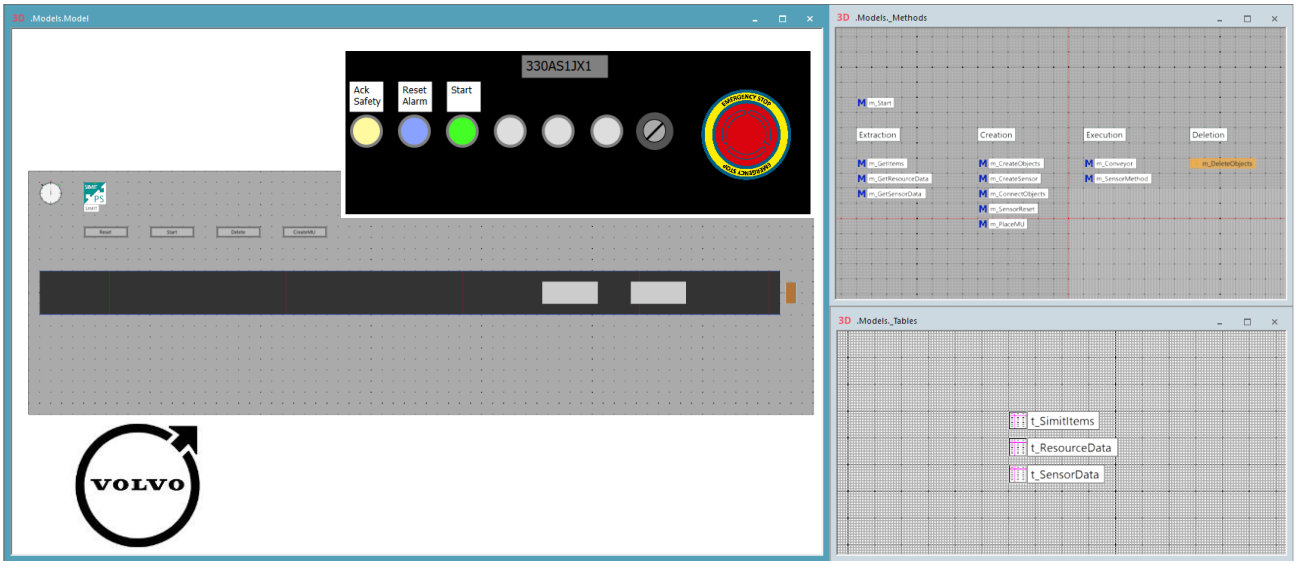
CHALMERS
UNIVERSITY OF TECHNOLOGY



Integrating Discrete Event Simulation with Real Controllers

An Early Phase Approach to Virtual Commissioning in Automotive Manufacturing

Master's thesis in Production Engineering

ABHINAV JOSHI & FREDERICK REXTON MORIS

DEPARTMENT OF INDUSTRIAL
AND MATERIALS SCIENCE

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024
www.chalmers.se

MASTER'S THESIS 2024

Integrating Discrete Event Simulation with Real Controllers

An Early Phase Approach to Virtual Commissioning in Automotive Manufacturing

ABHINAV JOSHI
FREDERICK REXTON MORIS

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Production Engineering
Division of Production Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024

Integrating Discrete Event Simulation with Real Controllers
An Early Phase Approach to Virtual Commissioning in Automotive Manufacturing
ABHINAV JOSHI & FREDERICK REXTON MORIS

Supervisors: Per Nyqvist, Department of Industrial and Materials Science
Anton Albo, Volvo Cars Corporation, Virtual Preparation and Commissioning
Dennis Andersson, Volvo Cars Corporation, Virtual Manufacturing

Examiner: Björn Johansson, Department of Industrial and Material Science

Master's Thesis 2024
Department of Industrial and Material Science
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone +46 31 772 1000

Cover: Virtual model of a conveyor line in Siemens Plant Simulation with its corresponding HMI in Siemens SIMIT in the upper right corner.

Printed by Chalmers Reproservice
Gothenburg, Sweden 2024

Integrating Discrete Event Simulation with Real Controllers
An Early Approach to Virtual Commissioning in Automotive Manufacturing
ABHINAV JOSHI & FREDERICK REXTON MORIS
Department of Production Engineering
Chalmers University of Technology

ABSTRACT:

Virtual commissioning is a continuously growing and developing area in the field of production, it helps companies stay competitive in the ever-growing sector of manufacturing. VC is a way to test logic control using computerized models even before they are commissioned onto the production line. This thesis aims at creating a comprehensive technical setup to test PLC Logic through Discrete Event Simulation (DES). A virtual model created with DES software is used to visualize and monitor the functioning of the PLC Logic. This thesis makes use of literature study and interviews with stakeholders to gather knowledge on the virtual commissioning framework, based on which the technical setup was built. An experimental phase was carried out to get a better understanding of all the tools and how to integrate them. The development phase involves using the knowledge gathered through literature study and the experimental phase to build a virtual model and connect it to a virtual PLC.

Post-development, the interconnected setup was tested to check the reliability of the integration. As a result, a completely tested and validated setup was established to test PLC logic programs with virtual models. Conversely, the reusability and repeatability of the virtual model too are greatly increased due to the extended functions such as automated object generation and real-time visualization of the PLC logic.

Keywords: Virtual Commissioning, Discrete Event Simulation, Virtual model, PLC logic, Virtual PLC, Visualization.

ACKNOWLEDGMENTS:

"Sometimes you have to be the one to stir the pot."
-Anton Albo, Volvo Cars

"Keep it simple."
-Dennis Andersson, Volvo Cars

Firstly, we would like to thank our Examiner **Björn Johansson** for his valuable insights and guidance. We also want to thank our academic supervisor **Per Nyqvist** for bringing great ideas to the table and for his unending support throughout this project. Without them, the completion of this thesis would not have been possible.

Huge thanks to our supervisors from Volvo Cars, **Dennis Andersson**, and **Anton Albo** for providing us with the necessary tools and motivation to carry out the work. Additionally, we thank them for providing us with the opportunity to work on such a valuable and interesting thesis project.

Special thanks to **Andreas Pettersson** and **Richard Thorsson** from Volvo Cars for their exceptional technical support with PLC programming. We also acknowledge and thank **Dennis Andersson** for training us in Plant Simulation and patiently clarifying our doubts.

Finally, we would like to extend our gratitude, to our friends and family for their love and support throughout this process.

Abhinav Joshi
Frederick Rexton Moris
Gothenburg, 2024

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

DES: Discrete Event Simulation
PLC: Programmable Logic Controller
DT: Digital Twin
SHM: Shared Memory
HMI: Human Machine Interface
MU: Mobile Unit
TBL: Tripple bottom line

1 INTRODUCTION

This chapter preludes to the background of this thesis on Volvo Cars. It also defines the purpose, and limitations and formulates problems by defining two research questions which will be addressed in the final report.

1.1 Background

Volvo Cars is a car manufacturing company focused on building safe and reliable cars. Manufacturing these cars needs a lot of systems to work in sync, in this case, the PLC will communicate with the conveyor and prepare it for a build. To achieve this effective communication between systems digitalization is the key, and it has become a necessity for every manufacturing company to stay competitive in the market, industry 4.0 along with digitalization not only improves the time required to manufacture but also improves the quality of products while ensuring the cost remains as low as possible.

A virtual environment helps simulate real-world data and validate the system before implementing it on a physical level. Simulation is a process of mimicking a real-world scenario to understand its behaviour in this case DES, and emulation is a process of mimicking a system in another system, in this case, emulation of PLC. For VC both simulation and virtual models must work in tandem, but this hasn't been investigated much. Integrating emulated PLC with DES models will reduce the commissioning time all the while making the model more efficient and functional by improving the life cycle of the simulation.

Preparing a specific system requires a lot of decision-making making and this in turn increases the time taken to implement a solution, therefore experimenting, and validating various scenarios in a virtual environment significantly helps shorten the decision-making time. Having an integrated system also allows for better communication between the domains.

1.2 Purpose

This project aims to find different ways of evaluating and assessing the interconnection between different systems. Building every model and having to test and validate them is a very tedious and time-consuming process. Having an integrated system makes it easier to commission as it requires fewer simulation models and increases the life cycle of the model. Integrated systems not only solve the issue of time but also can help reduce costs. This project also focuses on making sure only the relevant information is being exchanged within the test environment which would aid the testing process later in the study. Virtual commission is a growing area in production, preparing stations for production takes up a lot of time, increasing the takt time. Integrating these systems not only benefits Volvo with reduced production start-up time but also helps with the decision-making process.

1.3 Scope of Project

This project will focus more on Plant Simulation as a tool for machine-closed- simulation and Simit as an interface for signals shared between Plant Simulation and PLCSIM advanced, the reason being Plant Simulation, SIMIT, and TIA portal all being Siemens software which makes it much easier for integration and testing. As mentioned in (Ganesan S, 2019) WinMOD, despite having numerous advantages over SIMIT, is considered a disadvantage in terms of licensing cost, the time wastage in converting files, and the loss in quality of data during the transfer. The project does not deal with actual production data or integrating physical IT systems.

1.4 Limitations

This project is primarily focused on testing, verifying and validating the interconnection between the simulated and emulated model i.e. connection between the emulated PLC and simulated DES model via SIMIT which will be an interface between both. The DES model will include a specific assembly system and only the conveyor system will be considered, this project will not include other production systems. This project will deal with an already existing physical assembly line rather than designing one from scratch. Since the goal of this project is to verify the interconnection of a virtual system and a simulation model, real PLC systems won't be used; instead, they will be emulated and accessible via PLCSIM and TIA Portal.

1.5 Problem Formulation

This thesis project aims to answer the following Research Questions.

RQ1: How to establish a proper technical setup for virtual commissioning PLC logic at a comprehensive level using Discrete Event Simulation?

RQ2: When compared to simulation models with inbuilt logic, what are the advantages of using virtual models controlled by PLC logic?

RQ3: How does the established setup help improve the design decision-making process for the stations in a conveyor line?

2 THEORY

2.1 Virtual Commissioning

Virtual commissioning (VC) is a process of simulating a real-world environment in a virtual environment using real controllers. VC will help reduce the time required to amend in the real world. Commissioning is a very key part of the entire process, and it is carried out almost in the end. Although VC only amounts to 25% of the entire development time, it is the stage where most of the delays tend to happen (Wladimir Hofmann, 2017). According to Reinhart and Wunsch (Wünsch, 2007), 70% of the delays are due to errors in the control software. The traditional commissioning process happens only after the entire process is set up and physical commissioning takes place. During this process, issues tend to occur, so the entire production must be shut down. This entire process of identifying and resolving issues not only stops production leading to losses in terms of money but also in terms of time and personnel, which is not ideal. VC, however, is employed to tackle this problem of identifying issues after commissioning, validating the entire process much earlier in the development stage, and reducing the time required for commissioning. A VC usually comprises three key aspects, a digital model (in this case the simulated model of the assembly line or DES model), a controller code (in this case the emulation of PLC which will communicate with the sensors and send outputs) an interface (in this case SIMIT which connects both emulated and simulated models and allows communication between both). VC allows for testing software in parallel, ensuring there are minimal errors during the implementation phase. VC also ensures improved safety for workers because there is proper control of the entire process, this also means the machines are less likely to fail as the tests are already carried out.

Both traditional commissioning and its modern counterparts, including HIL (Hardware in the Loop), RIL (Reality in the Loop), and SIL (Software in the Loop), offer distinct approaches to verifying control programs. While traditional commissioning involves validating control programs on actual production systems, HIL utilizes both real PLC systems and simulated automation setups. RIL, on the other hand, employs simulated PLC systems alongside real process systems. SIL integrates simulated control and process systems for comprehensive testing. Notably, both HIL and SIL methods enable the detection and rectification of PLC

errors within a virtual environment, ensuring minimal disruption to real production processes. (Wünsch, 2007) provide a comparison between HIL and SIL methodologies. HIL facilitates the commissioning of complex control programs across various plant levels within laboratory settings, utilizing real PLC hardware. Conversely, SIL operates with the plant model and PLC program running on a standard PC, eliminating the need for PLC hardware. However, challenges arise due to outdated software versions, leading to the unavailability of certain control systems, and the abstract model limits its ability to accurately replicate control behaviours (Viktor Engström, 2017).

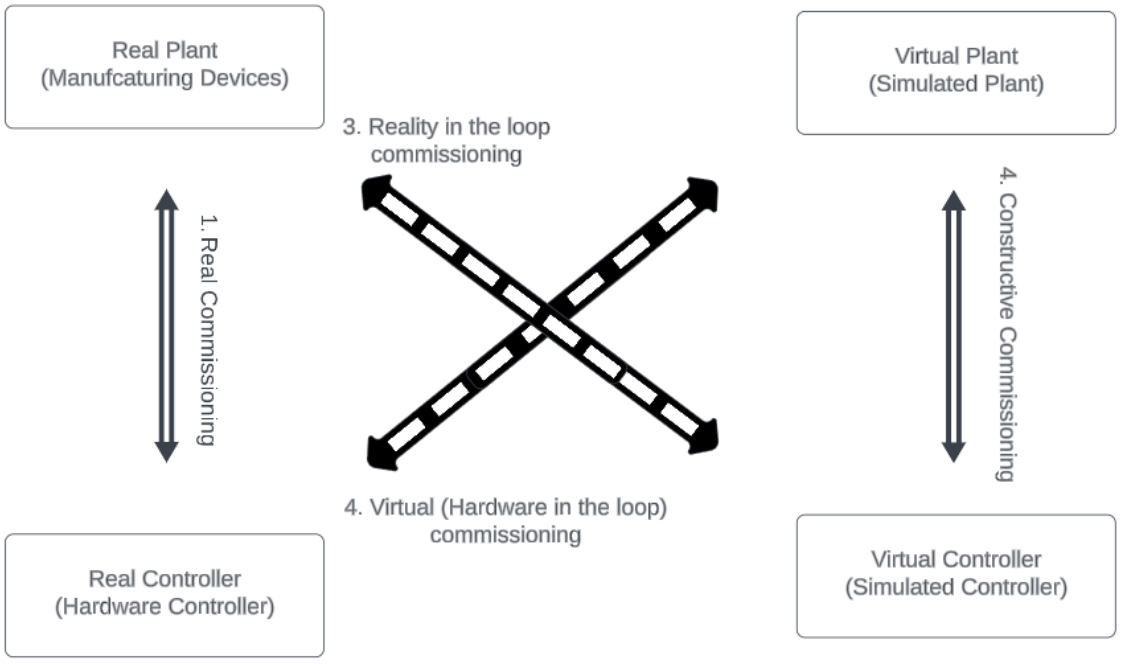


Figure 2.1 Types of commissioning as depicted by (Chi G. Lee S. C., 2014).

2.2 Simulation and Emulation

2.2.1 The Concept of Simulation

Simulation can be defined as a process of mimicking the operations of an existing system and making decisions based on different test cases. (Ingalls, 2011) defines simulation as " simulation is the process of designing a dynamic model of an actual dynamic system for the purpose either of understanding the behaviour of the system or of evaluating various strategies (within limits imposed by a criterion or set of criteria) for the operation of the system." (Shannon, 1998) defines it as "the process of designing a model of a real system

and conducting experiments with this model to understand the behaviour of the system and/or evaluate various strategies for the operation of the system". Simulation is applicable when there is a constraint of not being able to work physically, simulation uses a mathematical model to solve problems with variable time scale (Shannon, 1998)

2.2.2 The Concept of Emulation

Emulation is a process of making one system behave like the other. All models are approximate replicas of real systems, as they are replicas there will always be some differences in the emulated model, like the performance of both systems. These differences often tend to be “credibility gap”, an emulated model tries to bridge this gap by mimicking the actual model and bringing the model close to being as real as possible (Chi G. Lee S. C., 2014).

Because the control system is typically isolated from the model itself in virtual models, repeatability is unpredictable due to asynchronous and unexpected communication events. The control system and the model operate on separate clocks and synchronize through a communications layer that is subject to operating system choices. For a simulation practitioner accustomed to the reassuring repeatability of the discrete event simulation world, the ensuing uncertainty is unexpected, yet real-world control systems cope with uncertainty regularly (José Alberto Valdez Becerril, 2021).

Engineers evaluate various situations, such as experiment design, using simulation models, and then compare the results using metrics. Emulation is more frequently used to assess how well a system—like a PLC control system—is performing. While the virtual model operates in real-time, the simulation model performs scenarios in a very short amount of time. The repeatability of the outcome is very crucial in simulation. Robustness is emphasized more in emulation (Viktor Engström, 2017)

2.2.3 The Importance of Repeatability for Simulation

Two or more model runs will always execute in the same way and produce precisely the same results if no parameters are changed between runs. Any impression of randomness in a simulation model is due to the use of pseudo-random numbers to generate certain events such as break-downs, cycle times and so on. Repeatability is necessary to recreate and understand events during the model run, as well as to debug the model as it is built. All events that influence the model execution are contained within the model and are therefore repeatable (Chi G. Lee S. C., 2014).

2.2.4 The Importance of Robustness for Emulation

Because in most virtual models the control system is separate from the model itself, repeatability is uncertain, as communication events are asynchronous and unpredictable. The model and the control system work with different clocks and synchronize via a communications layer, itself prone to the decisions of the operating system. The resulting uncertainty comes as a shock to the simulation practitioner used to the comforting repeatability of the discrete event simulation world, but real-world control systems deal with this on a regular basis. (The relationship between simulation and emulation (Chi G. Lee S. C., 2014).

The material flow is monitored through statistical results, which is required as "results remain statistically meaningful" (McGregor, 2002). In the virtual model, the control logic can be validated by ensuring that all jobs are routed to their intended destinations (Schiess, 2001). The uncertainty in emulation is communication networks since it is non-deterministic. Thus, a robust virtual model is important to ensure control system can run under real conditions (Viktor Engström, 2017).

2.3 Digital Twin

A digital twin, also known as mirroring the real world, is a virtual version of an actual object that mimics real-world behavior(Tao, 2018). A DT is built upon 4 levels, geometry, physics, behaviour and rules (William de Paula Ferreira, 2020). Digital model, digital shadow and digital twin are the three levels of integration under digital twin as proposed by Kritzinger. A varying degree of integration between physical and digital objects is represented by each category. A digital twin is a fully integrated digital entity with real-time data flow, whereas a digital model is a manually modelled object that is not connected to the actual object. (Werner Kritzinger, 2018) . The figure below shows the data flow.

