components matches the actual drives and servos. As one of the main goals of VC is to save both time and company resources, the cross-compatibility between software tools proved beneficial. For example, TIA Portal, which was used for PLC programming in the project, had add-on (SMG) capable of generating hardware configuration and drive telegrams directly from the TIA Portal project to SIMIT.

NX MCD virtual modelling does not fully correspond to the real world, which has both advantages and disadvantages. If desired, the pieces of the model can be made to collide with each other or with the workpiece by creating collision surfaces. Although in the real world, particles are not allowed to collide with each other, there are situations where simulation is made easier when particles can pass through each other in certain situations.

As a small side note, in NX MCD version 2406, the gripper is able to successfully grasp an object only if it is in direct contact with the gripping surface at the exact moment the gripping signal is activated. For, example in the case of test bed, where the suction is activated while the gripper is still approaching the panel surface, the object will not be grasped successfully. This behaviour requires modification to be made in the SIMIT model to ensure correct simulation.

5.8 Impact on Industrial Practices

The application of VC, as demonstrated in this study, reflects an industry 4.0 approach in industrial automation towards simulation modelling. By adopting VC, companies can reduce commission risks, shorten physical commissioning and enhance collaboration between engineering disciplines. This approach is particularly valuable in projects with custom-tailored solutions or tight physical commissioning schedules. Also, by using this approach new functionalities can be tested, and possible production capacity can be optimized.

The findings of this study contribute to higher product quality and help commission engineers to build expertise earlier in the process. Results also create transparency around the delivered system from the very start of design phase.

5.9 Practical Recommendations for Virtual Commissioning

When making a simulation model for sensors and actuators, it is beneficial to build an HMI-style chart (Figure 27). The chart may include signals that are regularly needed for testing. Signals can be implemented using pushbuttons, switches or sliders when a variable value is needed. Common use cases for these signals include:

- Workpiece generation and removal
- Fault resets
- Enabling and disabling automatic mode
- Regularly used manual functions

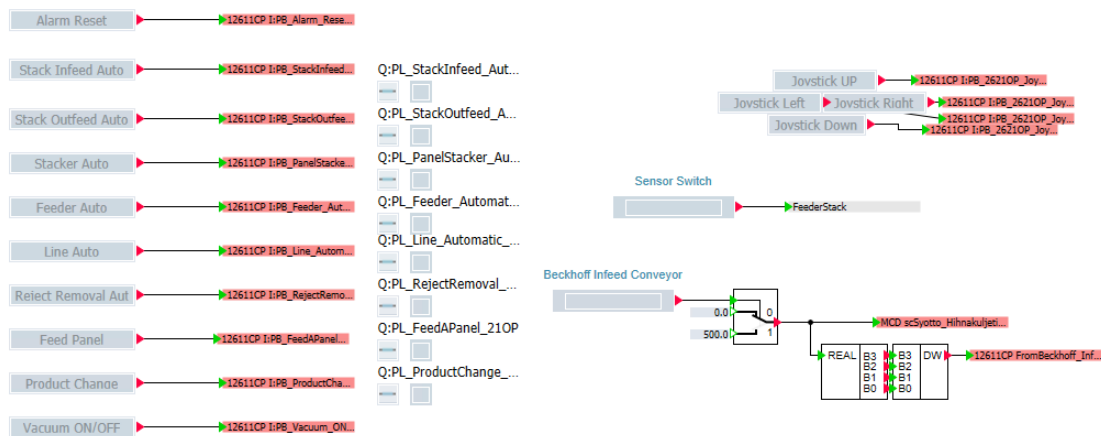


Figure 27. SIMIT HMI chart example

Creation a simulation model is already part of testing. Especially when creating simulation model for complex automation system, it would be important to resource the time of the automation program designer also for making the simulation model. The presence of the automation designer in the implementation and testing of the simulation model ensures that the model functions as planned.

When simulating sensors, it is essential to replicate the expected behaviour as accurately as necessary for the wanted purpose. However, for adjusting the simulation performance and simplicity, it is also important to evaluate which sensors can be simplified in the model.

For example, certain sensors can be represented using basic collision detection instead of detailed physical modelling. By simplifying the sensor logic where possible, the 3D model's real-time performance can be improved by reducing the computational load.

Slowdowns during the executions reduces the real-time performance and can even lead to an undesirable simulation experience.

The properties of the workpiece must also be carefully considered. For example, in the tesbed environment used in this study, the workpiece is wooden panel. Unlike standardized materials, wood is not homogeneous. That fact may lead to unpredictable behavior, for example when the panel is curved, or the surface of panel does have some defects. These characteristics should be considered when designing 3D models of workpiece or simulation logic, especially in scenarios involving gripping, detection or precise alignment.

As a general principle, all modifications related only to the simulation environment, such as signal manipulation, sensor triggers or workpiece handling should be made in simulation platform rather than in PLC code. This separation ensures that the automation logic software remains production-ready and unaffected by simulation-specific behavior. It is also important to comment on these simulation-specific changes in the simulation platform so that the tested understands why the change was made.

CONCLUSIONS

This research has provided valuable insight into how the application of VC can reduce the risks of the project, such as delays and equipment damage. Importantly, it underscores the broader implications for adopting virtualisation practices in industrial automation, highlighting the increased transparency, better cross-disciplinary collaboration and improvements to overall project outcomes. Future exploration into deeper integration with ALM practices and the potential evolution to comprehensive DT offers promising paths towards continued innovations.

6.1 Summary

This thesis explored the development of and implementation of a standardized workflow for VC and systematic validation model for PLC programs. The research was conducted within a Siemens simulation environment using a testbed on a custom-tailored panel repairing line.

The study began by analysing the challenges of physical commissioning of custom-tailored, customer-specific automation systems. Traditional commissioning methods pose risks of costly delays and equipment damage. To address these challenges, VC was selected to overcome these problems.

As a result, standardized VC workflow was developed, beginning with the preparation of 3D model in NX MCD and proceeding through the configuration of simulation platform, coupling signals, preparing actuator parameters and integrating automation logic. The workflow model underlined the importance of model simplification, signal naming and actuator modelling. A separate focus was given to sensor simulation strategies and simulation-specific configurations in the simulation platform.

Second key result is a staged PLC validation model. It applies a bottom-up approach starting from POU-oriented testing of custom-made program blocks before virtual commissioning. Next phases of validation occur during the VC. Key testing steps includes the evaluation of testing of program logic, HMI functionality, alarm systems and capacity testing. The model also highlights best practices in documenting test cases and the use of external validation tools. The results also demonstrated that the proposed VC and validation framework improves reliability and efficiency but also provide additional benefits to the supplier to the customer such as transparency and commission engineer expertise.

6.2 Future Research

While this study has focused on the development of and implementation of VC workflow, a logical next step for future research is the transformation of the existing virtual environment into a fully functional DT. The combination of VC and DT leads to more precise simulations, promotes reuse, and improves asset life cycle management [66]. Virtual commissioning has already established a simulation mode that replicates the behaviour of the physical system for software validation purposes. That study could be continued by implementing a bidirectional communication between virtual and physical systems with for example OPC UA. With DT both customer and supplier could benefit by applying use cases such as predictive maintenance, real-time diagnostics, continuous optimization and operator training [68].

VC enables early validation of automation systems by simulating PLC program and physical processes in a virtual environment. While VC focuses on testing and verification in digital environment, its effectiveness and traceability could be supported by with the integration of application lifecycle management (ALM) practices.

ALM provides a structured framework to manage the development, testing, deployment and maintenance of software and systems across their entire lifecycle [67]. It encompasses five key stages (Figure 28) [70]:

- Defining requirements: All stakeholders collaborate to identify business, compliance and technical needs. Requirements are structured either hierarchically or as use cases.
- Development: The product is built based on the defined requirements. Cross-functional collaboration ensures that the application meets user requirements and is ready for testing.
- Testing and quality assurance (QA): Testing overlaps with development and includes unit-, integration- and system level testing to validate product against requirements. Continuous integration practices are often used to streamline feedback and bug resolution.
- Deployment: The application is released to users. The method of deployment varies depending on the software type.
- Maintenance and improvement: After release, the application is monitored, updated, and supported through lifecycle.



Figure 28. ALM process. Applied from [70]

Another valuable future direction could investigate the performance of the VC workflow and PLC validation model resulted by this study. This study could interpret the financial perspective as accurately as possible. How many errors were found per testing phase, how much time was saved from physical commissioning and whether any physical equipment damage was saved. Such analysis would evaluate the validation framework's practical value and effectiveness.