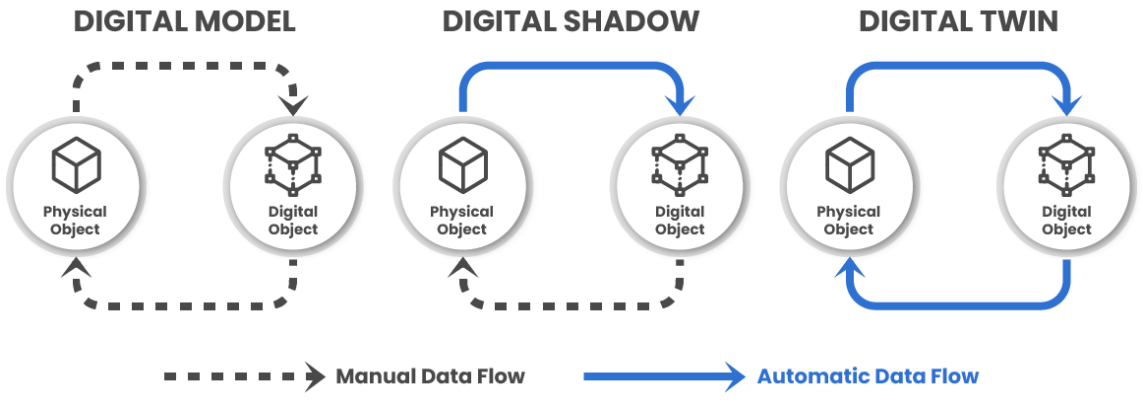


Figure 2.2 Types of data integration as depicted by (Andrade, 2023).

In order to determine if the virtual model acts and functions in a way that is similar to the actual case situation and meets client expectations, verification and validation are important processes. From their points of view, different writers present the background for verification and validation in different ways. In (Ülgen, 2001) verification is defined as "the model is akin to debugging—confirming that the model functions as the modeller intends", and validation as "model confirms that the model is an accurate representation of the current or proposed system relative to all performance metrics to be assessed by management. The more closely the behavior system matches the simulation model, the more accurately the model's performance may be evaluated. Consequently, ensuring client pleasure via a relationship based on strong trust (José Alberto Valdez Becerril, 2021). If there is as little of a difference as possible between the behavior system and the simulation model, the model's accuracy is deemed good. The figure below from (Ülgen, 2001) represents the relationship between verification and validation. (Hinchey, 2009) describes validation as conforming “doing the things right” so it aligns with the end user, and verification as “doing the right thing” by testing, analyzing and inspecting.

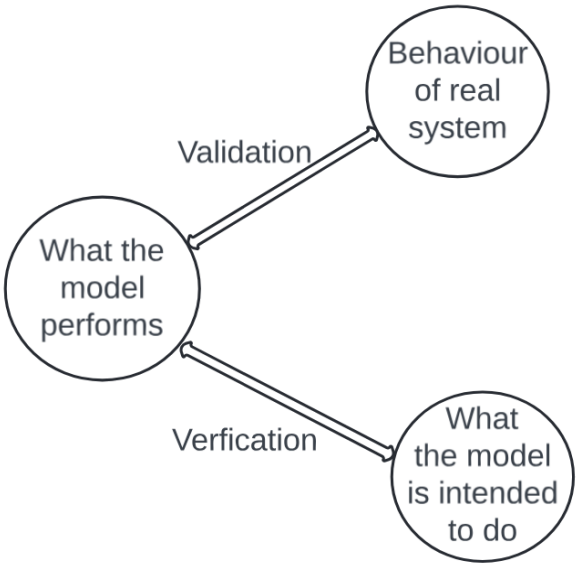


Figure 2.3. Relationship between validation and verification.

2.4 Industry 4.0

"It is a significant transformation of the entire industrial production by merging digital and internet technologies to conventional industry."
(STĂNCIOIU, 2017)

One may argue that industry 4.0 is the new reality for industries, and that it is the next step that many need to take in order to maintain their competitive advantage over their rivals. Industry 4.0 has already begun for one-third of businesses worldwide, and in the next five years, that percentage is likely to rise to 72% (STĂNCIOIU, 2017). When it comes to the definition of Industry 4.0, (Lasi, 2014) defines "Industry 4.0" as primarily IT-driven modifications to production systems. When it comes to Industry 4.0, it's crucial to remember that advancements will impact the organization in addition to the technological domain.

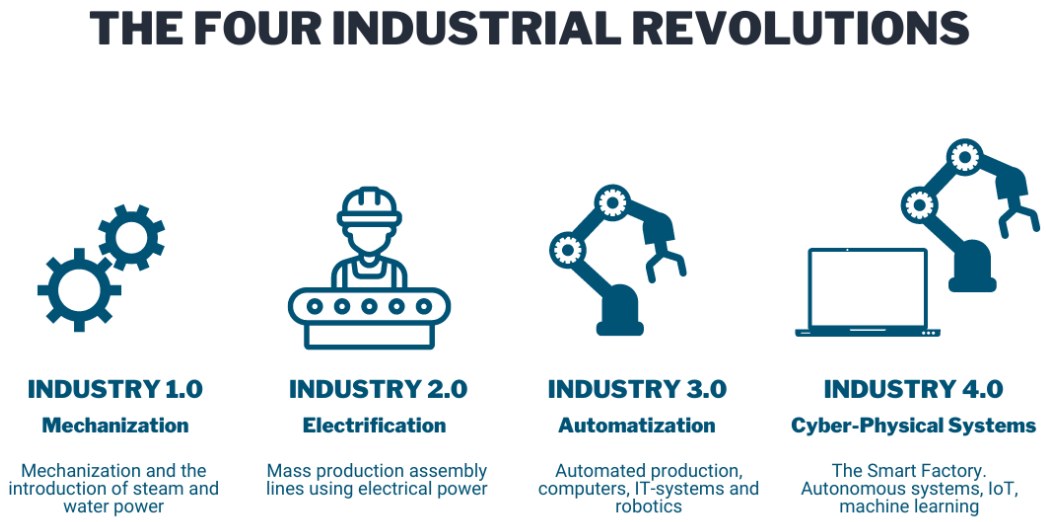


Figure 2.4. Evolution of industry 1.0 to 4.0 as depicted by (MELOENY, 2022).

Key concepts that enable technologies and development trends within Industry 4.0 include big data and analytics, cybersecurity, the cloud, green IT, the industrial internet of things, autonomous robots and systems, simulation modeling, horizontal and vertical system integration through new standards, additive manufacturing, and augmented reality. Simulation may be used as a decision support tool to enable system and component testing and validation as well as solution creation(Rodič, 2017). In the past, manufacturers had to utilize trial and error to observe how a newly constructed system behaved in order to determine if it operated successfully and efficiently. This approach led to disruptions in the system under test. Industry 4.0 introduces virtualization to build a digital twin, or virtualized version of the real world, where ideas may be tried and modeled (Gilchrist, 2016). This

would allow manufacturers to more accurately and focusedly conduct manufacturing system development (Erik Gerdin, 2018).

2.5 Discrete Event Simulation

It is becoming increasingly crucial to be able to evaluate protocols, modifications, information flows, etc. without interfering with the actual production system. (Sharma, 2015) has developed a simulation model that looks at a system's current state as well as its potential future development. The general behavior of a discrete event simulation may be described as follows: When an event occurs, the system quickly changes from its initial state to the new one. This behavior will continue over time since it will halt for a while within a state(Mansharamani, 1997). “Of all the simulation techniques, DES is the one that models the operation of a system as a discrete sequence of events in time. Each event occurs at a particular instant in time and marks a change of state in the system” (Sharma, 2015).

2.5.1 Advantages and Disadvantages of DES

(Sharma, 2015) outlines the advantages and disadvantages of applying the DES approach. Benefits include testing the likelihood that a phenomena will occur and studying and experimenting with a (complex) system. Among the disadvantages are that creating a simulation model requires specific expertise, and simulation is often associated with unpredictability. Also, to not affect the operations that already exist DES is preferred to be applied in larger areas (Muthanna Jamil, 2016). The later research identifies other flaws in the simulation, such as the replication of all inconsistent variables present in a manufacturing line and the disregard for human error or skill since it treats the data more as qualitative than quantitative.

(Yingling, 2016)explored why organizations would find discrete event simulation important and its benefits. One of the advantages they highlight is the opportunity to use simulation to show management the overall benefits of lean manufacturing and to give them a clear picture of what the new system would look like in the future. (Muthanna Jamil, 2016) further, discuss a few advantages in their essay. They discuss how fact model simulation has increased the line balancing ratio, which has improved work productivity, and how employing simulation may save time during the line balancing process.

2.6 Sustainability

Sustainability is a key factor in any manufacturing company, Volvo cars takes pride in adhering to sustainable practices. Sustainability not only includes the environmental aspect but also the economic and social aspects. This is also called the triple bottom line. The triple bottom line can be defined as a way to measure the economic, environmental, and social impact on an organization (Amos Ojo Arowoshegbe, 2018). (Amos Ojo Arowoshegbe, 2018) states the economic aspect of TBL refers to "the impact of the organization´s business practices on the economic system" The Social aspect of TBL refers to "conducting beneficial and fair business practices to the labour, human capital and the community" The Environmental aspect of TBL refers to "engaging in practices that do not compromise the environmental resources for future generation."

Figure 2.5. represents the three aspects of Triple Bottom Line (TBL) as depicted by (Amos Ojo Arowoshegbe, 2018).

3 METHODOLOGY

This chapter will provide a detailed explanation of the methodology and work procedure implemented to answer the research question of this thesis. Each research question requires a different type of methodology to answer, to tackle this, different types of analysis will be used for each question. Since the field of integrating simulation and virtual models has not been explored extensively, initially literature study was carried out to gather as much knowledge as possible followed by qualitative analysis. This gave me a solid foundation to start building both simulation and virtual models. This process of qualitative analysis after the literature study was iterative to increase the reliability of the interconnection. Fig 3.1 gives the workflow of this thesis in a flowchart. A semi-structured review was conducted to identify themes, and perspectives in a theoretical manner and list out the common issues that would occur while conducting tests (Snyder, 2019). The semi-structured review is carried out because it is not possible to review all the findings related to the topic, so this review technique identifies how the research in that specific area has advanced over the years (Snyder, 2019). This gives a broader understanding of complex themes in the research area. It helped in blending knowledge from different researchers' perspectives and creating scope for future research.

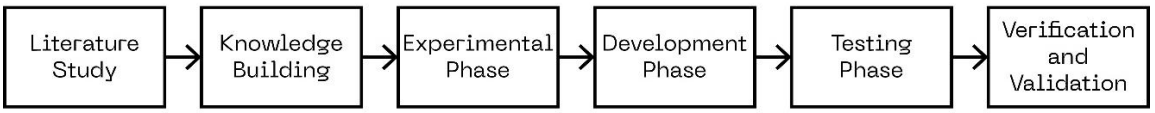


Figure 3.1: Flowchart of work procedure.

The literature study was carried out in the field of integrating an automatic object generating model with an emulated PLC and the data was analyzed from this research field. The data that was used in this project was primary data that was directly obtained from SIMIT. The experimental phase was done to get an understanding of the software, and this was an iterative process to improve competency. The development phase was used to prepare theories based on knowledge gathered from the literature study and experimental phase. This was followed by verification and validation of this technical setup.

For building the simulation model Siemens Technomatics Plant Simulation 2021 was used. To emulate the PLC Siemens SIMATIC S7-PLCSIM Advanced V6.0 and to integrate both the

simulation and emulation model Siemens SIMIT V11.1 was used. To emulate the PLC logic Siemens TIA PORTAL V17.0 was used.

3.1 Literature Study

The literature study consisted of researching and studying in the area of integrating simulation and virtual models. This study included topics mentioned in the theoretical framework (Chapter 2). The inductive reasoning approach was preferred to draw conclusions and build the model. Inductive reasoning begins with collecting data that is relevant for building models, the data is then carefully analyzed to identify patterns and later develop a theory to explain the patterns (Sirisilla, 2023). Deductive reasoning on the other hand is the complete opposite of inductive reasoning, it starts with a theory to ends up with a logical conclusion (Sirisilla, 2023). Therefore, to support the research questions in this thesis inductive approach was more logical as it starts with collecting the available conveyor and sensor data and analyzing it to formulate a theory on how to interconnect it with the virtual model.

The literature aiding this report mainly originated from sources like already published research papers, online articles and books. References dating after the year 2012 were selected because of topics like digital twin, and industry 4.0 as they are still in development, although references before 2012 have been used in this paper they had to be carefully assessed for relevance. The database used for the literature study in this thesis was mainly obtained from Google Scholar, Chalmers online library, Scopus and Science Direct. Keywords such as virtual commissioning, digital twin, industry 4.0, simulation, emulation, SIMIT, PLCSIM advanced, TIA portal and a combination of the above-mentioned keywords.

3.2 Knowledge Building

Knowledge building started with gathering knowledge related to the scope of this thesis, which included software, systems and concepts. In this phase software knowledge about Siemens Plant Simulation and Siemens SIMIT was gained, the interconnection and specific functions of both the software. For example, methods in Plant Simulation use “SimTalk” as a language for coding, the Plant Simulation virtual model is created in a “frame”, and “symbols” in SIMIT denote the signals going into Plant Simulation. The primary method of enhancing software expertise was qualitative analysis; this involved compiling data from SIEMENS handbooks and tutorial films on how to use and integrate specific software, such as SIMIT and Plant Simulation. In addition, interviews the industrial supervisors and SIEMENS representatives helped develop a greater understanding of the software. The supervisor interview helped get a rudimentary Plant Simulation model for familiarization and preliminary experimentation. The early stages of conducting this thesis did not reveal which

software was going to be utilized to make this setup, having multiple interviews with the supervisors at Volvo gave a better understanding of the tools that are going to be utilized.

3.3 Experimental Phase

The main objective of the experimental phase was to get a profound understanding of the software and how the interconnection reacts. This phase built up a rudimentary link between Plant Simulation and SIMIT by using the knowledge from the knowledge-building phase. The experimental phase began by establishing the connection. An iterative approach was implemented. To test the interconnection and get a better understanding of how the interconnection works a start/stop button was configured on Plant Simulation that would operate the simulation and was controlled by SIMIT. This exercise provided a clear knowledge of how the two software are connected to each other. This experiment consisted of a roller bed which is a conveyor in Plant Simulation that actuated when the value was changed in SIMIT. This study helped to understand how to read and write signals relating to Plant Simulation, giving a basic idea of how to control the model. Furthermore, it gave a better understanding of Shared Memory, which is a way SIMIT communicates with Plant Simulation.

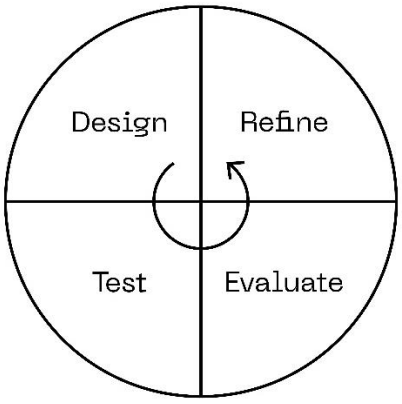


Figure 3.2: Iterative experiment process.

3.4 Automated Object Generation

The first step in automatic object generation was successfully integrating SIMIT with Plant Simulation. The roller bed data was collected in an Excel file. The data consisted of roller bed dimensions and position in the XY axis, the data was loaded into Excel manually. The Excel file containing all the data is imported into Plant Simulation with the help of methods. The objects are created by matching the signals from SIMIT and are then connected. There are several ways of building a DES model and they differ in how complex the model is supposed to be but they all are similar in modelling methods (J. A. B. Montevechi, 2015). A

method used for representing the interconnection between a simulation model and an emulation model is HSEM(Hasan's Hybrid simulation and emulation model) (Hasan, Methodology to develop hybrid simulation/emulation, 2005). HSEM model describes the steps in integrating an emulation model with a virtual model(in this case). The first step is defining the problem followed by designing and integrating the virtual model and behavior model respectively.

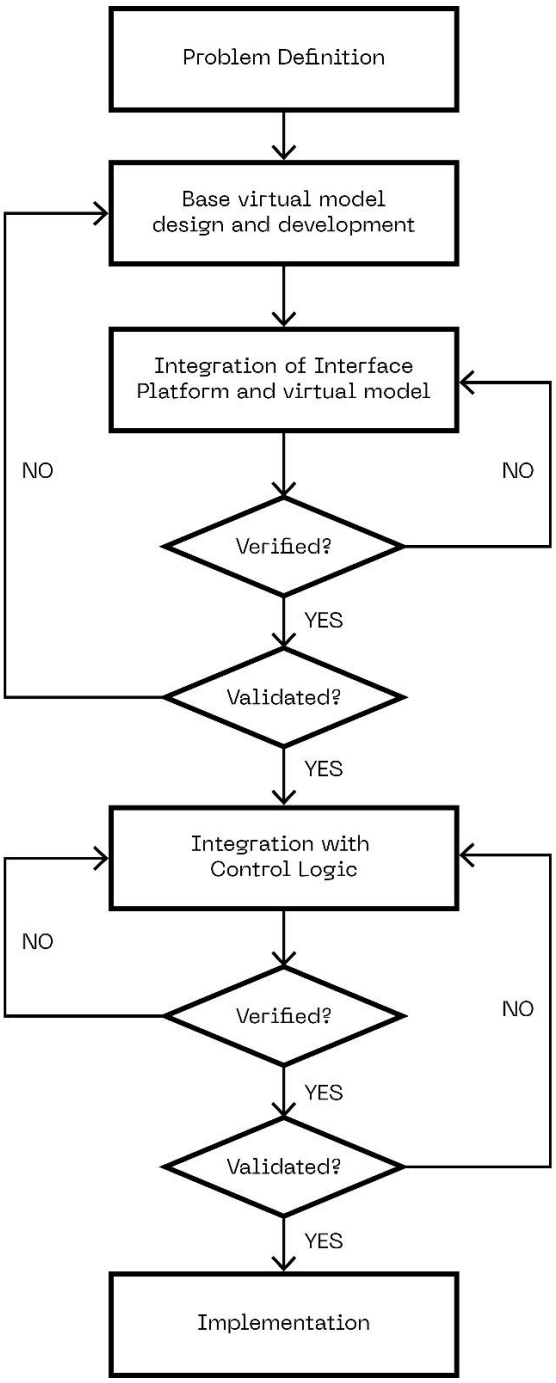


Figure 3.3: Modified HSEM for integrating emulation and virtual model.

3.5 Testing Phase

The testing phase included integration of all software i.e., virtual PLC, behavior model and model. The objective was to integrate all the above-mentioned software together and observe how well this integration works and get the automatic object generation on Plant Simulation to work. The test was to evaluate how well the framework for building an emulation model with an emulated PLC functioned.

Two testing methods were used to test the interconnection between all the software and if the PLC logic is working accordingly. The first was visual, this was a preliminary test conducted to check if the PLC logic is working as it is intended to, this was conducted by first getting TIA portal online by connecting it with PLC sim advanced. This interconnection acted as emulation of PLC, TIA portal is where the PLC logic was written, PLC sim advanced as instance creation. On the other side SIMIT and Plant Simulation were paired and Shared Memory being the means of communication. In Plant Simulation first the data was extracted from Excel to string format, then the emulation was run. The second testing method was a behavioral test to test the behavior of the connection between all the software. This was to make sure that the data flow was bi-directional between all the software.