REFERENCES

- [1] A. Pennanen, Automation Manager, Raute Oyj. Interview on Virtual Commissioning. Interview. 10 June 2024.
- [2] A. Sirviö, Product Manager, Raute Oyj. Interview on virtual commissioning of panel repairing line. Interview. 18 September 2024.
- [3] Visual Components, "Increasing control software quality with virtual commissioning," Visual Components, 18 March 2016. [Online]. Available: <https://www.visual-components.com/blog/increasing-control-software-quality-with-virtual-commissioning/>. Accessed: 21 September 2025.
- [4] P. Hoffmann, On Virtual Commissioning of Manufacturing Systems: Proposals for a systematic VC simulation study methodology and a new simulation model building approach, Doctoral Thesis, University of South Wales, 2017.
- [5] B. Ding, C. Flanagan, and M. Aguilar Gamboa, "Reducing Commissioning Time by 40% with a Digital Twin," Rockwell Automation, 1 June 2023. [Online]. Available: <https://kalypso.com/viewpoints/entry/reducing-commissioning-time-by-40-with-a-digital-twin>. Accessed: 27 September 2025.
- [6] M. Schamp, S. Hoedt, A. Claeys, E. Aghezza, and J. Cottyn, "Impact of a virtual twin on commissioning time and quality," IFAC-PapersOnLine, vol. 51, no. 11, pp. 1047–1052, 2018.
- [7] R. Ruzarovsky, T. Horak, R. Zelnik, R. Skypala, M. Csekei, J. Sido, E. Nemlaha, and M. Kopcek, "Development and Validation of Digital Twin Behavioural Model for Virtual Commissioning of Cyber-Physical System," Applied Sciences, vol. 15, no. 5, p. 2859, 2025.
- [8] Computerworld, "Moth in the machine: Debugging the origins of 'bug'," 11 September 2011. [Online]. Available: <https://www.computerworld.com/article/1537941/moth-in-the-machine-debugging-the-origins-of-bug.html>. Accessed: 11 April 2025.
- [9] AWS Amazon, "What is Debugging?," [Online]. Available: <https://aws.amazon.com/what-is/debugging/>. Accessed: 11 April 2025.
- [10] Computer History Museum, "This Day in History. What Happened on September 9th," [Online]. Available: <https://www.computerhistory.org/tdih/september/9/>. Accessed: 11 April 2025.
- [11] A. Beilby, "What is Virtual Commissioning?," Virtual Commissioning, 15 September 2025. [Online]. Available: <https://virtualcommissioning.com/what-is-virtual-commissioning-2/>. Accessed: 27 September 2025.
- [12] GMP Insiders Expert Team, "FAT and SAT in GMP: Importance in Equipment Validation," 19 February 2024. [Online]. Available: <https://gmpinsiders.com/fat-and-sat-in-gmp/>. Accessed: 23 March 2025.
- [13] M. Kopcek, T. Skulavik, P. Tanuska, and D. Mudroncik, "Systematic Approach to Factory Acceptance Test Planning," Computer Aided Chemical Engineering, vol. 33, pp 1597–1602, 2014.
- [14] A. Gąsior and D. Jerzy, Industry 4.0: A Glocal Perspective. New York: Routledge, 2022.

- [15] S. U. Rehman, "Industry 4.0 Adoption And Firm Efficiency: Evidence From Emerging Giants In Asia Pacific Region," *Brazilian Journal of Operations & Production Management*, vol. 20, no. 3, p. 1958, 2023.
- [16] S. Tay, L. Te Chuan, A. Aziati, and A. N. A. Ahmad, "An Overview of Industry 4.0: Definition, Components, and Government Initiatives," *Journal of Advanced Research in Dynamical and Control Systems*, vol. 10, p. 14, 2018.
- [17] E. A. Lee, "Cyber Physical Systems: Design Challenges," in *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, Berkeley, 2008.
- [18] F. Tao, Q. Qinglin, L. Wang, and A. Nee, "Digital Twins and Cyber-Physical Systems toward Smart Manufacturing and Industry 4.0: Correlation and Comparison," *Engineering*, vol. 5, no. 4, pp. 653–661, 2019.
- [19] X. Yao, J. Zhou, Y. Lin, Y. Li, H. Yu, and Y. Liu, "Smart manufacturing based on cyber-physical systems and beyond," *Journal of Intelligent Manufacturing*, vol. 30, no. 8, pp. 2805–2817, 2019.
- [20] A. Aras, M. Ayaz, E. Özdemir, and N. Abut, "Investigation on industry 4.0 and virtual commissioning," *International Journal of Engineering Technologies*, vol. 4, no. 2, pp. 108–114, 2018.
- [21] J. Molten, S. Zimmer, L. Margies, M. Karkowski, and R. Müller, "Development of a characterization and modeling method for Digital Twins in an assembly use case," *Procedia CIRP*, vol. 126, pp. 763–768, 2024.
- [22] M. Oppelt and M. Urbas, "Integrated virtual commissioning: An essential activity in the automation engineering process: From virtual commissioning to simulation supported engineering," in *IECON 2014 - 40th Annual Conference of the IEEE Industrial Electronics Society*, Dallas, 2014.
- [23] F. Auinger, M. Vorderwinkler, and G. Buchtela, "Interface driven domain-independent modeling architecture for 'soft-commissioning' and 'reality in the loop'," in *WSC'99. 1999 Winter Simulation Conference Proceedings. 'Simulation - A Bridge to the Future'*, Phoenix, 1999.
- [24] B. Davis, "How to validate machines with virtual commissioning," *Plant Engineering*, 3 September 2020. [Online]. Available: <https://www.plantengineering.com/how-to-validate-machines-with-virtual-commissioning/>. Accessed: 27 September 2025.
- [25] D. Siegrist, A. Matsikis, S. Dirbach, A. Volz, and F. Bellalouna, "A Virtual Commissioning Selection Approach for Machine Automation," in *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, Stuttgart, 2022.
- [26] Modapto, "Virtual commissioning and digital twins," Modapto, 14 February 2024. [Online]. Available: <https://modapto.eu/virtual-commissioning-and-digital-twins/>. Accessed: 27 September 2025.
- [27] Siemens, "CASE STUDY: Using virtual commissioning to reduce commissioning time by 70 percent," Siemens, 2024. [Online]. Available: <https://resources.sw.siemens.com/en-US/case-study-wipro-pari>. Accessed: 8 November 2024.
- [28] N. Shahim and C. Møller, "Economic justification of Virtual Commissioning in automation industry," in *2016 Winter Simulation Conference (WSC)*, 2016.