3.6 Verification and Validation

Verification and validation are two key methods to measure trustworthiness of a research (Schlesinger, 1979). Model validation is defined as "substantiation that a computerized model within its domain of applicability possesses a satisfactory range of accuracy consistent with the intended application of the model" (Schlesinger, 1979) and model verification is defined as "ensuring that the computer program of the computerized model and its implementation are correct" (Schlesinger, 1979). A model should be designed for a very specific purpose and the validity of the model should be determined for that specific purpose (Sargent, 2010). A models accuracy determines if the model is valid for it intended purpose (Sargent, 2010). To identify and potential risks and errors that may occur during the model building and testing, background verification and validation was implemented in every step (Morse, 2002).

Verification and validation were done iteratively on the virtual model throughout the modeling process (as represented in *Figure 3.3*) and also tested the connection between the behavior model in SIMIT and the virtual model in Plant Simulation. The model was further validated by having interviews with supervisors who had previously worked on making the PLC logic. The additional issues and suggestions that were received for improvements from the supervisors and engineers regarding the virtual model were noted down as future work. Once the model was up and running an interview was held to test the entire connection to verify the PLC logic and test if the objects are automatically generating, feedback from all

the engineers working in each of the field was used as a means to improve the framework and develop it further.

4 RESULTS

This section of the report will emphasize the framework of the proposed solution. The documented results will be presented here along with the development process of the solution, integration of the tools utilized, testing of the integrated framework, and then finally the verification and validation process of the complete set-up.

4.1 Framework

The development of the framework was based on the knowledge gained through literature study, qualitative analysis, and multiple interviews with the concerned stakeholders. The entire framework can be simplified into 4 domains, namely, PLC Logic, PLC Emulation, Interface Platform and finally virtual model.

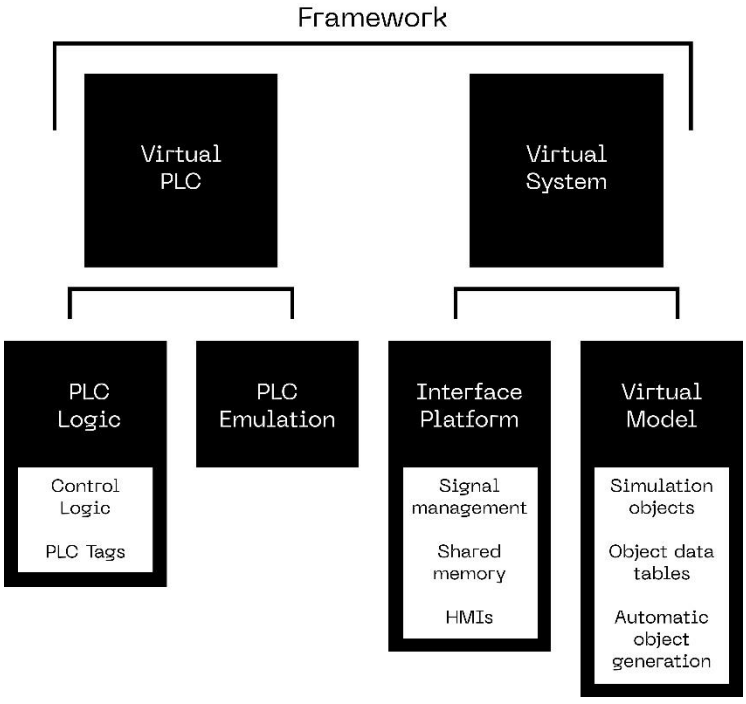


Figure 4.1: Overall Framework.

PLC logic serves as the brain for the entire framework as it consists of the logic control that operates the virtual model. PLC emulation uses the control logic to emulate a virtual PLC. The interface platform is used to manage the I/O signals of the PLC program and finally, the virtual model consists of objects that are controlled by the PLC logic along with other objects that aid the automated object creation function.

The following sections will elaborate more on the tools used, basic requirements of each tool, the integration process, the testing process and finally the validation process to ensure the testing setup functions the way it is framed to.

4.2 Development Phase

Establishing a proper technical setup for testing PLC logic involves integrating several tools and forming an elaborate framework. For the solution in hand, a total of 4 tools have been used to represent their respective domains, each with their purpose of completing the loop and forming a two-way stream of unrestricted signal flow.

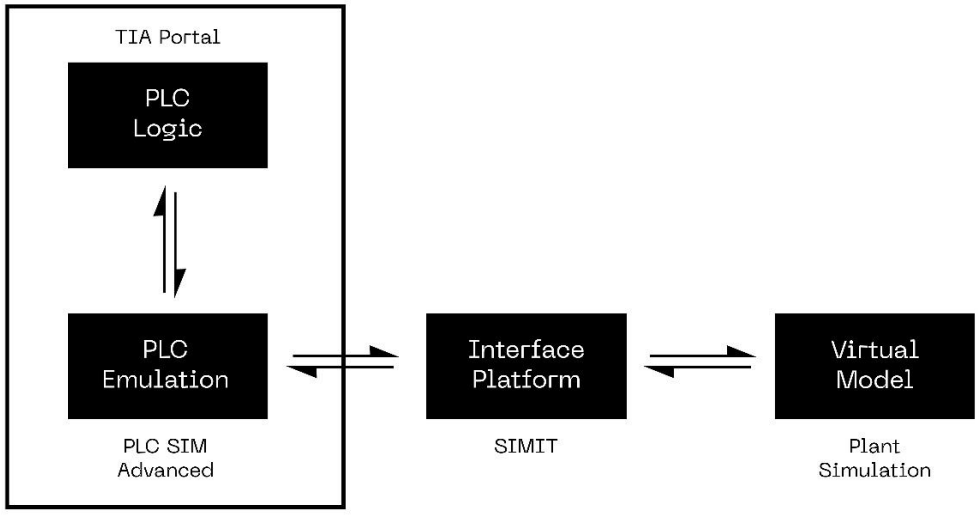


Figure 4.2: Software Framework.

Figure 4.2 illustrates the flow of data between the different domains; the PLC program is created in TIA Portal which is then emulated with the help of SIMATIC S7-PLCSIM Advanced to function as a virtual PLC. The PLC tags from TIA Portal are imported to SIMIT to be managed and receive any changes in the PLC signals. Signals are then “shared” with Plant Simulation in the form of Symbols via Shared Memory. The emulated objects in Plant Simulation then react to the inputs received and send sensor signals back to the PLC program, thus completing the loop.

4.2.1 General Requirements

To build the test environment, the framework must follow a set of general rules and requirements that make the integration process viable.

- The integration process depends heavily on where the tools and software are installed. In this case, the symbols from SIMIT are shared with Plant Simulation through a Shared Memory (SHM), therefore it is preferred all the software and tools are installed on the same system.

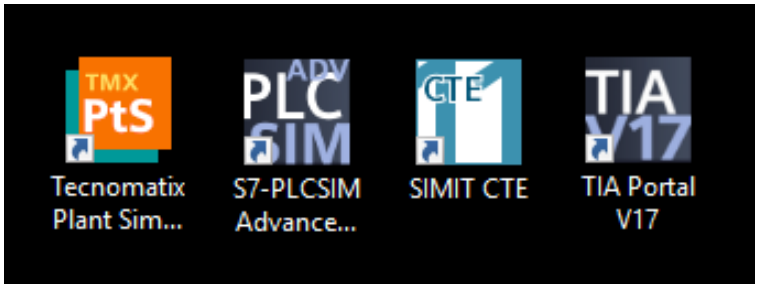


Figure 4.3: Software used in establishing the test environment.

- Plant Simulation must be connected to a valid license server to be able to use the SIMIT interface package. Since the license server used in this case is part of a secured network, a VPN connection needs to be set up to access the server. In this case, Plant simulation uses an educational license.

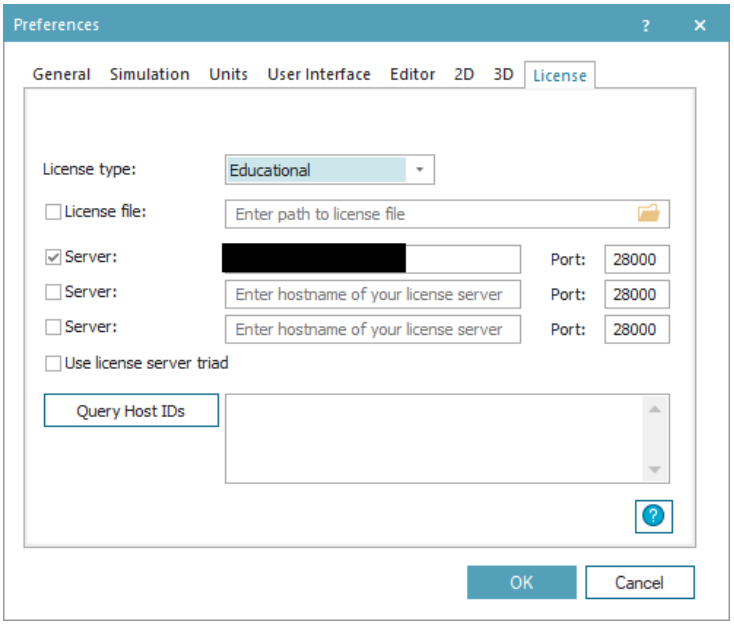


Figure 4.4: License server configuration dialog box in Plant Simulation.

- Since the solution case uses a TIA Portal PLC program that is used in an actual plant, it is crucial to make a copy of the project file with a different name and

under a different directory and to check the TCP/IP connection before making considerable changes or commencing the integration process.

- A uniform naming standard must be established between the tools, to increase interpretability of the signal identifiers and resource data.

4.2.2 PLC Program (TIA Portal V17)

The primary requirement for establishing a virtual commissioning setup is a functioning control logic that can be used to emulate the objects present inside a virtual model. TIA Portal is a tool used to program PLC logic to be used by a physical or a virtual PLC to control a given system. The PLC program operates on input signals and sends out output signals. The I/O signals of the PLC program use identifiers known as PLC tags and these tags are created based on the data types of their respective signals.

The TIA Portal project used in this case is from an earlier version of the software, TIA Portal V17 to be exact. Hence it is important to have the correct version of the software installed to run the PLC program without any issues.

It is important to note that either the absence or blurring of images in this section is to protect the anonymity of the project file, as it is considered classified information by Volvo Cars Corporation

4.2.2.1 PLC Tags

PLC tags are variables created with a specific data type which also determines its memory allocation. These tags help carry signal values across the PLC program or even in and out of it in the form of I/O signals. Identifying the PLC tags that directly play a role in controlling the virtual model is crucial to understanding the functioning of the PLC logic. In the testing phase, it becomes evident how the output from the virtual model directly reflects in the PLC program.

	Name	Tag table	Data type	Address
91	310CL01M1SG1:12	Standard I/O	Bool	%I303.0
92	310CL01M1SG6:12	Standard I/O	Bool	%I303.1
93	310CL01M1SG7:4	Standard I/O	Bool	%I303.2
94	310CL01M1SG8:4	Standard I/O	Bool	%I303.3
95	310CL01M1SG13:12	Standard I/O	Bool	%I303.4
96	SPARE_I303.5	Standard I/O	Bool	%I303.5
97	SPARE_I303.6	Standard I/O	Bool	%I303.6
98	SPARE_I303.7	Standard I/O	Bool	%I303.7
99	310CL01M1QH1:14	Standard I/O	Bool	%I304.0
100	310CL01M1SG2:4	Standard I/O	Bool	%I304.2

Figure 4.5: PLC tags from the program.

4.2.2.2 PN/PN Couplers

The PLC logic was programmed in-house at Volvo Cars to control an actual line in the production plant. Therefore, the logic not only depends on the input signals from the lines but also on the preceding and succeeding lines controlled by other PLCs. In other words, the control logic of all the relevant PLCs works together in harmony to make the system work. The scope of this work deals with the logic programmed for PLC “PL175001” therefore its respective logic must be isolated from the I/O signals of the neighbouring PLCs’ logic.

PN/PN Coupler is a coupling between two PROFINET controllers, such that the I/O signals of one controller can be used by another. The PN/PN couplings of the PLC “PL175001” must be tethered in a way that there are no external influences on the controller but also the PLC functions in accordance with the PLC logic programmed in TIA Portal. It is also worthwhile to note that TIA Portal requires a login password to be able to modify or delete any PLC tags or safety blocks.

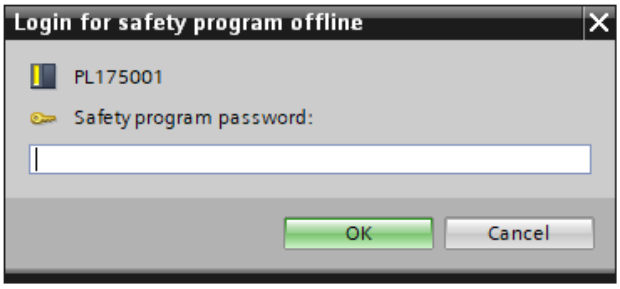


Figure 4.6: Login dialog box.

4.2.2.3 Safety Signals

Safety signals are used to ensure that the environment is safe for both the equipment and the personnel involved. Since the PLC tags carrying signals coming in and out of PN//PN couplers are either deleted or tethered from the PLC logic, the safety function blocks don’t function properly as they do not receive the required signals from the neighbouring PLCs. In turn, the safety blocks set off alarms, restricting the ability to put the system in auto-operation mode. Therefore, as an added step, foreign safety-related signals are handled in a way such that they do not set off any alarms.

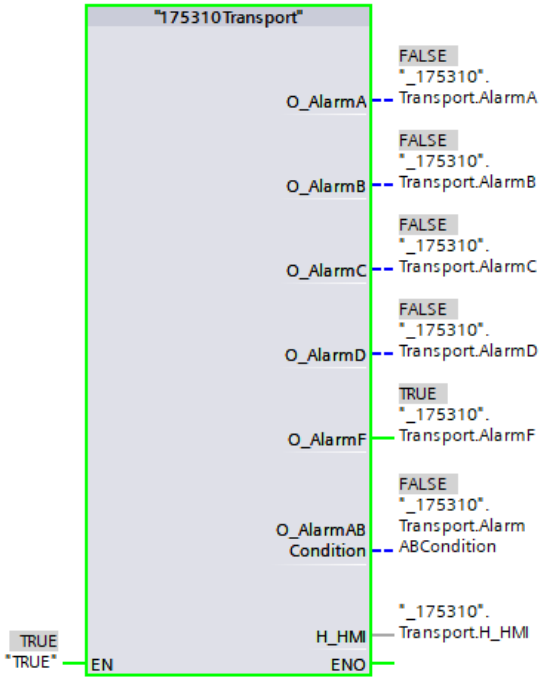


Figure 4.7: Transport alarm signals of a conveyor.

Once the PN/PN Couplers and Safety signals are “cleaned up”, and the relevant PLC tags are identified, the interface platform (SIMIT) can be set up to manage, filter and forward the signals to the virtual model.

4.2.3 Interface Platform (SIMIT 11.1)

Often, the control logic consists of I/O signals that are crucial for the operation of the system however, they could be irrelevant to the scope of the virtual model, for example, safety signals and cabinet signals to name a few. One of the many purposes of SIMIT in this case is to function as a filter to manage the I/O signals from the PLC program such that only the relevant signals are passed on to the virtual model.

The SIMIT model provided by VCC was highly comprehensive and was modelled specifically to work with the control logic of the PLC “PL175001”. As far as the functionality of the SIMIT model extends, the PN/PN coupler signals were not accounted for, hence the “cleaning up” process in the previous section is a mandatory step for the SIMIT model to function in accordance with the PLC program.

The PLC tags from the program were already imported and assigned to the respective drives, motor starters, HMIs and so on in the SIMIT model. Therefore, the output signals such as speed can be sent to the conveyors and signals such as Status words and HMI controls can be sent back to the PLC program as inputs.

For the virtual model to run on the PLC program, the signals from the control logic must be processed into signals that can be used by the simulation objects. In this case, the SIMIT model emulates electrical components such as drives and motor starters by processing input signals like SafetyOK and Control words and converting them into speed signals that can be directly used by the roller beds in the virtual model. In the absence of SIMIT, the virtual model would require additional logic to comprehend the input signals from the control program, therefore SIMIT eliminates the hassle and provides the virtual model with only the necessary data.

4.2.3.1 Contec Library

The SIMIT model provided by Volvo Cars came with its own pre-built control chart. The control chart consists of simulation components such as conveyors and lifts that are arranged in an orderly fashion that resembles the actual line. Contec library is an add-on available for SIMIT that provides these simulation components which are highly customizable according to the user's needs and requirements. The scope of the project involves emulating the single roller beds in a conveyor line. The line chosen in this case (Line 330) consists of 9 roller beds. Hence, to create objects in the virtual model, it is important to acquire data related to the roller beds such as length, width, the number of sensors present in the conveyor and so on.

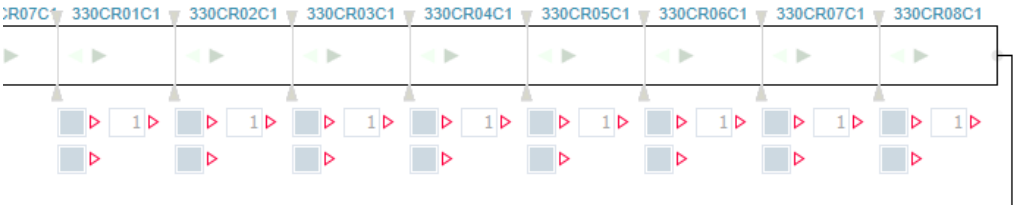


Figure 4.8: Control chart with Contec library components.

The single roller beds in the control charts can be clicked to view their attributes in the "Properties" tab, the physical attributes can be found inside the "General" subtab and the number of sensors can be viewed by going into the "Output" subtab.

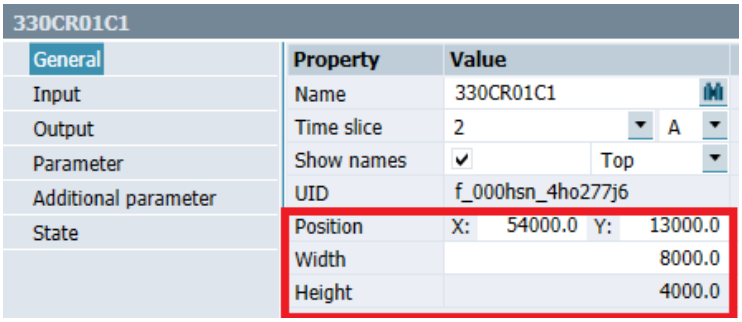


Figure 4.9: General properties of a conveyor displaying its total width and length.

The data of all the single roller beds in the conveyor line is collected and documented in an Excel sheet to be used with Plant Simulation; this will be explained in depth under in Chapter 4, section 4.3.3 (Plant Simulation).

A default Contec conveyor object will have a maximum of 8 sensors as shown in *Figure 4.10*, however, it was found that only one of them is the actual stop sensor, and the others don't serve a purpose in this case. Whenever an object passes one of these sensors, the sensor passes the signal as an input to the PLC program in TIA portal. The relative location of the sensors with respect to the length of the conveyor too is obtained from the "Output" subtab of the conveyor object, thus, the location of every sensor is obtained and documented in an Excel file. The Excel tables comprising conveyor and sensor data are included in the Appendix.

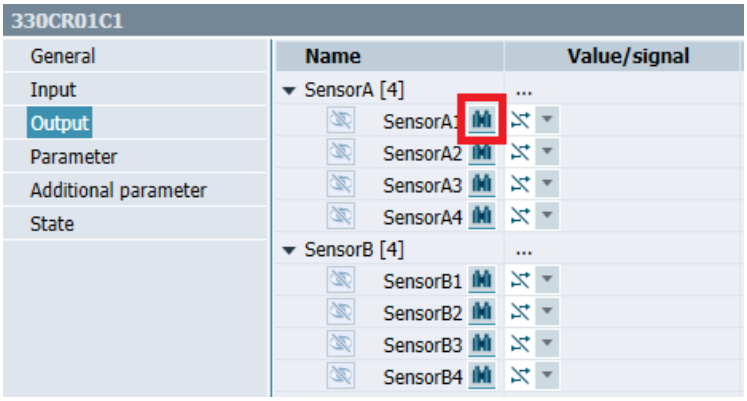


Figure 4.10: Sensors in a conveyor.

The Contec objects are fed input values like speed and in turn, the sensors return output signals. Since Plant Simulation is used for virtual model creation and, SIMIT for signal management, the signals received by the conveyor objects in the control charts must be redirected to the virtual model, similarly, the sensor signals must be returned by the objects in the emulation plant and not by the control chart components. Therefore, to achieve this, the speed and stop sensor signals are tracked back using the cross-reference function (binoculars button) as shown in *Figure 4.10* to their respective modules and the tags are modified in a way such that the modules receive input signals and send output signals respectively to the virtual model.

4.2.3.2 Shared Memory (SHM)

Once the conveyor data is documented and the signals to be redirected are identified, a Shared Memory (SHM) must be established to share the I/O signals with the virtual model. Since all the tools used in the loop are installed in the same system, SHM is a better and faster option than setting up an online server to share the signals with the virtual model.

Setting up a SHM inside SIMIT involves three steps.

- Creating a SHM coupling
- Renaming the SHM coupling
- Creating symbols

Creating a SHM is done by double-clicking the “New coupling” option under Project Navigation and choosing “Shared Memory”. This will create an unnamed SHM under Couplings.

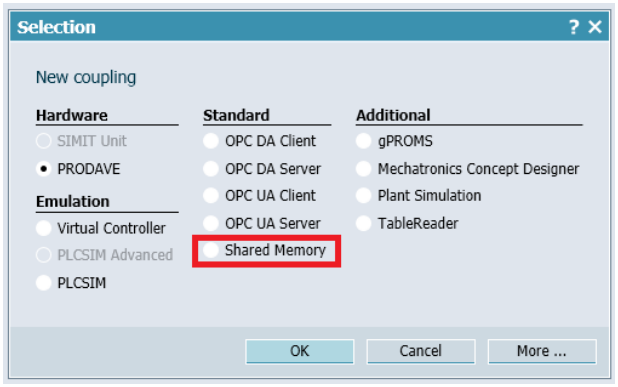


Figure 4.11: New coupling window.

For Plant Simulation to identify the SHM created in SIMIT, a name must be given to the SHM. Something to note is that SHM naming is highly case-sensitive. In this case, the SHM is named “PlantSimulation”, along with the name, the Mutex Name too must be changed to “PlantSimulationMutex” for the SHM to properly function.

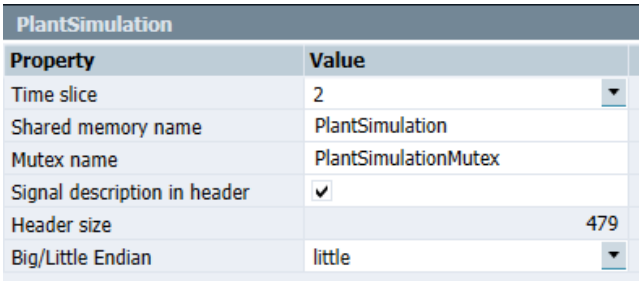


Figure 4.12: Shared Memory properties window.

The next two sub-sections will cover the final step of establishing a working SHM, defining the symbols that need to be shared across the SHM with Plant Simulation.

4.2.3.3 Symbol Naming Standard

Symbols, like PLC tags used in TIA Portal, are variables whose values change based on the operands. As mentioned in Chapter 4, section 4.2.3.1 (Contec Library), for the I/O signals to be redirected to the virtual model, custom symbols are created inside the SHM. These symbols are then shared across the SHM to the virtual model to operate the objects within the model.

For Plant Simulation to understand the purpose of a signal and to identify the simulation object corresponding to the signal, it is mandatory to maintain a naming standard while naming the symbols. In section 4.2.3.1, the signals to be redirected were identified and classified as speed and stop sensor signals, while speed signals function as input data, and the stop sensor signals function as output data from the virtual model. The ID (name) of the objects can be identified through the Contec components.

After acquiring all the required information, the symbols are named using the following format:

Object Type_Object Name_Signal type

With the naming standard established, Plant Simulation can understand, the type of object for which the signal is meant for, the object's ID, and finally, which attribute of the object, the signal is meant for.

Figure 4.13: Example of the symbol naming standard.

Figure 4.13 presents an example of the naming standard used for simulation objects, in this case, "CR" stands for conveyor, "330CR01C1" stands for the name of the roller bed and finally "Speed" stands for the signal type, which Plant Simulation then utilizes to identify if the signal is an input or an output signal. As the scope of this project covers only a single line, only single roller beds were created, as a result, all the emulated objects have the prefix of "CR".

4.2.3.4 Signal Data Type and Memory Allocation

SHM requires additional information upon the creation of the symbols, such as the "Data type" and "Address" of the same. Symbols too have data types similar to PLC tags, in this use case, there are just two types of signals stop and speed, with the former being the output and the latter being the input.

It is known that stop signals can only have two values (True or False), whereas speed signals can have any value within a set range, therefore, it becomes mandatory to declare a data type for the created symbols. Assigning a data type to a sensor signal also comes with the hassle of allocating memory space for the symbol inside the SHM. To understand the memory allocation process, it is important to know the memory requirement of each data type. The stop sensor signals are of Boolean type and therefore, require 1 Bit of space, on the other hand, the speed signals carry data of Real type, thus requiring 4 Bytes of space.

Data type	Size	Notation	Value range
BOOL	1 bit	M<byte>.<bit>	True/False
BYTE	1 byte (8 bits)	MB<byte>	0 ... 255 or -128 ... +127
WORD	2 bytes	MW<byte>	0 ... 65,535
INT	2 bytes	MW<byte>	-32,768 ... 32,767
DWORD	4 bytes	MD<byte>	0 ... 4,294,967,295
DINT	4 bytes	MD<byte>	-2,147,483,648 ... 2,147,483,647
REAL	4 bytes	MD<byte>	$\pm 1.5 \times 10^{-45}$ to $\pm 3.4 \times 10^{38}$

Figure 4.14: Data types and their memory allocation (Siemens AG, 2020).

Once the data type is entered in the SHM, it is time to allocate memory addresses to the symbols with respect to their data types. Addressing the symbols is a sequential process as the symbols are addressed one after another to avoid confusion. As shown in *Figure 4.15*, it is important that memory addresses allocated to the symbols do not overlap.

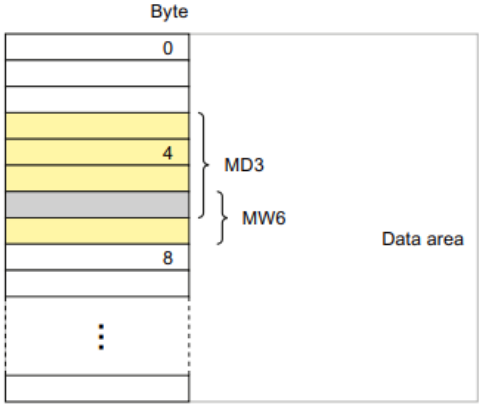


Figure 4.15: Memory allocation overlap (Siemens AG, 2020).

Once the addresses are assigned to the symbols, the Shared Memory is ready to go online and be accessed by Plant Simulation.

PlantSimulation (SHM)

Inputs Reset filter

Default	Symbol name	Address	Data type	Comment
0.0	CR_330CR01C1_Speed	MD4	REAL	Roller Bed Speed
0.0	CR_330CR02C1_Speed	MD8	REAL	Roller Bed Speed
0.0	CR_330CR03C1_Speed	MD12	REAL	Roller Bed Speed
0.0	CR_330CR04C1_Speed	MD16	REAL	Roller Bed Speed
0.0	CR_330CR05C1_Speed	MD20	REAL	Roller Bed Speed
0.0	CR_330CR06C1_Speed	MD24	REAL	Roller Bed Speed
0.0	CR_330CR07C1_Speed	MD28	REAL	Roller Bed Speed
0.0	CR_330CR08C1_Speed	MD32	REAL	Roller Bed Speed
0.0	CR_330CR09C1_Speed	MD36	REAL	Roller Bed Speed

Outputs Reset filter

Symbol name	Address	Data type	Comment
CR_330CR01C1_Stop	M0.0	BOOL	Roller Bed Stop Sensor Signal
CR_330CR02C1_Stop	M0.2	BOOL	Roller Bed Stop Sensor Signal
CR_330CR03C1_Stop	M0.4	BOOL	Roller Bed Stop Sensor Signal
CR_330CR04C1_Stop	M0.6	BOOL	Roller Bed Stop Sensor Signal
CR_330CR05C1_Stop	M1.0	BOOL	Roller Bed Stop Sensor Signal
CR_330CR06C1_Stop	M1.2	BOOL	Roller Bed Stop Sensor Signal
CR_330CR07C1_Stop	M1.4	BOOL	Roller Bed Stop Sensor Signal
CR_330CR08C1_Stop	M1.6	BOOL	Roller Bed Stop Sensor Signal
CR_330CR09C1_Stop	M2.0	BOOL	Roller Bed Stop Sensor Signal

Figure 4.16: Shared Memory after creating the symbols.

Once SHM is set up, there is one last thing to be done before the SIMIT project can go online, and that is to redirect the signals to the virtual model by tethering the Contec Components inputs and output and substituting them with the SHM symbols as mentioned back in Chapter, section 4.2.3.1 (Contec Library). This is done by cross-referencing signals for all 9 roller beds in the conveyor line, accessing the respective drive modules and deleting the existing connection to the Contec components and connecting the respective symbols to the modules.

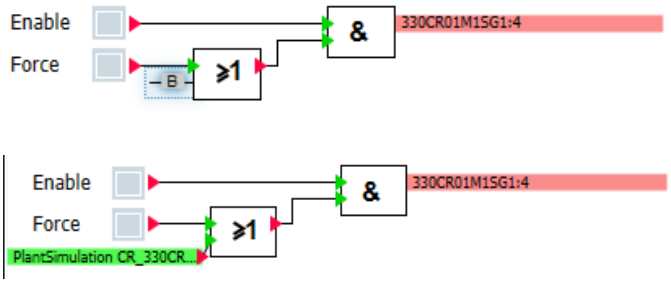


Figure 4.17: Stop Sensor input before (above) and after(below) substituting with SHM symbol respectively.

4.2.3.5 HMIs

The SIMIT project came with an array of HMIs modelled by Volvo. HMIs are used to set the corresponding conveyor lines to Auto mode by pressing the "Start" button, as shown in *Figure 4.18*. Despite the TIA Portal project housing in-built HMIs for the lines, the HMIs in the SIMIT project make it much simpler and easier to operate. The HMI for the line "330AS1JX1" will be used in this case.

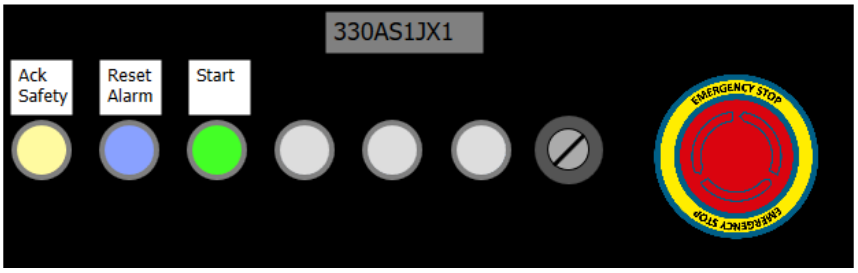


Figure 4.18: HMI of the line 330.

With the Shared Memory set up, SIMIT was ready to function as the interface platform within the setup. It was also in this state, that the work faced a lot of delay and most of it was due to license acquisition for the Contec library and updating the SIMIT software from version 10.2 to version 11.1.

4.2.4 Virtual Model (Plant Simulation)

Plant Simulation can be considered as the end of the line in the framework, which houses the virtual model. The goal here is to use a control logic from a PLC program to emulate objects in the Plant Simulation model instead of creating a control logic inside the model. The emulated objects operate on the input received from the PLC program and in turn, send output values back to the program.

4.2.4.1 Frames

The complete virtual model can be classified into three frames, namely, model frame, method frame and table frame. Each frame has its assigned purpose, from storing acquired data to generating simulation objects in the model frame.

i. **Model Frame:**
The model frame consists of simulation objects along with other objects such as the event controller and SIMIT object. It simply provides space for the methods to

generate simulation objects in it. Basic controls that operate the model such as Stop, Reset, and Delete can also be found in this frame.

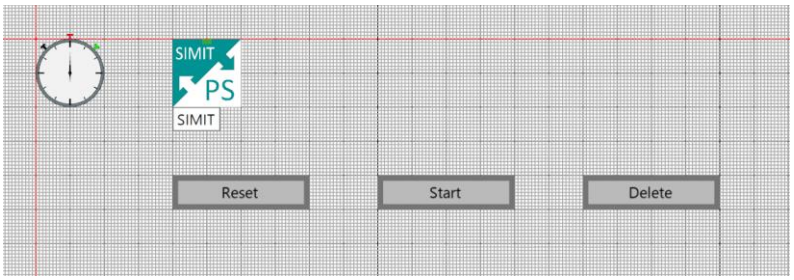


Figure 4.19: Model Frame.

ii. **Methods Frame:**

The method frame can be considered the backbone of the model as it controls the vital functions of the model. It consists of methods to acquire data from Excel sheets and SIMIT objects, cross-check SIMIT data with resource conveyor data, generate objects automatically, reset conveyor sensors and finally delete all simulation objects.

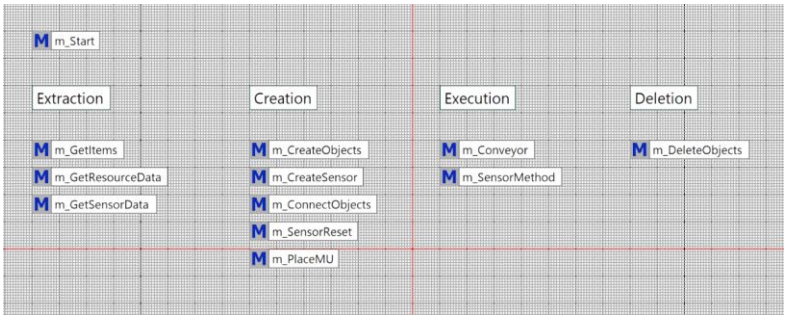


Figure 4.20: Method Frame.

iii. **Table Frame:**

The Table frame Consists of all the data tables storing the data acquired from different sources. Data required by the method frame to generate simulation objects is accessed from the data tables in the frame.

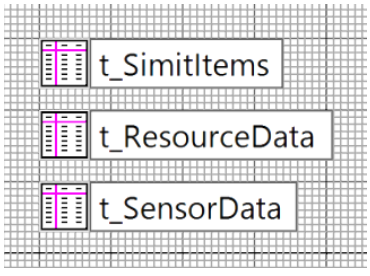


Figure 4.21: Table Frame.

4.2.4.2 Object Naming Standard

For Plant Simulation to recognize and distribute the incoming signals, the symbols require a naming standard, similarly, the objects inside the virtual model too require a naming standard such that the signals reach the respective objects successfully.
The symbol names in SIMIT follow the following nomenclature:

"Object Type"_"Object Name"_Signal type

In SIMIT "Object ID" refers to the name of the roller beds, however, in Plant Simulation, object names cannot begin with a number, therefore, "Object Type" is used as a prefix to the actual object name.
The final naming standard for objects in the virtual model is:

"Object Type"_"Object Name"

Figure 4.22: Example of simulation object naming standard.

Figure 4.22 presents an example of the naming standard used for simulation objects, in this case, "CR" stands for conveyor and "330CR01C1" stands for the name of the roller bed.

The naming standard in the virtual model extends to all the other objects as well, such as data tables, methods and even frames. In this case, instead of developing a new naming standard, the standard developed by Volvo was employed. The name of the object starts with a prefix defining the object followed by a separator, an underscore "_" in this case, and then finally the object name. However, for frames, the standard is a bit different as the naming starts with an underscore, followed by the name of the frame. The naming standard has been followed uniformly throughout the virtual model with no exceptions.

Object Type	Naming Standard	Prefix
Frames	_"Frame name"	"_"
Methods	m_"Method name"	"m_"
Tables	t_"Table name"	"t_"

Table 4.1: Name Standards for non-simulation objects in Plant Simulation.

4.2.4.3 Shared Memory

Once the Shared Memory coupling is created inside SIMIT, it becomes accessible by Plant Simulation. However, to access it, a SIMIT object is placed in the model frame. To be able to use the SIMIT object from the information flow tab, it is mandatory to enable the SIMIT interface package inside the Manage Class Library window.