- [29] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital Twin in manufacturing: A categorical literature review and classification," *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 1016–1022, 2018.
- [30] M. Schamp, L. Van De Ginste, S. Hoedt, A. Claeys, E.-H. Aghezzaf, and J. Cottyn, "Virtual Commissioning of Industrial Control Systems - a 3D Digital Model Approach," *Procedia Manufacturing*, no. 39, pp. 66–73, 2019.
- [31] International Test and Evaluation Association (ITEA), "Digital Twin: A Quick Overview," 2023. [Online]. Available: <https://itea.org/wp-content/uploads/2023/12/Digital-Twin-ITEA-Industry-Defitnion-ITEA-2023-Symposium.pdf>. Accessed: 14 May 2025.
- [32] N. Striffler and V. Tobias, "Concepts and trends of virtual commissioning – A comprehensive review," *Journal of Manufacturing Systems*, vol. 71, pp. 664–680, 2023.
- [33] D. Walica and P. Noskievič, "Application of the MiL and HiL Simulation Techniques in Stewart Platform Control Development," *Applied Sciences*, vol. 12, no. 5, p. 2323, 2022.
- [34] R. B. Hill, J. Delbos, S. Trebosc, J. Tsague, G. Feroldi, J. Martin, T. Master, and N. Lassabe, "Improving interoperability of Virtual Commissioning toolchains by using OPC-UA-based technologies," in *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Vasteras, 2021.
- [35] J. Muelaner, "The Role of PLCs in Industrial Control and Test and Measurement," *DigiKey*, 21 May 2021. [Online]. Available: <https://www.digikey.com/en/articles/the-role-of-plcs-in-industrial-control-and-test-and-measurement>. Accessed: 27 September 2025.
- [36] B. Wayand, "What Is a PLC (Programmable Logic Controller)?," *MRO Electric and Supply Company, Inc.*, 20 March 2020. [Online]. Available: <https://www.mroelectric.com/blog/what-is-a-plc/>. Accessed: 25 March 2025.
- [37] Automation Ready Panels, "PLC PROGRAMMING," [Online]. Available: <https://www.automationreadypanels.com/plc-hmi/plc-programming/>. Accessed: 25 March 2025.
- [38] M. T. White, *Mastering PLC Programming: The Software Engineering Survival Guide to Automation Programming*. Packt Publishing Ltd, 2023.
- [39] D. H. Hanssen, *Programmable Logic Controllers: A Practical Approach to IEC 61131-3 Using CODESYS*. Chichester: Wiley, 2015.
- [40] Siemens Knowledge Hub, "All about TIA Portal in 120 seconds," 11 November 2011. [Online]. Available: <https://www.youtube.com/watch?v=wnntZg3CHdI&t=111s>. Accessed: 23 March 2025.
- [41] R. Yahia, "An Introduction to PLCSim Advanced," *SolisPLC*, 2024. [Online]. Available: <https://www.solisplc.com/tutorials/plcsim-advanced>. Accessed: 28 May 2024.
- [42] Siemens, "SIMATIC, S7-1500, S7-PLCSIM Advanced Function Manual," September 2019. [Online]. Available: https://cache.industry.siemens.com/dl/files/153/109739153/att_895955/v1/s7-plcsim_advanced_function_manual_en-US_en-US.pdf. Accessed: 30 October 2024.
- [43] A. Lidell, S. Ericson, and A. H. C. Ng, "The Current and Future Challenges for Virtual Commissioning and Digital Twins of Production Lines," in *Proceedings of*

the 10th Swedish Production Symposium (SPS2022), Amsterdam; Berlin; Washington, DC, 2022.

- [44] J. Uotila, "Virtuaalinen käyttöönotto - tehosta suunnittelua ja käyttöönottoa," Siemens, 2024. [Online]. Available: <https://www.siemens.com/fi/fi/tuotteet/teollisuus/virtuaalinen-kayttoonotto.html>. Accessed: 28 May 2024.
- [45] Siemens, "Mechatronic concept design," 2025. [Online]. Available: <https://plm.sw.siemens.com/en-US/nx/cad-online/automation/mechatronic-design/>. Accessed: 23 March 2025.
- [46] K. Sneha and G. M. Malle, "Research on software testing techniques and software automation testing tools," in 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), Chennai, 2017.
- [47] K.-H. John and T. Michael, IEC61131-3: Programming Industrial Automation Systems. Heidelberg: Springer, 2010.
- [48] L.-J. Koo, C. M. Park, C. H. Lee, C. Park, and G.-N. Wang, "Simulation framework for the verification of PLC programs in automobile industries," International Journal of Production Research, vol. 49, no. 16, pp. 4925–4943, 2011.
- [49] GeeksforGeeks, "Test Case," 26 September 2024. [Online]. Available: <https://www.geeksforgeeks.org/test-case/>. Accessed: 24 March 2025.
- [50] COURSERA, "How to Write Test Cases: A Step-by-Step QA Guide," 4 June 2024. [Online]. Available: <https://www.coursera.org/articles/how-to-write-test-cases>. Accessed: 27 September 2025.
- [51] H. K. Leung and P. W. Wong, "A study of user acceptance tests," Software Quality Journal, vol. 6, pp. 137–149, 1997.
- [52] M. Jamro, "POU-Oriented Unit Testing of IEC 61131-3 Control Software," IEEE Transactions on Industrial Informatics, vol. 11, no. 5, pp. 1119–1129, 2015.
- [53] Siemens, "Sales and delivery release incl. Download: SIMIT Rapid Tester 2407," Siemens, 19 September 2024. [Online]. Available: <https://support.industry.siemens.com/cs/document/109954444/sales-and-delivery-release-incl-download-simit-rapid-tester-2407?dti=0&lc=en-US>. Accessed: 18 November 2024.
- [54] Siemens, "Enhance engineering efficiency with SIMIT Rapid Tester," Siemens, 2024.
- [55] S. Jain, "V Model Testing," BrowserStack, 22 November 2024. [Online]. Available: <https://www.browserstack.com/guide/v-model-testing>. Accessed: 14 April 2025.
- [56] J. H. Roquette, S. Matos, and M. Santos, "On the Applicability of Behavior Driven Development for Automotive Software Testing at the Functional Model Level," Journal of Applied Instrumentation and Control, vol. 1, no. 10, 2022.
- [57] A. Oppermann and B. Whitfield, "What Is the V-Model in Software Development?," Built In, 18 June 2025. [Online]. Available: <https://builtin.com/software-engineering-perspectives/v-model>. Accessed: 21 September 2025.
- [58] T. Lechler, E. Fischer, M. Metzner, A. Mayr, and J. Franke, "Virtual Commissioning – Scientific review and exploratory use cases in advanced production systems," in 52nd CIRP Conference on Manufacturing Systems, Erlangen, 2019.