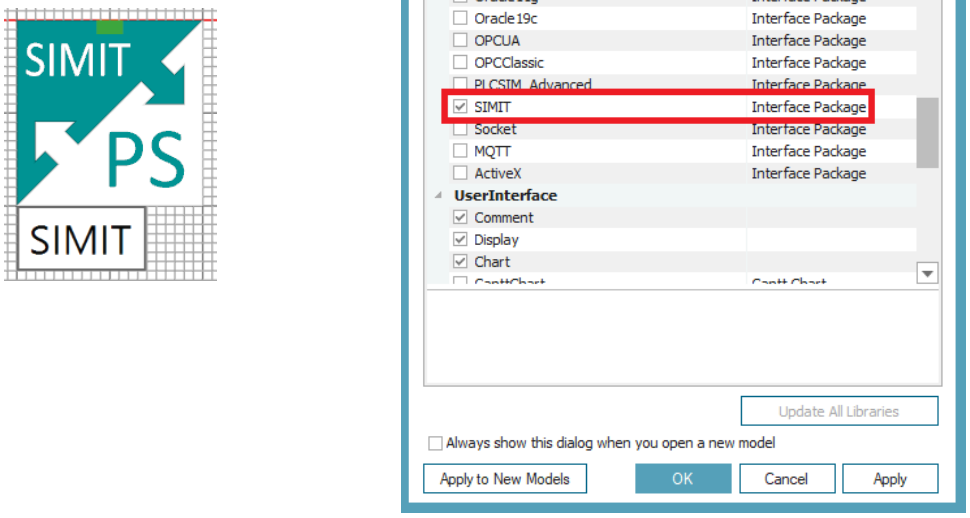


Figure 4.23: SIMIT Object (Left); Manage Class Library window (Right).

Upon enabling the SIMIT Interface package inside the Manage Class Library window, the SIMIT object becomes accessible under the information flow tab. The SIMIT object is then placed in the model frame. Upon opening the SIMIT object, we are given a field to input the name of the SHM and as mentioned in Chapter 4, section 4.2.3.2 (Shared Memory), the name is “PlantSimulation”, and it is very important to remember that the naming is highly case sensitive. Once the name is typed into the field, the “Active” option is checked, this step connects Plant Simulation to the SHM successfully (Note: The SIMIT simulation must be started for this step to work). *Figure 4.23* shows the different actions available in the SIMIT object, it can be noticed that there are three buttons in the object window, namely “Items”, “Import Items” and “Show Item Values”. Firstly the “Import Item Values” button is clicked, this imports all the symbols created inside the SHM coupling in SIMIT.

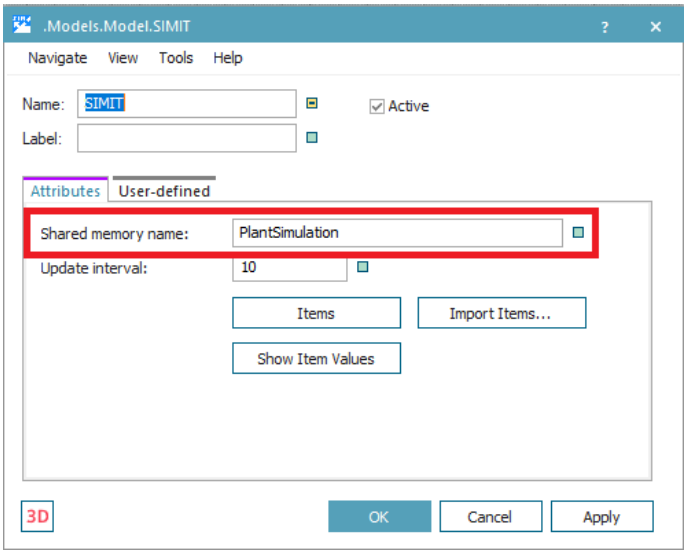


Figure 4.24: SIMIT object window.

To check if all the symbols have been imported successfully into Plant Simulation, the “Items” can be clicked, this brings up a data table, consisting of all the symbol names, and their data types and even classifies the signals as Input or output based on the symbol creation in SIMIT. There is however another column inside the “Items” data table with the header “Changed-Value Control”, in the fields under this column, the user has the freedom to input the name of a method, when the value of the corresponding symbol changes, Plant Simulation automatically calls the mentioned method. In this case, as and when the speed value changes, the method “m_Conveyor” is called to distribute the speed value to respective roller beds.

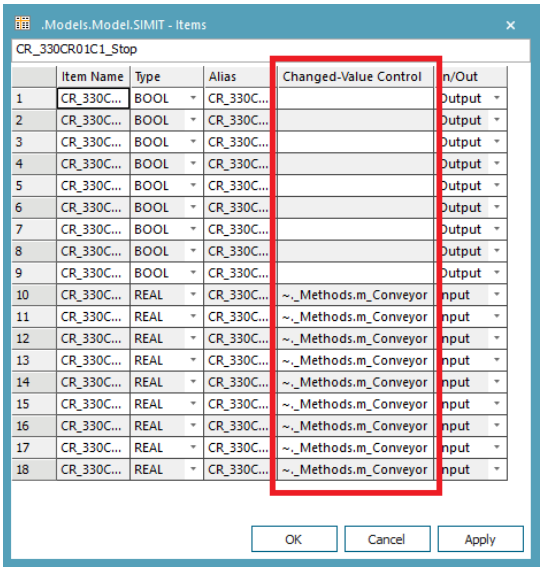


Figure 4.25: Items table with the Changed-Value Control methods assigned to relevant signals.

The third option “Show Item Values” displays the values of the symbols imported into Plant simulation in real-time as shown in *Figure 4.26.*

.Models.Model.SIMIT - Items

CR_330CR01C1_Stop

string	string 1 Alias	string 2 Type	string 3 Value
1	CR_330CR01C1_Stop	BOOL	FALSE
2	CR_330CR02C1_Stop	BOOL	FALSE
3	CR_330CR03C1_Stop	BOOL	FALSE
4	CR_330CR04C1_Stop	BOOL	FALSE
5	CR_330CR05C1_Stop	BOOL	FALSE
6	CR_330CR06C1_Stop	BOOL	FALSE
7	CR_330CR07C1_Stop	BOOL	FALSE
8	CR_330CR08C1_Stop	BOOL	FALSE
9	CR_330CR09C1_Stop	BOOL	FALSE
10	CR_330CR01C1_Speed	REAL	0.000000
11	CR_330CR02C1_Speed	REAL	0.000000
12	CR_330CR03C1_Speed	REAL	0.000000
13	CR_330CR04C1_Speed	REAL	0.000000
14	CR_330CR05C1_Speed	REAL	0.000000
15	CR_330CR06C1_Speed	REAL	0.000000
16	CR_330CR07C1_Speed	REAL	0.000000
17	CR_330CR08C1_Speed	REAL	0.000000
18	CR_330CR09C1_Speed	REAL	0.000000
19			

Figure 4.26: "Show Item Values" table.

4.2.4.4 Methods

Moving on to the frame consisting of methods, the complete method pool can be sub-classified into 4 different groups, namely, extraction, creation, execution and then finally deletion.

i. Extraction:

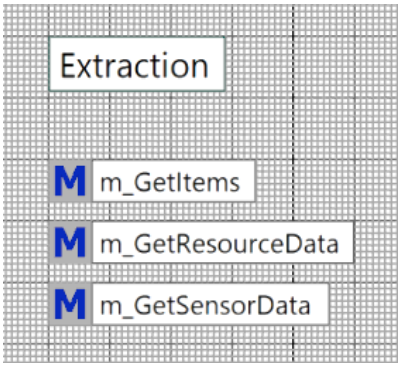


Figure 4.27: Data Extraction methods.

The methods under the Extraction group help acquire data from different sources and import them into Plant Simulation. The methods in this group are run before the emulation process is started.

The Extraction group houses 3 methods, m_GetItems, m_GetResourceData and m_GetSensorData. The three methods can be run in any sequence. The first method, m_GetItems however should only be run after importing the items from SHM through the SIMIT object, as the method acquires symbol data from the “Import Items” table and copies it to another data table in the tables frame. The second method, m_GetResourceData imports the conveyor data stored in the Excel sheet as mentioned back in Chapter 4 section 4.2.3.1 (Contec Library). Finally, the third method imports the sensor data corresponding to each roller bed from an Excel sheet like conveyor data. It is important to note that the Excel file must be saved in the same directory as the Plant Simulation file.

The Excel sheets have been included in the appendix section.

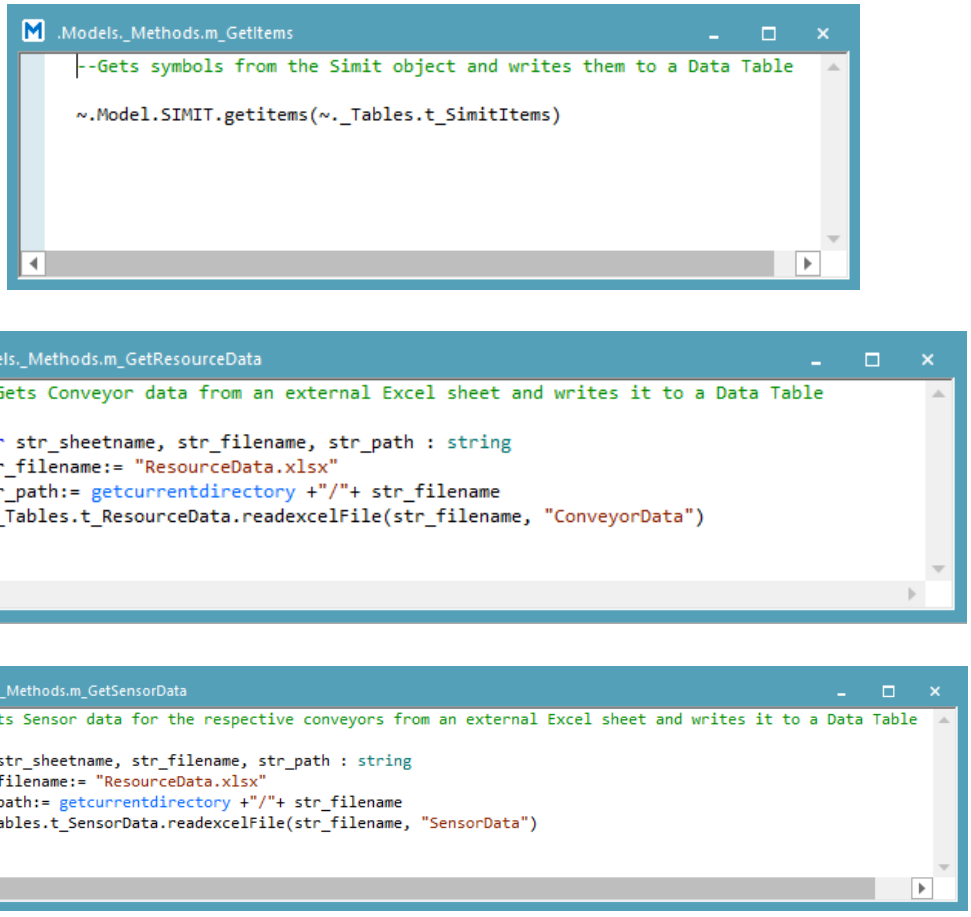


Figure 4.28: m_GetItems (Top); m_GetResourceData (Middle); m_GetSensorData (Bottom).

ii. Creation:

The Creation group is a set of 5 methods, each focused on creating and connecting objects based on data acquired through the methods in the Extraction group.

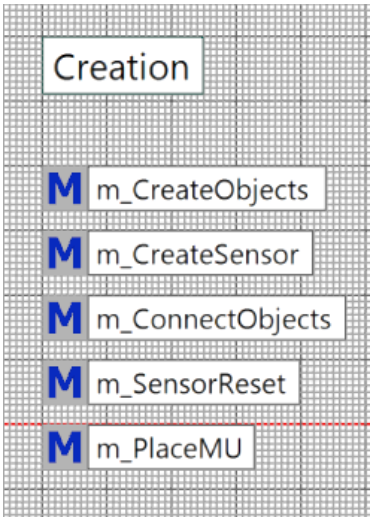


Figure 4.29: Object Creation methods.

Unlike the methods in the Extraction group, the Creation methods must be run in a set sequence as the methods are highly dependent on the preceding methods. Another requirement for the Creation methods to work properly would be, to have already run the Extraction methods, since the objects cannot be created if Plant Simulation does not possess sufficient data to do so. The methods in this group are run before the start of the emulation process.

Since object generation is a very comprehensive and lengthy process in its regard, it will be discussed in Chapter 4, section 4.2.4.6. (Automated Object Generation).

iii. Execution:

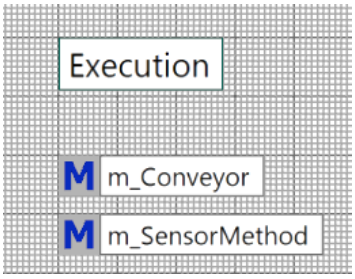


Figure 4.30: Execution methods.

The Execution methods do not have to be executed manually as they are automatically called by the simulation objects upon starting the emulation. The first method “m_Conveyor” is called by Changed-Value Control through the SIMIT object, which was assigned back in Section 4.4.3.3.

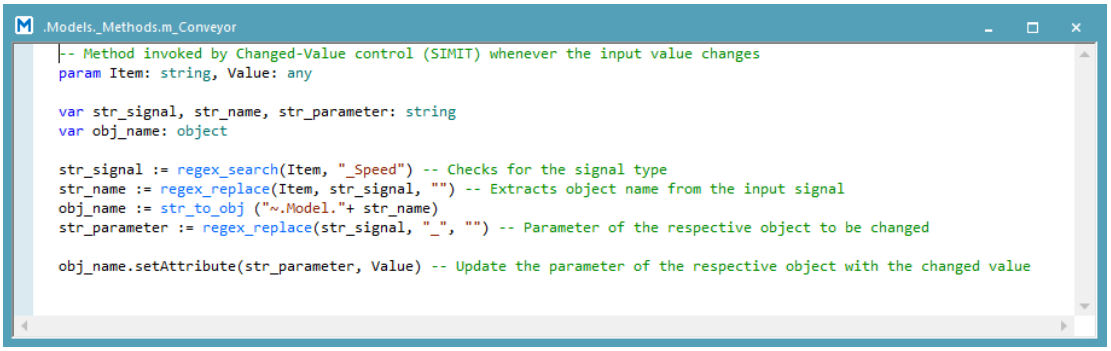


Figure 4.31: m_Conveyor method.

The m_Conveyor method is manually assigned to the Changed-Value Control of speed symbols, as they are the only input values received by the virtual model through Shared Memory. Therefore, in the event of a change in the value of a speed signal, m_Conveyor is called. The method takes the name of the symbol (string) and its value (any) as inputs, then extracts the roller bed's name from the string and finally assigns the input value as the speed of the roller bed in the virtual model.

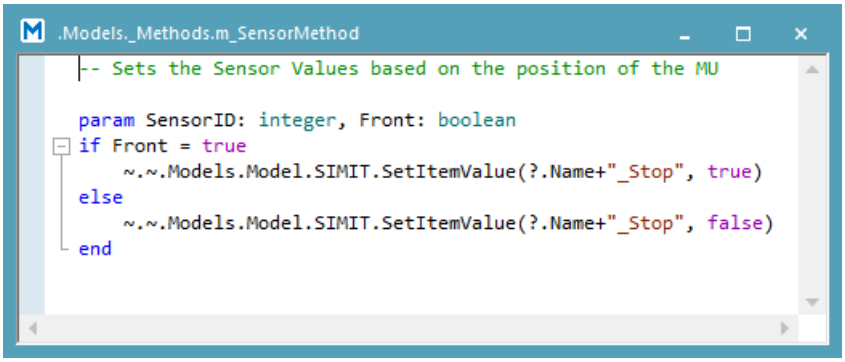


Figure 4.32: m_SensorMethod method.

Moving on to the second method, m_SensorMethod changes the value of the stop signals whenever a Mobile Unit (MU) activates the stop sensor of a roller bed. The method accepts the sensor ID and the Boolean value of the sensor as inputs, then sets the corresponding roller bed's stop signal to either "TRUE" or "FALSE".

iv. Deletion:

The deletion process is executed post emulation, to delete all simulation objects in the model frame. The Deletion group consists of only one method, m_DeleteObjects, when run, the object deletes all objects except the SIMIT object and the buttons used to execute these methods.

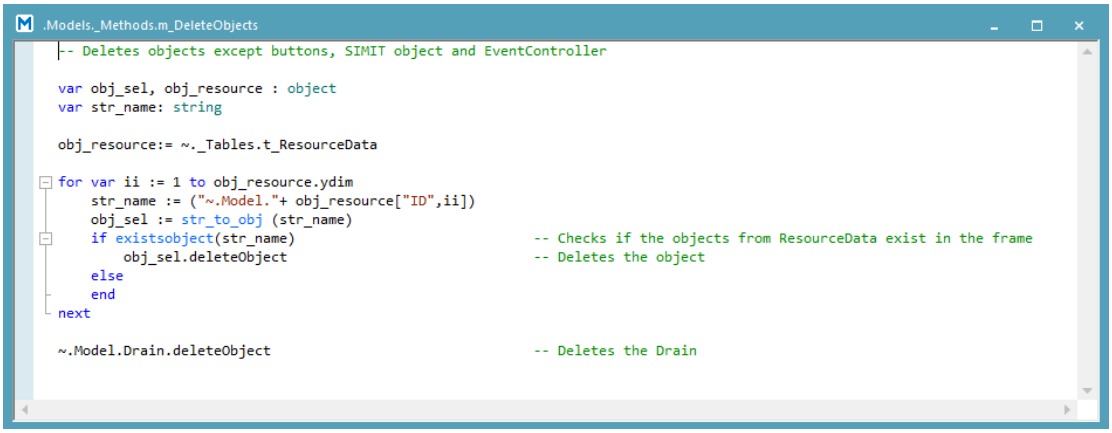


Figure 4.33: m_DeleteObjects method.

To reduce the hassle of running each method manually at the start of every emulation, specific buttons that run the methods in a given sequence have been added to the model frame, such that objects can either be created or deleted with a single click.

4.2.4.5 Tables

The table frame consists of three tables, namely, t_GetItems, t_GetResourceData and t_SensorData. As mentioned in the previous section, the methods in the Extraction methods group import data to their respective data tables. Both t_GetResourceData and t_SensorData use the same Excel file but different sheets to acquire data on conveyors and their sensors. The Excel file name however must be manually updated inside both m_GetResourceData and m_GetSensorData methods for Plant Simulation to identify the file.

	string 1	string 2	real 3	real 4	real 5	real 6	real 7
string	ID	Type	PosX	PosY	Direction	Length	Width
1	CR_330CR01C1	Conveyor	0.00	200.00	0.00	8.00	4.00
2	CR_330CR02C1	Conveyor	160.00	200.00	0.00	8.00	4.00
3	CR_330CR03C1	Conveyor	320.00	200.00	0.00	8.00	4.00
4	CR_330CR04C1	Conveyor	480.00	200.00	0.00	8.00	4.00
5	CR_330CR05C1	Conveyor	640.00	200.00	0.00	8.00	4.00
6	CR_330CR06C1	Conveyor	800.00	200.00	0.00	8.00	4.00
7	CR_330CR07C1	Conveyor	960.00	200.00	0.00	8.00	4.00
8	CR_330CR08C1	Conveyor	1120.00	200.00	0.00	8.00	4.00
9	CR_330CR09C1	Conveyor	1280.00	200.00	0.00	3.00	4.00
10							

	string 1	integer 2	real 3	string 4
string	ID	SenNumber	StopSensorPos	SensorPosition
1	CR_330CR01C1	1	6.30	Length
2	CR_330CR02C1	1	6.30	Length
3	CR_330CR03C1	1	6.30	Length
4	CR_330CR04C1	1	6.30	Length
5	CR_330CR05C1	1	6.30	Length
6	CR_330CR06C1	1	6.30	Length
7	CR_330CR07C1	1	6.30	Length
8	CR_330CR08C1	1	6.30	Length
9	CR_330CR09C1	1	2.00	Length
10				

Figure 4.34: t_ResourceData (above); t_SensorData (below).

4.2.4.6 Automated Object Generation

Under the Creation group, a set of methods was written to create simulation objects based on the data found in the table frame. Automated object generation is one of the key functions of the virtual model in this case, as it also serves as a means to verify the connection between SIMIT and Plant Simulation in Chapter 4, section 4.6 (Verification and Validation).

i. Conveyor and Sensor Creation:

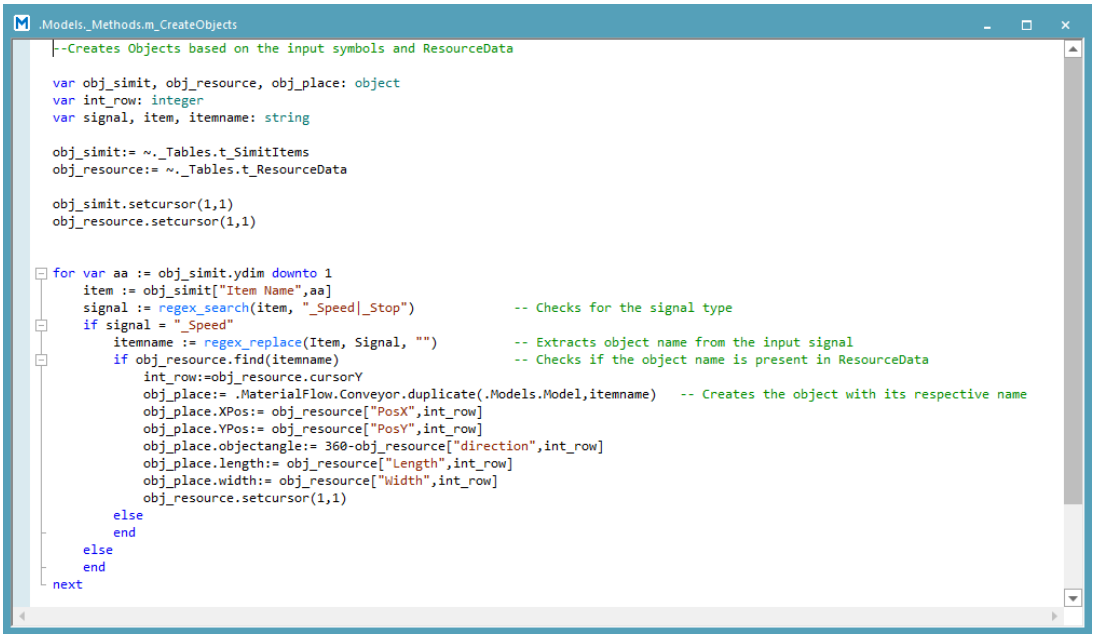


Figure 4.35: m_CreateObjects methods.

The method sequentially goes through the list of imported signals, if the signal has the "_Speed" suffix, the method extracts the object name from the symbol, and the extracted name is then Cross-referenced with the t_ResourceData data table. If the object name matches any of the object IDs in the data table, the method acquires other data relating to the conveyor such as its position and dimension and then places the conveyor in the model frame. This process repeats itself until the method reaches the end of the Symbols data table. To avoid the loss of MUs at the end of the line a drain was also created at the end of the line that represents the succeeding line in the plant. A separate line of code was used for the creation of the drain, the imported symbols do not have a specific symbol meant for a drain.

Similarly, once the conveyors are placed, dimensioned and positioned in the model frame, the sensors have to be placed on the conveyors, for which another method was written, m_CreateSensor, this method goes through the imported symbols list and

looks for the suffix "_Stop" if a symbol is found with the same suffix, the method then goes through m_SensorData to check if the object with the suffix "_Stop" has any sensor in it, if yes, the method then creates a sensor on the conveyor in its respective position and finally assigns the method m_SensorMethod to the sensor.

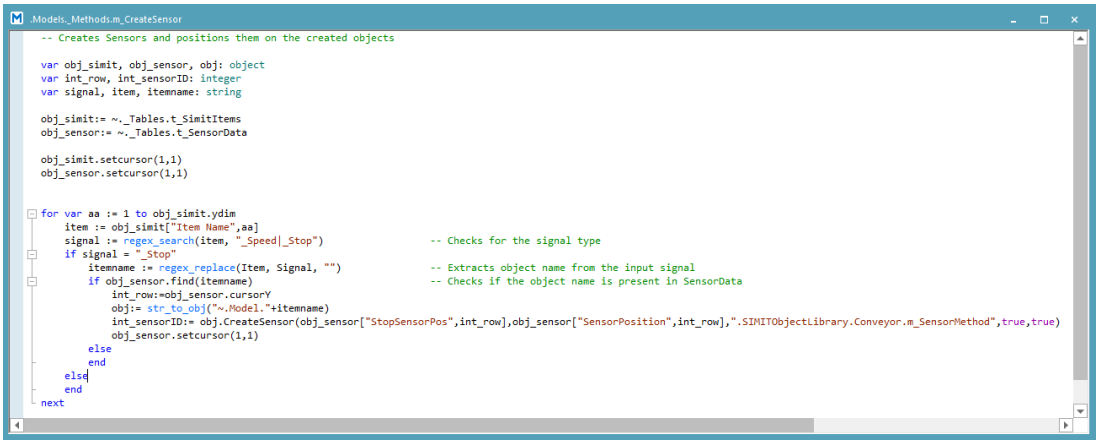


Figure 4.36: m_SensorMethod method.

ii. Object Connection:

To enable material flow between the created conveyors, a method was written, to check the objects in the model frame and compare it with the conveyor names from t_ResourceData, if two succeeding objects are both conveyors, the objects are connected using a connector. However, in the end, a separate line of code had to be written to connect the last conveyor in the line with the drain.

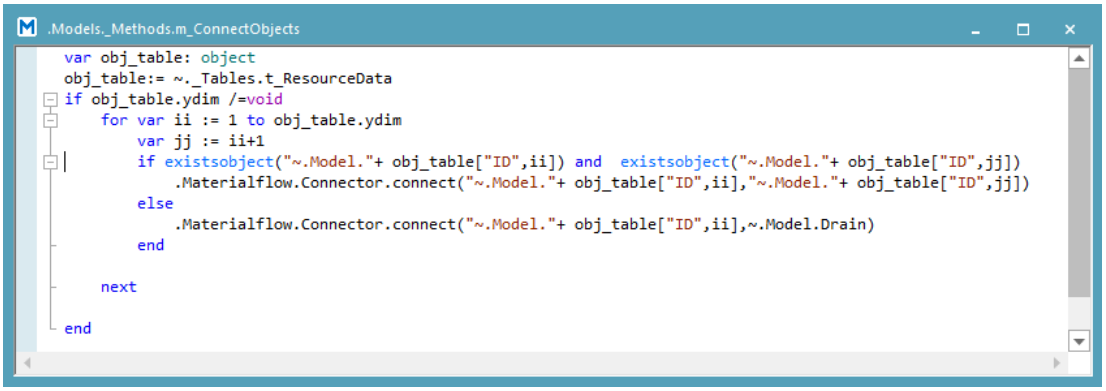


Figure 4.37: m_ConnectObjects method.

iii. Sensor Reset:

Finally, For the PLC program to function and emulate the conveyors properly, it is important to reset the conveyor sensors before the start of every emulation run. This was achieved by writing a code to set the value of the sensors to “False”. This method was also linked to a button named “Reset” in the model frame for ease of access.

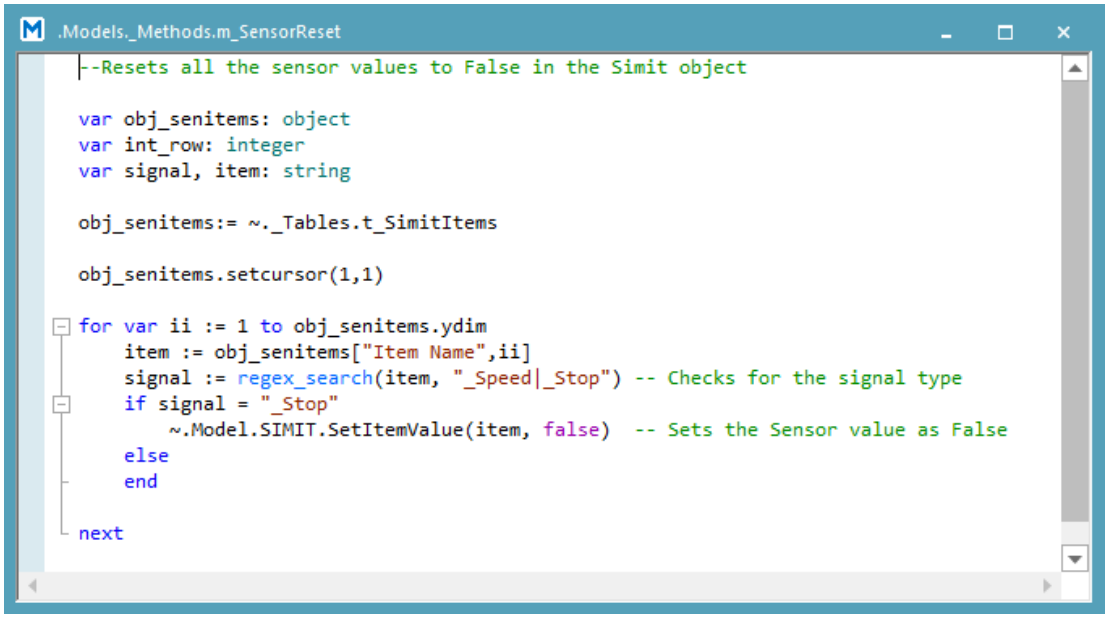


Figure 4.38: m_SensorReset method.

4.2.5 Integration and Emulation

With the PLC program, interface platform and virtual model all set up and ready to go online, it was time to integrate the tools and close the loop. To ensure unrestricted exchange of signals between TIA Portal and Plant Simulation, both tools are connected through the SIMIT model using “Couplings”. Coupling is a feature in SIMIT, that can be used to make a variety of connections to send and receive signals with other tools and software.

As mentioned in Chapter 4, section 4.2.3 (Interface Platform), the SIMIT model provided by Volvo already came with the PLC tags imported from the TIA Portal project, however, the coupling had to be deleted and created again, this will be explained in detail in Chapter 5, section 5.4 (Framework and Development). Once the PLC program is cleaned up, the TIA Portal project is saved under a different directory. Inside SIMIT a new PLCSIM Advanced coupling is created to import the TIA Portal project file. Although TIA Portal houses the PLC logic, PLCSIM Advanced is used to emulate the PLC logic to represent a virtual controller, thus, a PLCSIM Advanced coupling is

created. Once the coupling is created, “TIA Portal project” is selected and the corresponding file is opened, hence creating a new coupling between the interface platform and the PLC program.

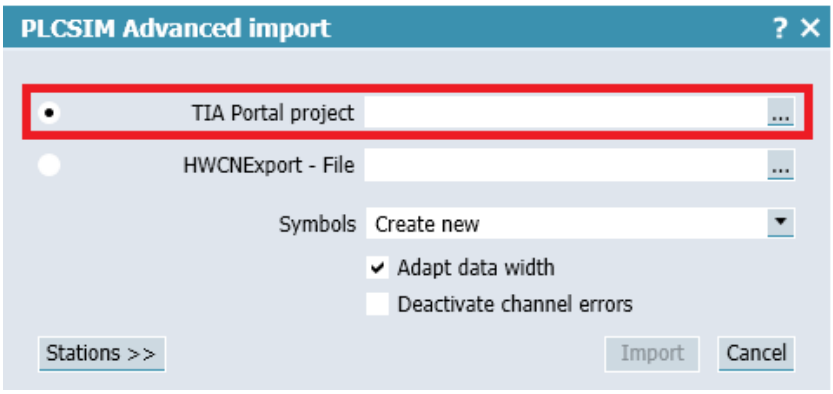


Figure 4.39: TIA Portal project files import window.

The coupling between the virtual model and the interface platform has already been defined in the form of Shared Memory in Chapter 4, section 4.2.3.2 (Shared Memory), therefore, the connections necessary for the signals to be exchanged are established.

Finally, to run the emulation, the PLC program must go online and to do that, a series of steps are followed sequentially. It is recommended that all tools and software are closed before beginning the process to ensure proper working of the technical setup.

i. Launch PLCSIM Advanced:

The first step is to launch PLCSIM Advanced so that the PLC program can be downloaded to emulate the virtual controller.

ii. Start SIMIT simulation:

SIMIT is launched and the simulation is started by clicking on the play button as shown in *Figure 4.40.* Starting the simulation allows Plant Simulation to connect to the Shared Memory.

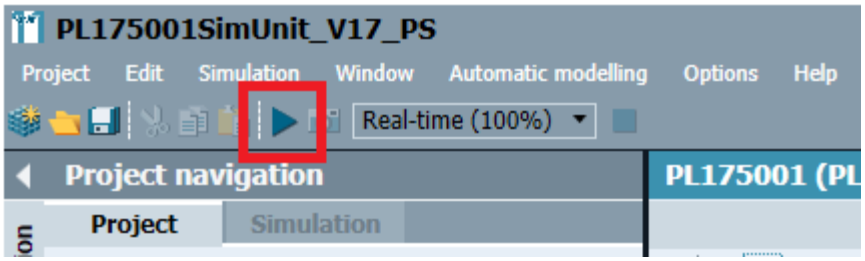


Figure 4.40: Start simulation button in SIMIT.

If the simulation is started successfully without any errors, the UI of SIMIT turns from blue to orange.

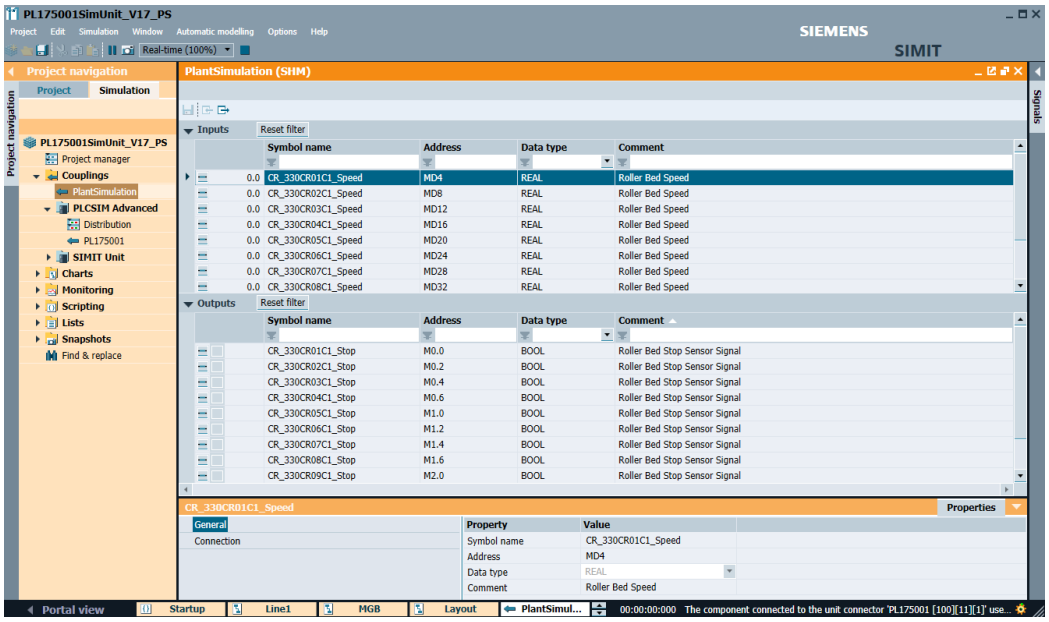


Figure 4.41: Indication that simulation is started.

iii. Connect to Plant Simulation to the Shared Memory:

Plant simulation is connected to the Shared Memory as instructed in Chapter 4, section 4.2.4.3 (Shared Memory) and the symbols are imported into the model.

iv. Download the PLC Program to PLCSIM Advanced:

TIA Portal is launched, and the project file is opened. For PLCSIM Advanced to emulate a virtual controller, the PLC program must be downloaded to it first. However, before downloading, the program is compiled to update any recent changes made in the program. Finally, the PLC program is downloaded to PLCSIM advanced by clicking the button as shown in *Figure 4.42*.

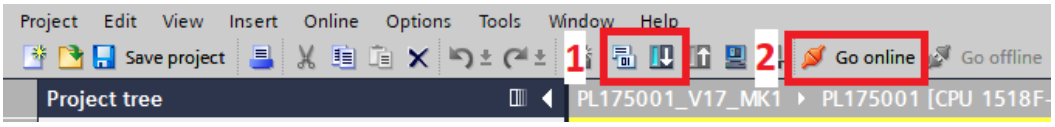


Figure 4.42: Compile and Download (1); Go Online (2).

v. Go online:

If the project gets downloaded to PCSIM Advanced without any errors, the PLC project is ready to go online. Once online the project is ready to accept input signals from the HMIs in SIMIT and start the emulation.