- [59] H. Xu, D. Wu, Y. Zheng, Q. Yu, and Y. Zhao, "Augmented reality-based virtual-real fusion commissioning: a novel approach to production commissioning," *The International Journal of Advanced Manufacturing Technology*, vol. 131, pp. 5527–5541, 2023.
- [60] M. A. Kuhn, C. M. May, Y. Liu, A. Kuhnle, W. Tekouo, and G. Lanza, "Towards narrowing the reality gap in electromechanical systems: error modeling in virtual commissioning," *Production Engineering*, vol. 17, pp. 535–545, 2022.
- [61] OPC Foundation, "Unified Architecture – Landingpage," [Online]. Available: <https://opcfoundation.org/about/opc-technologies/opc-ua/>. Accessed: 2 April 2025.
- [62] Siemens, "Virtual commissioning of machines with S7-PLCSIM Advanced, SIMIT and NX MCD," 4 July 2024. [Online]. Available: <https://support.industry.siemens.com/cs/document/109758943/virtual-commissioning-of-machines-with-s7-plcsim-advanced-simit-and-nx-mcd?dti=0&lc=en-TN>. Accessed: 4 April 2025.
- [63] Raute Oyj, "Crasman Studio. Retrieved from internal company resources.," 2024.
- [64] Siemens, "Test Suite Advanced: Example for the Application Test of Code Blocks," 8 September 2020. [Online]. Available: <https://support.industry.siemens.com/cs/document/109779805/test-suite-advanced-example-for-the-application-test-of-code-blocks?dti=0&lc=en-FI>. Accessed: 11 April 2025.
- [65] H. Korpilahti, R&D Engineer, and colleagues, Raute Oyj. Interview on 3D model creation steps for NX MCD. Interview. 24 April 2025.
- [66] Siemens, "APPLICATION EXAMPLE – SIMIT Rapid Tester with SIMATIC PCS 7," 2024. [Online]. Available: https://support.industry.siemens.com/cs/attachments/109972177/109972177_SIMIT_Rapid_Tester_DOC_en.pdf. Accessed: 4 May 2025.
- [67] H. Tanegue, W. de Paula Ferreira, R. de Assis, and D. Brodeur, "Synergies Between Virtual Commissioning and Digital Twins," *Engineering Proceedings*, vol. 89, no. 1, p. 12, 2025.
- [68] Unity, "Digital twin applications and use cases," [Online]. Available: <https://unity.com/solutions/digital-twin-applications-and-use-cases#manufacturing>. Accessed: 25 March 2025.
- [69] M. Peart, I. D'Souza-Wiltshire, C. Burke, N. Doelman, K. Vivek, A. Deore, J. Schroder, A. Buck, E. Regnier, P. Turegano, D. Kumar, J. Holtzman, D. Steinmetz, and E. Sauvé, "Application lifecycle management with Microsoft Power Platform – overview," Microsoft, 12 February 2025. [Online]. Available: <https://learn.microsoft.com/fi-fi/power-platform/alm/overview-alm>. Accessed: 25 March 2025.
- [70] A. Gillis, E. Tittel, and K. Brush, "What is application lifecycle management?" TechTarget, [Online]. Available: <https://www.techtarget.com/searchsoftwarequality/definition/application-lifecycle-management-ALM>. Accessed: 25 March 2025.