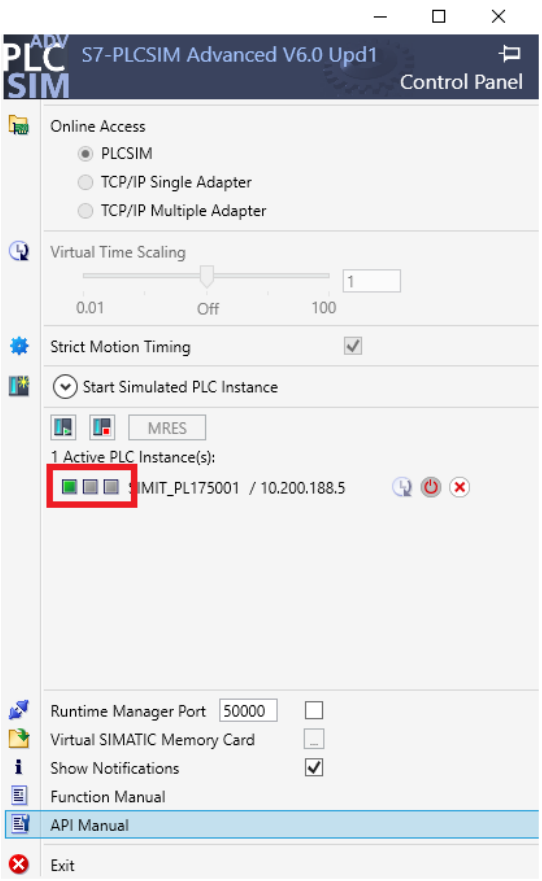


Figure 4.43: Simatic S7-PLCSIM Advanced V6.0.

The green light in PLCSIM Advanced indicates that the PLC program has been successfully downloaded to the virtual controller and is online.

vi. Set the line to auto in HMI:

After going online, the PLC program can receive input signals from the line and Multifunctional Gate Box (MGB) HMIs. Setting the line to auto allows the PLC logic to control the simulation objects in Plant Simulation automatically.

vii. Run the methods and start emulation:

Finally, the Plant Simulation window is restored, and all the necessary methods are run to extract data and create simulation objects. Clicking the start button commences the emulation process.

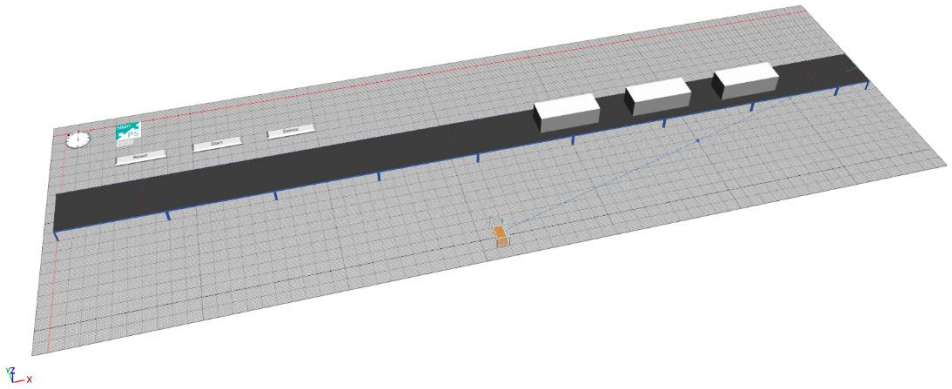


Figure 4.44: Final Plant Simulation Model after emulation.

Therefore, the software used in building the framework are integrated and the complete technical setup is established to be used for testing PLC logic using Discrete Event Simulation.

4.3 Testing Phase

The Integration between the tools must be tested to ensure the proper functioning of the established technical setup. For the PLC Logic to control the objects in the virtual model, the signals must be exchanged between the software and at the appropriate time, therefore, monitoring the signal values across the software can prove to be a viable method for testing the functionality of the setup.

In this case, the testing process is done across 3 tools, TIA Portal, SIMIT, and Plant Simulation.

Note: The screenshots in the following subsections do not all relate to the same instance of time of the emulation process, due to the process being continuous and real-time, it is difficult to capture the signal values across all tools at the same time.

4.3.1 TIA Portal

Once online and after commencing the emulation process in Plant Simulation, the PLC tags used in the PLC program can be monitored in TIA Portal to identify and study the changes in their values. Click on the “monitor all” button to bring up the real-time signal values.

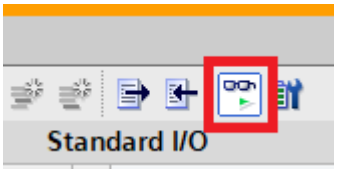


Figure 4.45: Button to monitor PLC tags.

If TIA Portal successfully receives the required HMI signals and stop sensor signals from SIMIT and Plant Simulation respectively, the PLC program processes the input and sends Control word values to SIMIT.

4.3.2 SIMIT

The Control word values sent by TIA portal can be monitored in the PLCSIM Advanced coupling inside SIMIT, however, to comprehend the speed value, the speed symbol values can be monitored in the input window of the Shared Memory coupling for any change in values.

PlantSimulation (SHM)

Inputs | Reset filter

	Symbol name	Address	Data type	Comment
0.0	CR_330CR01C1_Speed	MD4	REAL	Roller Bed Speed
0.0	CR_330CR02C1_Speed	MD8	REAL	Roller Bed Speed
0.0	CR_330CR03C1_Speed	MD12	REAL	Roller Bed Speed
0.0	CR_330CR04C1_Speed	MD16	REAL	Roller Bed Speed
0.0	CR_330CR05C1_Speed	MD20	REAL	Roller Bed Speed
0.0	CR_330CR06C1_Speed	MD24	REAL	Roller Bed Speed
80.0	CR_330CR07C1_Speed	MD28	REAL	Roller Bed Speed
0.0	CR_330CR08C1_Speed	MD32	REAL	Roller Bed Speed
0.0	CR_330CR09C1_Speed	MD36	REAL	Roller Bed Speed

Figure 4.46: Change in Speed value observed in Shared Memory coupling.

4.3.3 Plant Simulation

As the values change in symbols shared through the Shared Memory, the values should be reflected in Plant Simulation. The change in speed values can be monitored through the SIMIT object placed in the main model frame. The SIMIT object is opened, and the "Show Item-Values" button is clicked to monitor the speed values being changed in real-time.

.Models.Model.SIMIT - Items

CR_330CR01C1_Stop

string	string 1 Alias	string 2 Type	string 3 Value
4	CR_330CR04C1_Stop	BOOL	FALSE
5	CR_330CR05C1_Stop	BOOL	FALSE
6	CR_330CR06C1_Stop	BOOL	FALSE
7	CR_330CR07C1_Stop	BOOL	TRUE
8	CR_330CR08C1_Stop	BOOL	FALSE
9	CR_330CR09C1_Stop	BOOL	FALSE
10	CR_330CR01C1_Speed	REAL	0.000000
11	CR_330CR02C1_Speed	REAL	0.000000
12	CR_330CR03C1_Speed	REAL	0.000000
13	CR_330CR04C1_Speed	REAL	0.000000
14	CR_330CR05C1_Speed	REAL	80.000000
15	CR_330CR06C1_Speed	REAL	80.000000
16	CR_330CR07C1_Speed	REAL	80.000000
17	CR_330CR08C1_Speed	REAL	0.000000
18	CR_330CR09C1_Speed	REAL	0.000000
19			

Figure 4.47: Speed values of conveyors monitored through the SIMIT object.

The conveyor objects wait for Plant Simulation to receive the speed values through Shared Memory and distribute the values among the conveyors.

4.4 Verification and Validation

Once the interconnected setup passes the testing process, it is time to verify the functioning of the framework and how both PLC logic and the virtual model react to the I/O signals shared between them. The verification process is entirely done using the virtual model.

4.4.1 Verification Phase

Firstly, the emulated model is monitored to check if the roller beds are reacting to the inputs received through the Shared Memory, this form of verification however is purely visual. Once the emulation is started, conveyor beds must transport the MU to the drain placed at the end of the line respectively. In this case, the virtual model responded promptly to the inputs received by Plant Simulation, however due to a few breakpoints in the methods, the emulation process could not happen in real-time, thus causing a few

errors in the PLC program. Once the breakpoints were removed, the roller beds were able to transport the MU from the start of the line to the end without any issues.

Now, that the functioning of the virtual model as a whole, is verified, the working of the roller beds and their sensors are to be verified. The stop sensors in the roller bed should activate when the front of the MU contacts the sensor and similarly, the sensor should deactivate when the back of the MU passes the sensor. In other words, the sensor should set the stop sensor value to “TRUE” upon MU entry and “FALSE” upon exit. This is verified by running the emulation and comparing the sensor values side by side with material flow in the model frame. In this case, the sensors returned the appropriate values without any issues. The sensor values help the PLC program decide if the roller beds should transport and hand off the MU to the following roller bed, therefore it is important to verify the proper functioning to avoid collisions between the MUs in the virtual model.

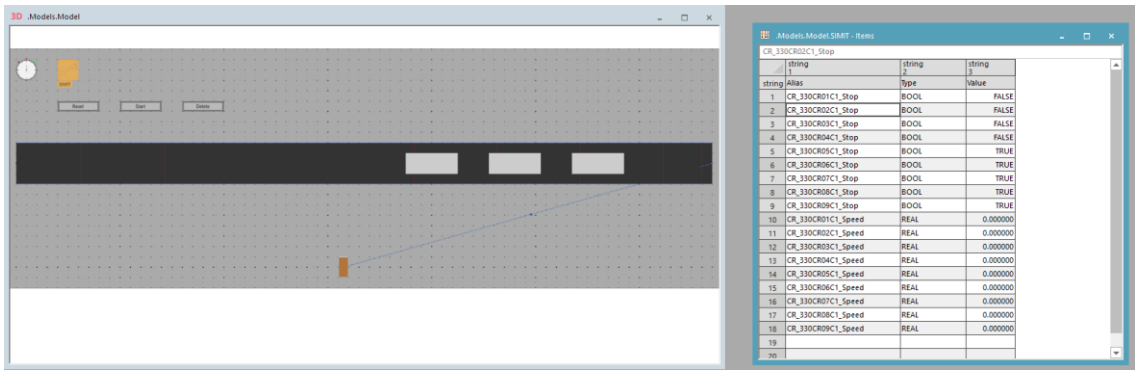


Figure 4.48: Model frame next to "Show Item-Values" table.

The conveyor sensors can also be verified using the SIMIT control chart, graphical components resembling sensors were created and placed right next to the corresponding roller beds. These graphical components are then connected to the output symbols from Plant Simulation and thus react to the sensor signal values from the virtual model.

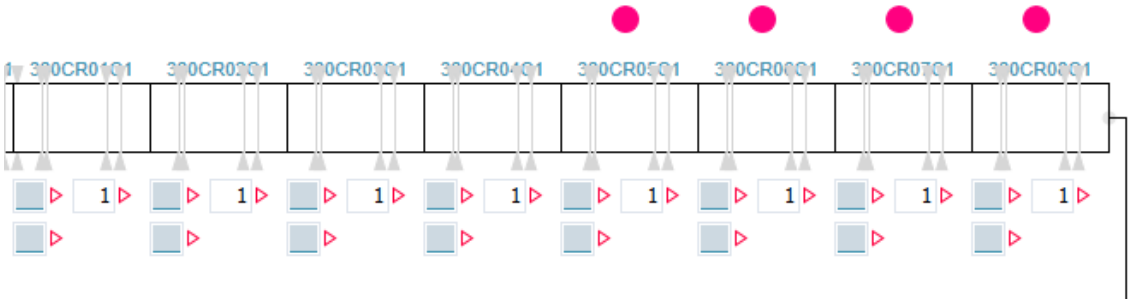


Figure 4.49: SIMIT control chart displaying Stop Sensors that are "TRUE".

4.4.2 Validation Phase

To validate the functioning of the framework, it is to be made certain that the PLC logic is visualized and reproduced in the virtual model. The validation process involves checking if the virtual model behaves the way the PLC logic wants it to.

Due to the lack of data from the actual line in the plant, the behaviour of the PLC logic could not be compared with any credible form of data, therefore the working of the PLC logic could not be completely validated. However, the queuing function of the control logic could be visualized in the emulated model and validated.

Note: The line emulated in this case is line 330 and it totally consists of 9 roller beds.

Multiple MUs were placed one after the other, the first MU was transported to the drain without any issues as no MUs were blocking the conveyor line, the second MU, however, was stopped on the 8th roller bed (330CR08M1) as the 9th roller bed (330CR09M1) did not receive any speed value. The 9th roller bed received no speed signal since, according to the PLC logic, the succeeding line, 340 has not received the first MU, as there is no roller bed from line 340 present in the virtual model.

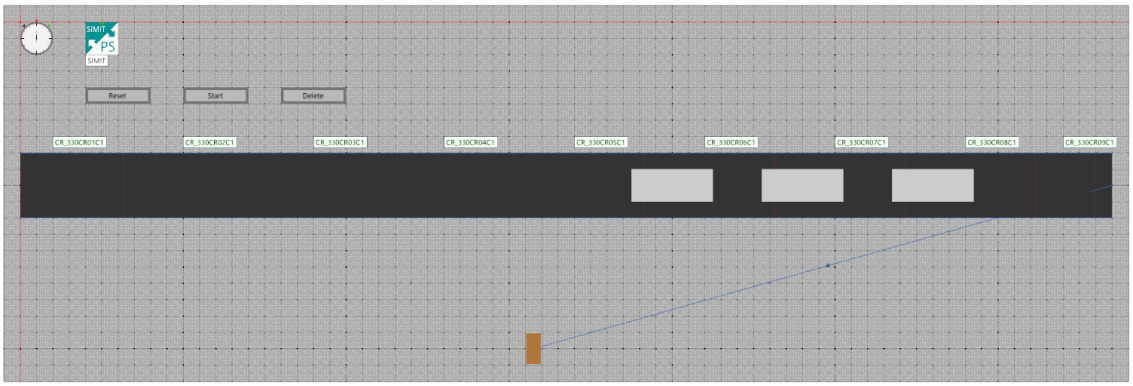


Figure 4.50: Queueing function observed in the virtual model.

Similarly, the third MU was stopped on the 7th roller bed, as the previous MU still had not left the 8th roller bed, and so on with the remaining MUs.

As shown in *Figure 4.50*, the queuing functionality of the line prevents the MUs from colliding with each other and the speed of the roller beds is set to zero until the succeeding roller beds become available to accept a new MU. Therefore, it can be inferred that the virtual model behaves in accordance with the PLC logic, thus validating the established setup.

5 DISCUSSION

This chapter will reflect on all the observations made during the development, testing and validation phases. This chapter also answers the research questions with regard to the development phase and literature study. RQ1 will focus on the overall framework of the technical setup. RQ2 will focus on the benefits resulting from establishing a proper technical setup versus simulation models. Finally, RQ3 will expand on how the established framework can help the design decision-making process and will also touch on the sustainability aspects of the project.

5.1 Research Question 1

How to establish a proper technical setup for virtual commissioning of PLC logic at a comprehensive level using Discrete Event Simulation?

The first research question revolves around the work done in the development phase and its framework. Virtual commissioning of PLCs is the process of simulating the logic used by the PLC to control a system in a virtual environment before it is implemented in the actual real-world system. To test a PLC program's functioning using DES, a framework must be built consisting of domains that serve a particular purpose and ultimately contribute to the virtual commissioning process. In this case, the framework consists of 4 domains, namely, PLC Logic, PLC Emulation, Interface Platform and virtual model.

The PLC Logic/ Program is the control logic that needs to be tested, the PLC Emulation domain deals with emulating a controller based on the PLC logic, acting as a virtual PLC. The Interface Platform on the other hand helps manage the I/O signals such that only the required signals are passed on to the virtual model. Finally, the virtual model accepts input signals from the Interface Platform and returns output signals. This continuous exchange of signals between the domains completes the loop and serves as the base framework for virtual commissioning PLCs.

A set of tools and software are required for each domain as mentioned, namely, TIA Portal which consists of the PLC Program, Simatic S7- PLCSIM Advanced for PLC Emulation, SIMIT for signal management and Shared Memory creation and finally Plant Simulation to build the virtual model. Each tool is readied to be integrated in a set sequence such that the signals can be exchanged between the PLC program and the virtual model. The PLC program is prepared such that the irrelevant safety signals and PN/PN couplers are managed. The Shared Memory coupling is created in SIMIT to create Input and Output symbols to be shared with the virtual model. The symbols created inside the Shared Memory coupling only exchange values that are relevant to the emulation mode. Inside Plant Simulation, specific frames are created to place emulation and non-simulation objects. To acquire symbols shared by SIMIT and to ensure the proper functioning of the emulation mode, auxiliary methods are written inside the frame of the methods.

Once the tools representing different domains are prepared for integration, a PLCSIM Advanced coupling is created inside SIMIT to import the TIA Portal project file. Since SIMIT is coupled to both Plant Simulation and TIA Portal, PLCSIM Advanced is launched in the background and simulation is started in SIMIT, in doing so, the Shared Memory goes online. Now, Plant Simulation can access the Shared Memory and the connection is established. Which enables the symbols to be imported into a data table in the virtual model. The PLC program can now be integrated into the loop through PLCSIM Advanced, the program is compiled and downloaded onto the virtual controller (PLCSIM Advanced) and upon successfully downloading the PLC Logic, TIA Portal can go online and control the virtual model.

However, it is also important to test and verify the connection by making sure that signals carry appropriate values across all tools as mentioned in Chapter 4, section 4.4.1 (Verification Phase). Therefore, this is how a technical setup can be established for testing PLC Logic using Discrete Event Simulation at a comprehensive level.

5.2 Research Question 2

What are the benefits of using PLC integrated virtual models over simulation models?

Although simulation models are considerably faster and easier to develop, virtual models have their own set of advantages that prove they are worthy enough as a concept to venture into. Simulating a model requires the control logic to be coded into the model, although, faster, it lacks the credibility a virtual model of the same functionality possesses. virtual models make use of external PLC logic to operate the objects they house; thus, the margin of error is relatively low. It is also possible that the simulation logic can sometimes be inaccurate with regard to the actual process design, as there is a factor of human error that can be caused due to misinterpretation of the process. virtual models help rectify all such

errors. Another major advantage of using virtual models would be replacing several case-specific simulation models to simulate different scenarios with different operating conditions, with a single virtual model.

DES allows for a very accurate graphical representation of the process, therefore using an external control logic in DES provides the combined advantage of precise visualization with a low margin of error. Another major advantage of using virtual models is that it helps verify the PLC logic before installing the PLCs in the actual line which in turn reduces the time taken to implement systems in a plant.

By integrating a virtual PLC, simulation objects can be controlled with any logic as they are not fixed to a certain logic, this increases the reusability and repeatability of the model. Any logic control can be used to emulate the objects as long as the logic is relevant to the virtual model in terms of the objects emulated, I/O signals and so on. The reusability of PLC-controlled virtual models is something that simulation models cannot achieve as they operate on inbuilt logic.

5.3 Research Question 3

How does the established setup help improve the design decision-making process for the stations in a conveyor line?

Virtual Commissioning makes it easier to run multiple emulations to ensure the PLC Logic is ready to be installed in the plant, in some cases this helps companies make important design decisions in station design. Testing pull sequences and IT system requests in a virtual environment is a significant advantage in improving the choice of technical solutions regarding material flow and speed in a virtual environment. If the concerned stakeholders are not happy with the performance or functions of a particular design, changes are made, which in turn results in changes in the PLC program or in some cases the logic is even completely swapped with another such that the expected results are achieved.

Virtual commissioning also aligns itself strongly with sustainability from the perspective of Triple Bottom Line.

i. Environmental Impact:

While trying to implement a change into an existing line in a plant, the line is stopped and the control logic to be implemented is tested using the physical line, this in turn increases the power consumption. Whereas in virtual commissioning, the PLC program is tested multiple times and perfected in a virtual environment before being implemented in the actual plant. Consequently, reducing the power consumption.

Moreover, as a result of running simulations using virtual models, the number of revisions per line is reduced significantly, therefore reducing the amount of resources used and discarded in the construction of the line in the factory.

ii. Social Impact:

Testing PLC logic physically in a plant requires workers to be present around heavy equipment and machineries. Using virtual models for testing purposes eliminates the need to be physically present in such harmful environments, thus contributing to the safety and wellbeing of the workers.

Having to redesign a line imparts high amounts of stress mentally and physically on the people involved in designing and constructing the line. The stress on the designers, engineers and constructors is reduced greatly by reducing the number of times a line has to be reworked.

iii. Economic Impact:

From an economic perspective, building a virtual model can be costlier than building a simulation model. Although having an economical advantage, simulation models are not as versatile as virtual models, thus requiring multiple case-specific models. Therefore, building one virtual model with high reusability can be more economical than building several case-specific simulation models.

5.4 Limitations in Development Phase

This thesis makes use of literature study and interviews with concerned stakeholders to define the framework of the technical setup. The software to be used to build the setup were provided by Volvo, therefore, sufficient documentation and literature on the set of software had to be researched. Due to the specificity and the lack of documentation on how to integrate some or even all of the software, a lot of the steps involved in the integration process were carried out on a trial-and-error basis.

Trying to build and integrate the technical setup came with its own set of issues and difficulties. One such issue was identified much later in the integration phase. After integrating the tools through couplings in SIMIT, and going online in TIA Portal, an MU was placed on a roller bed in the virtual model, and a dummy speed was given for test purposes. Although the MU came in contact with the stop sensor, the PLC logic did not provide the required speed value for the succeeding roller bed. Although changes in the sensor signal values were reflected in TIA Portal in real-time, there were no changes in the value of the output signals. After troubleshooting it was figured that the safety signals and PN/PN

couplers prevented the HMIs from setting the line on Auto operation mode. Therefore, the TIA Portal project had to be modified significantly. A considerable amount of time was spent on troubleshooting and modifying the PLC program; however, it would have been impossible to identify this issue before the integration phase.

As mentioned in Chapter 4, section 4.2.2 (PLC Program), certain PLC tags had to be deleted to tether any form of connection with the neighbouring PLCs. Due to this, the HMIs in SIMIT were not functioning as they should, this was caused by the safety alarm signals still being true, as some of the signals determining the values of the safety signals were missing. Therefore, a custom script was written inside SIMIT to help set the desired values to the required signals. Similarly in TIA Portal, a custom script was written to manage the transport alarms, especially the overtime alarms in conveyors and overtravel alarms in conveyor lifts.

The SIMIT project already had a PLCSIM Advanced coupling defined by Volvo, which consisted of an imported TIA Portal project file, as described in Chapter 4, section 4.2.3 (Interface Platform). In theory, the predefined coupling should have functioned as intended, however, only when integrating the various software, it was found that the simulation could not be started in SIMIT, as the imported file was corrupted. It was also noticed that there was a version mismatch between the imported project file and the TIA Portal software installed on the system. As a fix for this issue, the appropriate project file was identified by going through the local directory. A new PLCSIM Advanced coupling was created inside SIMIT and the V17 file was imported to solve the issue. Therefore, it is important to make sure the appropriate project file is used to create the PLCSIM Advanced coupling.

Initially, the idea was to emulate the complete line controlled by the virtual PLC. Unfortunately, due to complications caused by the PN/PN couplers and the safety signals required to operate the first and the last line, the idea was dropped, and a single line was chosen to be emulated. Although time-consuming, it would have been possible to manage the signals coming from the neighbouring lines (Upstream and Downstream) and control the complete line in the virtual model. However, due to the limitations set by the scope of the project and time availability, this could not be achieved.

The functioning of the technical setup was successfully verified, but not validated. Due to the lack of plant data, the results from the virtual model could not be compared to a credible source of data, hence validation of the PLC logic could not be carried out.

5.5 Observations and Takeaways

The emulation process is not an event-based simulation, instead takes place in real-time, the speed of the simulation cannot be changed inside Plant Simulation, and it is also mandatory that any breakpoints set inside either “m_Conveyor” or “m_SensorMethod” methods are

removed as they can disrupt the emulation process and can result in activating the overtime alarms in TIA Portal. However, in cases of such errors, the HMI corresponding to the line (330 in this case) can be compiled inside TIA Portal and then used to reset the transport memory of the particular roller bed to resolve the error.

When the start button is pressed in plant simulation, the material flow begins, and the simulation objects send sensor signals to the PLC program and the control logic processes the signals and sends speed signals back to the Plant Simulation model. However, there was a small lag noticed as the MU was handed off to the succeeding conveyor. In this case, the delay depends on the frequency with which the data is shared with the other software in the loop. The delay compounds over the various software resulting in a very noticeable lag in the virtual model. It was later learned that the lag between SIMIT and Plant Simulation can be reduced by decreasing the time slice value in the properties window of the Shared Memory coupling in SIMIT. Despite significant reductions in time slice size, the lag could only be reduced and not eliminated.

5.6 Future Scope

Due to certain limitations, various aspects of this project were not explored properly, such activities and research work relating to them are discussed in this section.

5.6.1 Case Study

The most significant limitation faced in the development phase was, not being able to build and control the complete conveyor system using the PLC logic. However, to emulate the complete system, the PN/PN coupler signals must be managed in a way that they don’t hinder the functioning of the first and last line in the system. Secondly, the generation of components other than single roller beds can be explored inside Plant Simulation, especially, if the PN/PN coupler signals are fixed.

5.6.2 Application

The virtual model’s integration with Product Lifecycle Management tools can be researched since manually acquiring data from the SIMIT Contec Library components can be a physically demanding task for longer lines, therefore, conveyor data can be automatically obtained through a database.

During this project, since all the required tools were located on a single computer, the number of people that could work on the project at a time was limited to one. However, integrating SIMIT to an OPC UA server coupling will enable different departments to work on different domains of the framework simultaneously.

As an Interface Platform tool, the functionalities of SIMIT are extensive, and its functions could be further explored. In addition to Plant Simulation, SIMIT could also integrate with additional visualization tools through its ability to create multiple Shared Memory couplings. For example, by establishing a connection with Emulate3D, the setup could harness the capabilities of Virtual Reality which would in turn have a massive societal impact, as it can enable training workers in a remote and safe environment. Additionally, research on connecting Plant Simulation to the actual plant with SIMIT may lead to the creation of Digital Shadows or even Digital Twins, thus possibly enabling other valuable functionalities such as predictive maintenance.

6 CONCLUSION

The aim of the thesis, to establish a technical setup to test PLC Logic using Discrete Event Simulation has been achieved. The framework presented in Chapter 4 section 4.2 (Development Phase) has been deployed to build a virtual model housing simulation objects that can be controlled by an external PLC logic. The purpose of the setup is to test PLC programs using virtual models built within a Discrete Event Simulation software. The integration process helps understand the requirements of each software and how they communicate with each other to exchange signals through the loop. Ultimately, this thesis aids in comprehending the advantages and versatility of using virtual models in virtual commissioning, in comparison to using simulation models. Although, developing a comprehensive technical setup for virtual commissioning can be complex in nature and time consuming, it can open new opportunities and pave way to the development of technologies such as digital twins and so on.

REFERENCES

Amos Ojo Arowoshegbe, E. U. (2018). *SUSTAINABILITY AND TRIPLE BOTTOM LINE: AN OVERVIEW OF TWO INTERRELATED CONCEPTS*. Retrieved from https://www.researchgate.net/publication/322367106_SUSTAINABILITY_AND_TRIPLE_BOTTOM_LINE_AN_OVERVIEW_OF_TWO_INTERRELATED_CONCEPTS

Andrade, A. (2023). *The 3 levels of digital twin technology*. Retrieved from https://vidyatec.com/blog/the-3-levels-of-the-digital-twin-technology-2/

Banks, J. (1999). *Introduction to simulation*. Retrieved from https://doi.org/10.1145/324138.324142

Bryman, A. a. (2011). *Business research methods. 3rd ed*.

Chi G. Lee, S. C. (2014). *Survey on the virtual commissioning of manufacturing systems*. Retrieved from https://doi.org/10.7315/JCDE.2014.021

Chi G. Lee, S. C. (2014). *Survey on the virtual commissioning of manufacturing systems,*. Retrieved from https://doi.org/10.7315/JCDE.2014.021.

Erik Gerdin, R. R. (2018). *Manufacturing System Improvement with Discrete Event Simulation : A case study with simulation applied to a manual manufacturing system*. Retrieved from http://uu.diva-portal.org/smash/get/diva2:1229062/FULLTEXT01.pdf

Gilchrist, A. (2016). *Industry 4.0-The Industrial Internet of Things*. Retrieved from https://link.springer.com/book/10.1007/978-1-4842-2047-4

Hinchey, H. S. (2009). *Model-Based Verification of Embedded Software*. Retrieved from 10.1109/MC.2009.125

Ingalls, R. G. (2011). *Introduction to simulation*. Retrieved from doi: 10.1109/WSC.2011.6147858

J. A. B. Montevechi, T. F. (2015). *Identification of the main methods used in simulation projects," 2015 Winter Simulation Conference (WSC)*. Retrieved from doi: 10.1109/WSC.2015.7408507

José Alberto Valdez Becerril, H. K. (2021). *Migration of 3D Simulation software in.* Gothenburg: CHALMERS UNIVERSITY OF TECHNOLOGY.

Lasi, H. F. (2014). *Industry 4.0*. Retrieved from https://doi.org/10.1007/s12599-014-0334-4

Mansharamani, R. (1997). *An overview of discrete event simulation methodologies and implementation*. Retrieved from https://doi.org/10.1007/BF02802549

McGregor, I. (2002). *Equipment interface: the relationship between simulation and emulation*. Retrieved from https://www.researchgate.net/publication/221529337_Equipment_interface_the_relationship_between_simulation_and_emulation

MELOENY, S. (2022). *Calsoft's Industry 4.0 Framework for Manufacturing*. Retrieved from https://www.calsoft.com/industry-4-0-calsoft-framework/#:~:text=Industry%204.0%20uses%20advanced%20technologies,competitive%20in%20the%20global%20market.

Moris, J. O. (2002). *Documentation of discrete event simulation models for manufacturing system life cycle simulation*. Retrieved from doi: 10.1109/WSC.2002.1166359

Morse, J. M. (2002). *Verification Strategies for Establishing Reliability and Validity in Qualitative Research.* Retrieved from https://doi.org/10.1177/160940690200100202

Muthanna Jamil, N. R. (2016). *Simulation of Assembly Line Balancing in Automotive Component Manufacturing*. Retrieved from DOI:10.1088/1757-899X/114/1/012049

Rodič, B. (2017). *Industry 4.0 and the New Simulation Modelling Paradigm*. Retrieved from DOI:10.1515/orga-2017-0017

Runa Patel, B. D. (2019). *Forskningsmetodikens grunder 2nd ed* .

Sargent, R. G. (2010). *Verification and validation of simulation models," Proceedings of the 2010 Winter Simulation*. Retrieved from doi: 10.1109/WSC.2010.5679166.

Schiess, C. (2001). *Emulation: debug it in the lab --- not on the floor*. Retrieved from https://dl.acm.org/doi/10.5555/564124.564338

Schlesinger, S. (1979). *Terminology for model credibility. SIMULATION. 1979;32(3):103-104.* Retrieved from doi:10.1177/003754977903200304

Shannon, R. (1998). *Introduction to the art and science of simulation*. Retrieved from 10.1109/WSC.1998.744892

Sharma, P. (2015). *Discrete-Event Simulation*.

Siemens AG. (2020). *SIMIT Simulation Platform (V10.2) Operating Manual.* NÜRNBERG: Siemens AG.

Sirisilla, S. (2023). *Inductive and Deductive Reasoning — Strategic approach for conducting research*. Retrieved from https://www.enago.com/academy/inductive-and-deductive-reasoning/

Snyder, H. (2019). *Literature review as a research methodology: An overview and guidelines*. Retrieved from https://doi.org/10.1016/j.jbusres.2019.07.039

STĂNCIOIU, A. (2017). *THE FOURTH INDUSTRIAL REVOLUTION „INDUSTRY 4.0”*.

Tao, F. C. (2018). *Digital twin-driven product design, manufacturing and service with big data*. Retrieved from https://doi.org/10.1007/s00170-017-0233-1

Viktor Engström, Z. L. (2017). *PLC Integrated Discrete Event Simulation for Production Systems.* Gothenburg: CHALMERS UNIVERSITY OF TECHNOLOGY.

Werner Kritzinger, M. K. (2018). *Digital Twin in manufacturing: A categorical literature review and classification*. Retrieved from https://doi.org/10.1016/j.ifacol.2018.08.474

William de Paula Ferreira, F. A.-E. (2020). *Simulation in industry 4.0: A state-of-the-art review*. Retrieved from https://doi.org/10.1016/j.cie.2020.106868

Wladimir Hofmann, S. L. (2017). *Integrating Virtual Commissioning Based on High Level Emulation into Logistics Education*. Retrieved from https://doi.org/10.1016/j.proeng.2017.01.055

Wünsch, G. R. (2007). *Economic application of virtual commissioning to mechatronic production systems*. Retrieved from https://doi.org/10.1007/s11740-007-0066-0

Yingling, R. B. (2016). *Quantifying benefits of conversion to lean manufacturing with discrete event simulation: A case study*. Retrieved from https://doi.org/10.1080/002075400189509

Ülgen, O. M. (2001). *SIMULATION METHODOLOGY, TOOLS, AND APPLICATIONS.” Chap. 90 in Maynard’s Industrial Engineering Handbook*. Retrieved from https://www.accessengineeringlibrary.com/content/book/9780070411029/chapter/chapter90

A APPENDIX

A.1 Hasan's Model for Hybrid Simulation and Emulation Model

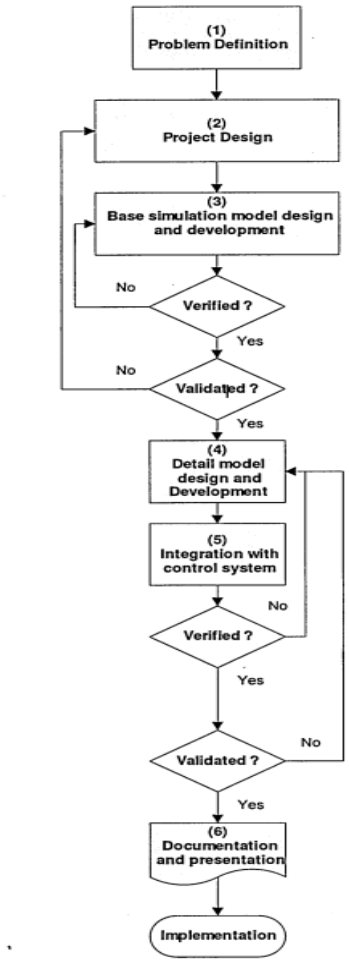


Figure A.1: Hasan's HSEM model

A.2 Conveyor Data Excel Sheet

	A	B	C	D	E	F	G
1	ID	Type	PosX	PosY	Direction	Length	Width
2	CR_330CR01C1	Conveyor		0	200	0	8	4
3	CR_330CR02C1	Conveyor		160	200	0	8	4
4	CR_330CR03C1	Conveyor		320	200	0	8	4
5	CR_330CR04C1	Conveyor		480	200	0	8	4
6	CR_330CR05C1	Conveyor		640	200	0	8	4
7	CR_330CR06C1	Conveyor		800	200	0	8	4
8	CR_330CR07C1	Conveyor		960	200	0	8	4
9	CR_330CR08C1	Conveyor		1120	200	0	8	4
10	CR_330CR09C1	Conveyor		1280	200	0	3	4
11								

Table A.1: Excel sheet consisting of conveyor data.

A.3 t_ResourceData Data Table from Plant Simulation

	A	B	C	D
1	**ID**	**SenNumber**	**StopSensorPos**	**SensorPosition**
2	CR_330CR01C1	1	6,3	Length
3	CR_330CR02C1	1	6,3	Length
4	CR_330CR03C1	1	6,3	Length
5	CR_330CR04C1	1	6,3	Length
6	CR_330CR05C1	1	6,3	Length
7	CR_330CR06C1	1	6,3	Length
8	CR_330CR07C1	1	6,3	Length
9	CR_330CR08C1	1	6,3	Length
10	CR_330CR09C1	1	2	Length

Table A.2: Excel sheet consisting of sensor data.

A.4 t_ResourceData Table from Plant Simulation

	string 1	string 2	real 3	real 4	real 5	real 6	real 7
string	ID	Type	PosX	PosY	Direction	Length	Width
1	CR_330CR01C1	Conveyor	0.00	200.00	0.00	8.00	4.00
2	CR_330CR02C1	Conveyor	160.00	200.00	0.00	8.00	4.00
3	CR_330CR03C1	Conveyor	320.00	200.00	0.00	8.00	4.00
4	CR_330CR04C1	Conveyor	480.00	200.00	0.00	8.00	4.00
5	CR_330CR05C1	Conveyor	640.00	200.00	0.00	8.00	4.00
6	CR_330CR06C1	Conveyor	800.00	200.00	0.00	8.00	4.00
7	CR_330CR07C1	Conveyor	960.00	200.00	0.00	8.00	4.00
8	CR_330CR08C1	Conveyor	1120.00	200.00	0.00	8.00	4.00
9	CR_330CR09C1	Conveyor	1280.00	200.00	0.00	3.00	4.00
10							

Table A.3: t_ResourceData data table consisting of imported conveyor data.

A.5 t_SensorData Table from Plant Simulation

	string 1	integer 2	real 3	string 4
string	ID	SenNumber	StopSensorPos	SensorPosition
1	CR_330CR01C1	1	6.30	Length
2	CR_330CR02C1	1	6.30	Length
3	CR_330CR03C1	1	6.30	Length
4	CR_330CR04C1	1	6.30	Length
5	CR_330CR05C1	1	6.30	Length
6	CR_330CR06C1	1	6.30	Length
7	CR_330CR07C1	1	6.30	Length
8	CR_330CR08C1	1	6.30	Length
9	CR_330CR09C1	1	2.00	Length
10				

Table A.4: t_SensorData data table consisting of imported sensor data.

DEPARTMENT OF INDUSTRIAL AND MATERIALS SCIENCE
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY