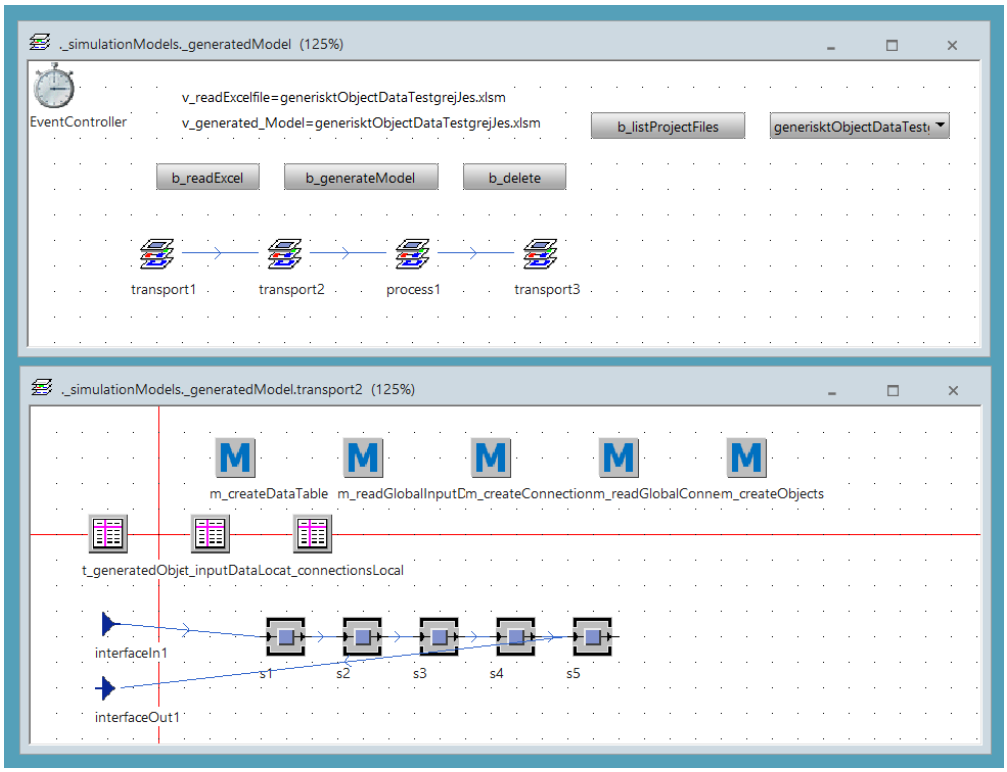
CHALMERS
UNIVERSITY OF TECHNOLOGY



Development of Generic Simulation Objects in Discrete Event Simulation

Enabling Automatic Generation of Simulation Models using Process-Related Data from a PLM system

Master's thesis in Production Engineering

MAX JOHANSSON
JESPER SÖDERSTRÖM

DEPARTMENT OF INDUSTRIAL AND MATERIALS SCIENCE
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Master's thesis 2025

Development of Generic Simulation Objects in Discrete Event Simulation

Enabling Automatic Generation of Simulation Models using Process-Related Data from a PLM System

MAX JOHANSSON
JESPER SÖDERSTRÖM

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Industrial and Materials Science
Division of Production Systems
Chalmers University of Technology
Gothenburg, Sweden 2025

Development of Generic Simulation Objects in Discrete Event Simulation
Enabling Automatic Generation of Simulation Models using Process-Related Data from a PLM System
MAX JOHANSSON
JESPER SÖDERSTRÖM

Industrial Supervisors: Anton Albo and Dennis Andersson, Volvo Cars Corporation
Academic Supervisor: Ebru Turanoglu Bekar, Department of Industrial and Materials Science
Examiner: Anders Skoogh, Department of Industrial and Materials Science

Master's Thesis 2025
Department of Industrial and Materials Science
Division of Production Systems
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Screenshot of a generated simulation model using the generic simulation tool inside Tecnomatix Plant Simulation.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2025

Development of Generic Simulation Objects in Discrete Event Simulation
Enabling Automatic Generation of Simulation Models using Process-Related Data from a PLM System
MAX JOHANSSON, JESPER SÖDERSTRÖM
Department of Industrial and Materials Science
Chalmers University of Technology

Abstract

The automotive and manufacturing industries are undergoing a transformation with the adoption of advanced simulation and data management systems. The practice of manually modeling simulation models is time-consuming and costly, which limits the feasibility of simulation studies when resources are insufficient.

This master thesis thereby investigates how process-related data from a PLM system can be utilized to automatically generate generic simulation objects that accurately represent manufacturing systems. It further explores the possibilities of integration between PLM and DES applications, presenting several concepts for automatic generation and PLM/DES integration.

The findings from the master thesis shows that automatic generation of generic simulation objects is achievable and effective in reducing manual efforts. Additionally, centralizing data in a single location improves input data management and enhances compatibility with different DES applications. To demonstrate this, a generic simulation tool and user-interface were developed. The solution provides improved interoperability opportunities between PLM and DES environments, reducing manual work and increasing efficiency for simulation engineers.

Keywords: Automatic Generation, Discrete Event Simulation, Generic Simulation Objects, Input Data Management, Interoperability, Product Lifecycle Management.

Acknowledgements

We are deeply grateful for the opportunity to conduct our master's thesis at Volvo Cars Corporation on such an interesting topic.

We would like to extend our sincere thanks to our industrial supervisors, Dennis Andersson and Anton Albo, for the support along the project. We are also very thankful to our academic supervisor Ebru Turanoglu Bekar, for her highly engagement and support along the thesis writing process. In addition, we would like to express our appreciation to our examiner Anders Skoogh for his valuable inputs and for being part of this master thesis.

We would also like to express our gratitude to all the simulation engineers at Volvo Cars Corporation who participated in the interviews. Your insights and shared experiences enhanced our understanding of the topics explored.

Max Johansson
Jesper Söderström
Gothenburg, June 2025

List of Acronyms

Below is a list of acronyms that have been used throughout the thesis report, listed alphabetically:

AML	Automation Markup Language
BOE	Bill of Equipment
BOP	Bill of Process
CAD	Computer Aided Design
CMSD	Core Manufacturing Simulation Data
CT	Cycle Time
DES	Descrete Event Simulation
DSRM	Design Science Research Methodology
DT	Delay Time
ERP	Enterprise Resource Planning
GDM	Generic Data Management
ISA	International Society of Automation
JPH	Jobs Per Hour
KPI	Key Performance Indicators
MDT	Mean Down Time
MTTR	Mean Time to Repair
MU	Mobile Unit
PLC	Programmable Logic Controller
PLM	Product Lifecycle Management
SMEs	Small or Medium Enterprises
SSIT	Standard Stock In Transfer
SSIP	Standard Stock In Process
SQL	Structured Query Language
TCT	Target Cycle Time
TT	Throughput Time
UML	Unified Modeling Language
VCC	Volvo Cars Corporation
XML	Extensible Markup Language

Contents

List of Acronyms **ix**

List of Figures **xiii**

List of Tables **xv**

1 Introduction **1**
1.1 Background . . . 1
1.2 Purpose . . . 2
1.3 Aim . . . 2
1.4 Research Questions . . . 3
1.5 Delimitations . . . 3

2 Theoretical Background **5**
2.1 Discret Event Simulation . . . 5
2.1.1 File Formats and Standards in DES . . . 6
2.1.2 DES Application - Siemens Tecnomatix Plant Simulation . . . 6
2.1.2.1 Standard Simulation Objects . . . 7
2.2 Product Lifecycle Management . . . 8
2.2.1 PLM Software - Teamcenter . . . 8
2.3 Transport Systems . . . 9

3 Methodology **11**
3.1 Design Science Research Methodology . . . 11
3.2 Literature Study . . . 12
3.3 Qualitative Study . . . 13
3.3.1 Stakeholder Analysis . . . 13
3.3.2 Semi-structured Interviews . . . 14
3.3.3 Unstructured Interviews . . . 15
3.4 Development of Generic Objects . . . 15
3.5 Verification of Generic Simulation Tool . . . 16
3.6 User-interface . . . 17
3.7 Standard Work Procedure . . . 18

4 Results **19**
4.1 Literature findings . . . 19
4.1.1 Input Data Management Methods . . . 19

4.1.2 Generic Data Structure and Automatic Generation in DES . . 20
4.1.2.1 Generic Data Structure 20
4.1.2.2 Intermediate Database 21
4.1.2.3 Applications for Data Input Management and Conversion 23
4.1.2.4 Direct integration 24
4.1.2.5 Alternative Approach: Data Derived From PLC . . . 26
4.1.3 Potential features 26
4.2 Results from Qualitative Study 27
4.2.1 Current State of DES 27
4.2.2 Input Data Parameters 28
4.2.3 Input Data Management 29
4.2.4 Design Specifications 31
4.3 Results from Development of Generic Simulation Objects 32
4.3.1 Description of the Utilized Generic Data Structure 32
4.3.2 Structure of the Built Generic Simulation Tool 33
4.3.3 Generation Methods 34
4.3.3.1 Global Methods 35
4.3.3.2 Local Methods 36
4.3.4 Tables 37
4.3.5 Drop-down list 38
4.3.6 Buttons 38
4.4 Developed User-interface 38
4.5 Standard Work Procedure 40
4.5.1 Standard Work Procedure for user-interface 40
4.5.2 Standard Work Procedure for Generic Simulation Tool 41

5 Discussion **43**
5.1 Research Questions 43
5.2 Contributions and Limitations of the Thesis 47
5.3 Future Work and Research 49
5.4 Ethical, Societal and Ecological Aspects of- the Thesis 50

6 Conclusion **51**

Bibliography **53**

A Appendix 1 - Consent Form **I**

B Appendix 2 - Interview Questions **III**

C Appendix 3 - Generic Simulation Tool **V**

D Appendix 4 - Input data table **VII**

E Appendix 5 - User-Interface Worksheets **IX**

List of Figures

3.1 DSRM process 12

4.1 Intermediate database solution (adopted by [5]) 22
4.2 Data input management tool (adopted by [4]) 23
4.3 Methodology for direct integration (adopted by [47]) 25
4.4 Data gathering supported by PLC and MES (adopted by [49]) 26
4.5 The designed generic data structure 33
4.6 The frame "_generateModel" with an example project generated . . . 34
4.7 Example of Generated Generic Simulation Object 35
4.8 User-interface: buttons 39

C.1 Generic Simulation Tool VI

E.1 User-Interface objectTemplate IX
E.2 User-Interface objects X
E.3 User-Interface connections XI
E.4 User-Interface objectMethods XII
E.5 User-Interface translation XIII

List of Tables

3.1 Stakeholders 14

4.1 Design specifications, needs 31
4.2 Design specifications, wishes 32
4.3 Example of a connection table 37
4.4 Example of translation table 38
4.5 ObjectTypes 39

D.1 Example of a main input data table VII

1
Introduction

This chapter presents the background of this master thesis at Volvo Cars Corporation (VCC), highlighting the growing complexity of manufacturing systems and the interoperability between advanced technologies (e.g. simulation tools) and data management systems. Additionally, the purpose, aim, and objectives are defined, along with the scope and delimitation of the thesis study.

1.1 Background

The automotive and manufacturing industries are experiencing a transformation through the integration of advanced simulation technologies and data management systems. Product Lifecycle Management (PLM) platforms play a critical role in managing the lifecycle of products and manufacturing systems by sorting, sharing, and updating data across multiple engineering disciplines [1].

At the same time, Discrete Event Simulation (DES) and Virtual Commissioning tools are becoming essential to optimize production processes before physical implementation [2]. These technologies enable engineers to simulate, test, and refine control strategies in a virtual environment, significantly reducing the risks associated with design flaws and inefficiencies [2].

Today, manufacturing systems, processes and information flow are becoming increasingly complex. For this, DES is a powerful tool for handling such complexity and for testing or developing new system configurations [3]. However, despite the potential, there is still room for improvement, particularly regarding input data management [4]. One of the most time-consuming phases of DES is acquiring input data, as users still rely heavily on manual work. Some of the challenges include the need to collect data from multiple sources, data availability and the lack of well-structured databases especially in small or medium enterprises (SMEs) [4],[5].

For data exchange in DES, several data formats and industrial standards are available, such as Core Manufacturing Simulation Data (CMSD), Automation Markup Language (AML) and International Society of Automation (ISA-95) [6]. Despite their existence, none of these formats currently comprehensively cover all key parameters necessary for DES. As alternative data exchange solutions, intermediary databases and complete manual data input, can also be used [5], [7].

In previous thesis studies at VCC, the challenges of integrating real-world data into Teamcenter and the interoperability between PLM and DES applications were investigated [8]. Additionally, the project focused on enhancing the data management of Teamcenter for DES. A few potential functions were proposed, and a work methodology for extracting data was developed. This thesis study will continue, with further research on improvements of interoperability between PLM and DES applications, and investigate how this can be utilized during the simulation modeling.

1.2 Purpose

DES is used for evaluation and design of new manufacturing processes as well as for performance improvement of preexisting manufacturing processes [3]. There are in general restricting factors to simulation projects such as time and expertise, which results in a considerable financial cost [5]. It is therefore not considered appropriate nor recommended to perform a simulation study if there are insufficient resources or time [9]. Simulation projects could for these reasons be considered unaffordable even if they are required, especially for SMEs. [3],[5].

The current method at VCC for modeling simulation models involves creating new simulation objects for each case. This approach is not only time-consuming, but also requires extensive customization. Instead, a more general and efficient method would be desirable. PLM systems have the potential to store process-related data, which is essential to modeling accurate DES models [1]. However, the integration between PLM and simulation applications remains quite limited today. Thus, the purpose of this thesis is to explore how data from a PLM system could be utilized to automatically generate reusable and generic simulation objects for a manufacturing system. This approach contributes to the field by enhancing efficiency through simplification and acceleration of modeling processes in simulation projects.

1.3 Aim

The aim of this thesis study is to develop reusable generic simulation objects used to automatically generate simulation models of manufacturing systems. These objects will utilize process-related data that could potentially be stored in a PLM system such as Teamcenter. Additionally, the study aims to design a user-interface that facilitates the configuration of transport systems and enables the automatic generation of transport objects based on these configurations. Thereby, this thesis study is divided into to the following objectives:

1. Creating generic simulation objects capable of representing various manufacturing systems, guided by process-related data.

2. Develop a function for automatic generation of the generic simulation objects.

3. Establishing a structured work procedure for integrating process-related data

from e.g. a PLM system into the simulation objects to ensure accurate system behavior in DES.

4. Designing an user-interface that enables users to define transport systems and buffer areas, facilitating the automatic generation of transport objects with accurate behavior and performance metrics.

5. Verifying that the simulation model have been generated with the correct process-related data.

1.4 Research Questions

Based on the purpose and aim, the following research questions will be addressed:

RQ1: How can a generic simulation object be designed so that it can accurately represent multiple manufacturing processes and transport systems, and enable automatic generation of simulation models?

RQ2: How can process-related data from a PLM system be effectively structured and integrated into DES applications to ensure accurate system behavior?

RQ3: How can a user-interface be designed to enable the configuration of transport systems for a DES environment?

1.5 Delimitations

The project will limit the scope to focus on a few manufacturing systems, particularly transport systems. Additionally, the integration between PLM and the simulation tool will be limited to extracting process-related data from an Excel file for creating generic simulation objects, rather than a comprehensive system-level integration. The simulation software used for this thesis is Siemens Technomatix Plant Simulation, and the software used for the user-interface is Microsoft Excel. Furthermore, the user-interface will be developed at a prototype or proof-of-concept level.

2
Theoretical Background

The purpose of this chapter is to provide a comprehensive overview of relevant information and key concepts to establish a solid foundation for understanding this thesis study. The first section begins with an exploration of DES, detailing its application within the industry. It covers various file formats and standards in DES, and introduces the specific simulation application utilized in this thesis study. Following this, the concept of PLM is explored, including an introduction of the specific PLM software considered in this thesis. Lastly, concepts concerning transport system are described.

2.1 Discret Event Simulation

As defined by Banks a simulation is "the imitation of the operation of a real-world process or system over time", where DES serves as an effective tool for modeling, optimizing, and analyzing real-world manufacturing systems [3], [9]. It is a powerful tool that enables the evaluation of different system configurations and the testing of "what if" scenarios without disrupting ongoing manufacturing systems. Furthermore, it facilitates bottleneck analysis, thereby enhancing productivity and due to its flexibility, DES has become a widely adopted tool across various industries. Initially, it was primarily used in the industrial sector, however over time, its application has expanded to other sectors, including logistics, construction, military, and healthcare [10].

In general, a DES model contains of five essential components: *entities, attributes queues, activities* and *resources*, which can be described as follows [10]:

- *Entities*: Items that flow through a system, such as product components in a manufacturing process, data in an information system, or patients at a hospital.

- *Attributes*: Specific characteristics that define an entity, such as product type, required activity time, or queue priority. These attributes are essential for the control logic of the model, where different conditions decide the model behavior.

- *Queues*: Locations where entities wait before proceeding to an activity, such as warehouses, buffers, or hospital waiting rooms.

- *Activities*: Processes performed on entities, including manufacturing operations, transportation, or service provision.

- *Resources*: Essential elements required to execute an activity, such as machine operators, vehicles, or healthcare professionals.

2.1.1 File Formats and Standards in DES

For interoperability in DES, various file formats and industry standards can be utilized. The more commonly used are AML, ISA-95, and CMSD [6]. AML is a Extensible Markup Language (XML) based, neutral data format designed to enable data exchange of process-related data within a manufacturing setting [11]. This standards aims to facilitate interoperability among different applications utilized in production environments such as planning tools, PLC programs etc. In general AML include data regarding: factory topology, geometry, kinematics and control logic [12].

ISA-95 is a standard intended to serve as a framework for data exchange between a company's systems [13]. This standard defines a structured hierarchy encompassing different levels of an enterprise, describing the flow of information from physical production all the way up to highest management level. By applying this standard, risks associated with integration of systems can be reduced. While ISA-95 is a standard rather than a concrete file format, its concepts can be effectively presented using Unified Modeling Language (UML) [6].

CMSD is a well-established standard by the Simulation Interoperability Standards Organization used for data exchange between simulation applications and other applications [14]. It is presented both in UML, demonstrating different data packages included in CMSD, and also as XML to enable data exchange between different applications. Information included can be process-related data such as layout, processes, scheduling, production flow, and inventory, etc. [6].

None of these standards can ensure a complete inclusion of all key parameters used within DES, but extensions can be made [6]. Among them, CMSD is the most commonly used for data exchange from and to simulation applications and has been shown to be more capable than other file formats regarding DES-related data exchange.

2.1.2 DES Application - Siemens Tecnomatix Plant Simulation

Siemens Tecnomatix Plant Simulation is a DES application developed by Siemens Digital Industries Software, designed to model, analyze and optimize manufacturing systems and logistics processes [15]. The application allows users to develop organized hierarchical models of industrial facilities, production lines, and production cells. It features predefined drag-and-drop objects and resources, making it more

user-friendly. For more complex manufacturing systems, control logic can be implemented through programming. When the default behavior of predefined objects is insufficient, the integrated programming language SimTalk can be used to define the simulation models behavior and logic [16].

The result obtained for using Tecnomatix plant simulations offer several benefits, including, cost reduction, lower investments risks, and optimized systems. The software is widely utilized across various industrial sectors, such as automotive, aerospace, military defense and logistics [15].

2.1.2.1 Standard Simulation Objects

Siemens plant simulation has built-in standard simulation objects with their own features and attributes [16]. Station, Assemblystation, Dismantlestation, Frame, Method, Buffers, Datatable, Button, and drop-down list.

- *The Frame object:* Is where the simulation model is created [17]. Simulation objects are inserted and combined to represent manufacturing processes. The Frame object can also be used for grouping objects and for building structured models. The entire simulation model can be represented in the main Model Frame, but subsections can also be built inside separate Frames and combined to a complete simulation model inside the main Model Frame.

- *The Interface object:* is a Material Flow Object which is used to connect frames to other frames or simulation objects [17]. It can be used both as an entrance and exit.

- *The Station object:* is a Material Flow Object designed to represent processing stations, such as machines[18]. It receives one Mobile Unit (MU) from its predecessor. After the set-up time and processing time have elapsed, the station object transfers the MU to its successors.

- *The AssemblyStation object:* is a Material Flow Object used for simulating assembly processes [18]. It moves the mounting parts either to the main MU based on the values entered into the Assembly Table, or it deletes them.

- *The DismantleStation object:* is a Material Flow Object used for simulating disassemble processes [18]. It removes mounting parts from the main MU or it creates new MUs.

- *The Buffer object:* is a Material Flow Object which can be used to store MUs between processing stations, ensuring a smooth flow of materials and preventing bottlenecks [17]. They can be configured to store a specific number of MUs and release them based on predefined conditions or rules.

- *The Method object:* is the container for user defined code and program routines [16]. It can be used for modify controls and executed during the simulation run.

- *The DataTable object:* is a list with several more columns [17]. The cells can contain information of different data types. Individual cells can be accessed via their index, designated by the number of the row and the number of the column.

- *The Button object:* is a User-Interface Object which can be inserted into the Frame [17]. When pressed, the button executes the program in the control. A Method can be selected as the control which make the button run the selected Method.

- *The DropDownList object:* is a User-Interface Object which enables the options to select an item from a predefined list [17].

2.2 Product Lifecycle Management

PLM is a management system for handling a company's products throughout their lifecycle [1]. It serves as a business strategy for managing company information and overseeing a product's lifecycle from its initial concept until the retirement of the product. A primary reason of using PLM is to integrate information and activities across different departments, thereby preventing issues such as contradictory data, information silos and redundant work. Since the tasks performed by various departments are often interrelated, they should be leveraged for cross-functional collaboration rather than allowing each department to work independently.

The implementation of a PLM system offers several advantages, including a reliable database and reduced development time due to more efficient data management [19]. Additionally, while enhancing management efficiency, it also reduces the management cost. Furthermore, the company gains a competitive advantage by adopting a technical strong solution for data managing, ensuring both consistency and high-quality data.

2.2.1 PLM Software - Teamcenter

Teamcenter is a PLM software developed by Siemens Digital Industries Software and is one of the most widely used PLM system [19]. It provides a centralized platform for integrating engineering, manufacturing and business operations, ensuring the utilization of reliable product information across departments. Teamcenter is an essential tool for maintaining competitiveness, particularly given the increasingly complexity of information within a company. Common type of information being managed include different documents, computer aided design (CAD) data, product data and workflow data that is shared [20].

When working with Teamcenter the software enables the creation of a comprehensive product structure within a shared database, allowing suppliers to get access and contribute with valuable data. Other useful functions includes for instance, Pro-

gram Evaluation Review Technique, which demonstrate the sequence of processes and equipment [21]. Additionally, the ability to add attributes is particularly useful when specific data needs to be extracted, provided that the necessary information has been properly recorded in the PLM system. Furthermore, for structuring information, particularly in the context of working towards manufacturing, the Manufacturing Process Planner function is appropriate. This type of structure should consists of the following elements [22]:

- *Manufacturing Bill of Material*: A detailed list of all materials and components required to manufacture a product.
- *Bill of Equipment (BOE):* A list of the equipment, tools and machines necessary for the manufacturing process.
- *Bill of Process (BOP):* A comprehensive list defining the processes and sequences involved in manufacturing a product.

2.3 Transport Systems

There are multiple variants of transport systems, such as "train" or "step by step" [23]. In a train transport system, every part moves simultaneously, and no delay time occurs. In contrast a step by step transport system works such that each part must wait for the part in front to move a certain distance before itself can start to move, causing a delay. The main difference is that a train system has zero delay time and is typically preferred. Delay time (DT) can especially be identified at elevators and turntables, where a part has to wait until the elevator or turntable returns to its original position and this time is represented by the DT. However, there are solutions to this, like for turntables that allow parts to enter and exit at the same time, drastically reducing the delay time.

Beyond these concepts, we have several other relevant terms related to transport systems [23]:

- *Buffer:* Can act as a storage, where a part can stand still. The purpose of its utilization is to better manage variations.
- *Throughput time (TT):* The time it takes for an part to travel to its next position. In a process, TT represents the time it takes for an part to enter an station and be processed.
- *Target cycle time (TCT):* Definition of the time, according to a company, that it should take to complete one cycle. I.e. how long time it should take to start and finish a step in the manufacturing system.
- *Cycle time (CT):* The actual time it takes in the reality to finish a cycle.

- *Standard stock in transfer (SSIT):* Refers to the parts that have to move in order to meet the TCT. It is calculated by dividing TT with TCT.

- *Standard stock in process (SSIP):* Refers to the parts being processed.

- *Availability:* The percentage representing when equipments/processes are working. Availability depends on the frequency of various failures.

- *Mean down time (MDT):* The average time that a process is unavailable or not operable.

3
Methodology

This chapter will describe the methods and studies used to collect information in order to answer the research questions. It will also cover how that information was applied in the development of the generic simulation tool. The methodology which was used is inspired by a Design Science Research Methodology (DSRM) [24].

3.1 Design Science Research Methodology

DSRM is a method used to develop a "Design research artifact" [24]. The involved activities in DSRM as shown in Figure 3.1 are Problem identification, Define the objectives for a solution, Design and development, Demonstration, Evaluation, and Communication.

The first two activities includes defining the specific research problem and later defining the objectives for a solution based on the problem definition [24]. This requires knowledge of the problem, the importance of its solution, and knowledge of what is possible. This information was gathered through current-state analyses from the literature and qualitative study. The third activity, Design and development, involves gathering theory which can be used to bear a solution and create the "Design research artifact" [24]. The literature and qualitative study were used to answer the research questions and to determine the desired features and functionality which was later used to develop the generic simulation objects and user-interface.

The fourth and fifth step, Demonstration and Evaluation, involves demonstrating how the "Design research artifact" works and evaluate how well the artifact solves the problem [24]. The demonstration comprised of several tests where production systems were defined in the project file using the User-interface, and later generated using the simulation tool. The Evaluation was designed to make sure that the simulation model was generated properly.

Lastly, Communication, which involves communicating the "Design research artifact", its effectiveness, and the theory it is based on [24]. The results from the studies and development was documented and is presented in Chapter 4. Documentation was also written explaining how the generic simulation tool and the user-interface works, and a standard work procedure was developed which will be used for knowledge transfer.

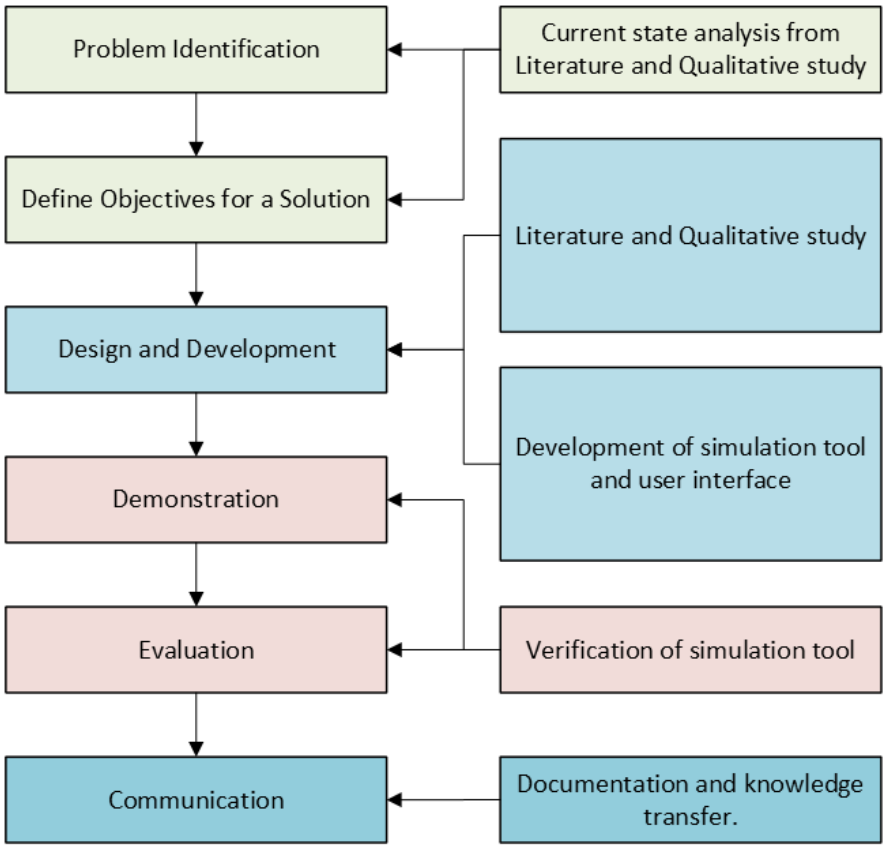


Figure 3.1: DSRM process

3.2 Literature Study

The literature study is a fundamental part of academic research projects, serving as a foundation for new research based on existing knowledge [25]. By considering empirical findings and different perspectives, it enhances the ability to address research questions more effectively. Additionally, It enables the identification of research gaps, which may suggest potential topics for future research or highlight areas that should be considered in a current research project [26]. Therefore, a literature study was conducted to examine the current state of knowledge on the integration between PLM and DES, generic data structures, and the automatic generation of simulation models. Additionally, capabilities of both DES and PLM were explored to find useful features for the proposed solution. Furthermore, additional literature was reviewed covering other relevant theories such as simulation data format and standards and internal transport systems.

The literature study was carried out by searching for relevant publications in the academic databases: Scopus, Science direct, Chalmers library, and IEEE Xplore. A method similar to a semi-systematic literature review was conducted. The aim was to synthesize the current state of knowledge, identify useful features and solutions, and narrow the scope of the review [25]. Instead of focusing on simulation in general, the review was limited to publications involving DES, making it more relevant for

the thesis. The keywords used in the search included: *Discrete event simulation, Product lifecycle management, Interoperability, Generic, Data Structure, Input data management, automatic simulation model generation, Automatic generation, and Automated input data.* In addition to this, snowball sampling was used to some extent. Where the reference lists of relevant publications was used to identify additional valuable sources [27].

To ensure the relevance of the publications, parts of abstracts, introductions and conclusions was read to ensure alignment with the research topic. The publication dates were also considered, were in general the most recent publications were prioritized [26]. In particular when authors had multiple publications on the same topic. Despite this, older publications (over 10 years) were included when still considered relevant, as the fundamentals of DES have remained largely unchanged [28]. Instead, most improvements in recent years have been related to areas such as visualization, simulation optimization and simulation speed. Another exclusion criterion was to focus primarily on the manufacturing sector.

3.3 Qualitative Study

A qualitative study is usually conducted to gain additional knowledge based on real-world experiences and to develop insights from practical understanding [29]. In this case the qualitative study involved collecting non-numerical data from interviews with stakeholders for each objective. The purpose was to gather information about the current state, and to determine desired features, functionalities, and process-related data necessary for the generic simulation object and user-interface. The study consisted of semi-structured interviews, designed with predetermined questions while allowing room for further elaboration. Additionally, unstructured interviews were conducted, in which participants freely discussed topics regarding DES and the thesis study.

3.3.1 Stakeholder Analysis

A stakeholder is defined as "any group or individual who can affect or is affected by the achievement of the organization's objectives" [30]. Stakeholder analysis is a method used to identify individuals or groups that may be influenced by a particular action or, in this case, a master's thesis study [31]. Additionally, this analysis examines the needs, influences, and authority of the stakeholders. Conducting such an analysis is crucial to succeed with a project, as it helps capture and fulfill stakeholders specifications while also leveraging the valuable information they possess [32]. This is particularly important in today's increasingly integrated environments, where no single person or group normally holds absolute authority.

Performing a stakeholder analysis at the project's initial phase was beneficial not only for identifying key stakeholders, but also for refining the project's scope, and clarifying the problem statement. By identifying and interacting with people who work in this field and possess knowledge unavailable elsewhere, combined with in-

ternal recommendations, key stakeholders could be determined [32]. The identified stakeholders, summarized in Table 3.1, included the project requester who initially proposed the problem. The main users, who are two engineers with more knowledge in the integration of Teamcenter and DES. In addition, two simulation engineers from other departments were included. Although they may not be direct users, their perspective could offer valuable insights worth considering during the development phase.

Table 3.1: Stakeholders

Stakeholder	**Role**	**Years of experience in DES**
Project requester	Simulation lead engineer	30 years
Main user	Virtual manufacturing engineer	2 years
Main user	Global area leader and simulation engineer	5 years
ME Propulsion components department	Simulation engineer for propulsion components	2 years
Plant business office department	Capacity and flow improvement engineer	2.5 - 3 years

Once the stakeholder analysis was completed, the next step was to assess how to gather stakeholder needs, specifications, and valuable information [31]. This was achieved through a combination of semi-structured and unstructured interviews.

3.3.2 Semi-structured Interviews

Semi-structured interviews were conducted to gain a better understanding of current practices in DES, what data is important, and desirable future improvements. Where the result could be used as inspiration for development of the generic simulation objects and the user-interface. These types of interviews are well-structured but allow for more flexibility, such as follow-up questions [33]. In addition, more in-depth knowledge can be obtained due to the open-ended questions, which allow the interviewee to elaborate further on a topic.

Before the interviews, the purpose of the thesis was introduced and a consent form was filled out. The consent form ensured that the participant was aware of their rights, such as the ability to end the interview at any time and the choice to decide whether or not to answer a question [34]. Furthermore, participants were asked for approval to record the interview to facilitate the data analysis process later on. The concent form can be seen in Appendix A.

During the interview phase, a five-step methodology was followed [33]. First, the purpose and objectives of the interview were established, focusing on current practices in DES, data management and desirable specifications for the project. After addressing these aspects, the next step involved developing an interview protocol

based on the objectives, as detailed in Appendix B. The protocol was structured as following: The initial segment featured open-ended questions, allowing participants to share and elaborate on their experiences while ensuring relevance to the project [34]. In the subsequent segment, the questions become more specific in order to obtain more in-depth knowledge relevant to the project's objectives. The concluding segment, remained open-ended, providing an opportunity to return to a specific topic, seek further clarification and wrap up the interviews. The third step involved assembling potential participants through a stakeholder analysis, which among other factors, supported the selection process. Additionally, Participants were identified through internal recommendation from the company. The fourth step was to select a platform for conducting the interviews, which were ultimately conducted in person with permitted audio recordings. The final step was to execution of the interviews.

Following the interview phase, all interview recordings were transcribed to create complete records of each conversations. Then, the answers from each interview were summarized to extract the key information and individual perspectives of the participants. Lastly, these summaries were synthesized, to identify recurring themes, enable the drawing of conclusions, and also highlight any differences in the answers.

The participants were kept anonymous to encourage truthfulness, thereby reducing the risk of withholding information [35]. However, one could argue that anonymity may negatively impact the reliability of the information provided. In this case, however, the participants were selected based on internal recommendations from the company, which enhances their credibility.

3.3.3 Unstructured Interviews

Some unstructured interviews were conducted throughout the project in the form of meetings with stakeholders and industrial supervisors. The purpose of which was to acquire a deeper understanding of DES and information regarding the project in general. These meetings were a less formal and conversational way of collecting data. Similarly to an unstructured interview, these meetings were about the general topic but with little to no predetermined format or specific questions [36]. Additionally, these interviews were conducted as a way to evaluate the solution, gathering iterative feedback for further improvements. The data was analyzed in a similar way to the semi-structured interviews but was summarized by taking meeting notes during the meetings.

3.4 Development of Generic Objects

A list of design specifications was created based on the result from the qualitative and literature study. The list was divided into "Needs" and "Wants" for the generic simulation tool and user-interface, detailing what they should include and be capable of. It also included the necessary input data parameters for accurate system behavior in the simulation model.

The first step in developing the simulation tool was to design how the generic simulation object should be represented and function. This included implementing DataTables and Methods, and defining the general area where the Interface objects and the Material Flow Objects such as Stations and AssemblyStations, are placed.

Once the generic simulation object was designed, several Methods was programmed to generate the object according to the established design. The Methods responsible for generating the objects are grouped in a frame called "_genericObjectMethods". These Methods were programmed in a generic way meaning that they used Anonymous Identifiers and could be copied and executed in any frame [17].

After completing the "_genericObjectMethods", "global" Methods were programmed in "_modelMethods", as per the specification list. These Methods would serve as the main functions of the simulation tool. The first Method to be programmed was the one which loads the project files and writes the desired worksheets into the desired DataTable. The second Method, "_generateModel", was programmed to use the information from the DataTables to build the simulation model. This Method was designed to delete previously generated models, create generic simulation objects, and also create the connections between them. Another Method was also made, including only the code for deleting the simulation model.

Lastly, a Method was programmed to list the files from the working directory into a drop-down list. To ensure only project files with the correct names and format are included, a cataloging function was implemented. Once all the Methods were complete, buttons were added to the simulation tool. These buttons were linked to specific Methods for a more user-friendly experience.

The entire programming process was iterative, with continuous improvements and changes. Minor changes included adding new input parameters to the project file, or making small adjustments to existing features when developing new ones. A major change occurred when the input data structure in the project file was changed to accommodate the user-interface. Although the function for reading the project file and writing the information into the data tables remained the same, extensive changes were required in order to handle the input data within the simulation tool.

3.5 Verification of Generic Simulation Tool

The generic simulation tool was continuously tested when implementing each step. First making sure that the generic simulation object was generated properly with the correct process-related data defined in the project file. Secondly, making sure that the entire simulation model is correctly generated. The Methods grouped in the frame called "_genericObjectMethods" is responsible for generating the generic simulation. These Methods were required to be copied into the generic simulation object and later executed in order to read the process-related data and generate the

generic simulation object. This meant that in order to test and validate those Methods, the copies had to be tested when they were inserted into the generic simulation object whilst the simulation model were generating.

Once the simulation tool was complete, tests were conducted with the intent of making sure that the automatic generation of simulation models worked as intended.

Three simulation models were defined in three project files using the user-interface. These different project files were moved into the working directory for the simulation object. Two invalid project files were also included into the working directory, one of which was wrongly named and one had the right name but the wrong file format. The simulation tool was used to generate the simulation models. To verify that the simulation tool worked as intended there were four tests:

- *Test one:* See if only the project files with correct name and file format were listed in the drop-down list "d_DropDownListExcelFiles".

- *Test two:* See if all of the input data from the project file is imported to the global tables correctly.

- *Test three:* See if the simulation model is generated or not. This includes cross checking the data table to make sure that all of the defined generic simulation objects have been generated.

- *Test four:* Check all generic simulation objects and make sure that they are generated as intended. This include checking that the data tables have been correctly generated with all the correct process-related data, making sure that all parts are included as the right standard simulation object, that all connections are as defined, and that the input parameters are inserted correctly.

3.6 User-interface

A structure for the input data was at first developed along side the generic simulation model. Making sure that all of the relevant data was included and that it could be loaded into the simulation tool properly. The structure was however changed when the user-interface was developed in order to achieve the desired features, such as being able to add new generic simulation objects, and add standard simulation object inside the generic simulation objects.

The user-interface was programmed with several macros and functions inside the project file. In order to ensure that the input data table had the correct format a template for a single generic simulation object was made. Macros were later programmed to copy this template to create an input data table. Data validation was also defined for some of the input data. These data validation were either made to only accept a certain type of value such as "time", or in some cases the data validation restricted the input to certain drop-down lists.

3.7 Standard Work Procedure

In order to clarify the use of the user-interface and the generic simulation tool, a standard work procedure were made. The documentation of this work procedure will contribute to knowledge transfer, aiming to maintain information and work progress. Knowledge transfer is an important aspect for companies, contributing to the development of capabilities that can be difficult to imitate, thereby enabling the creation of new applications and innovative performance [37].

The documentation of the work procedure sequentially explains, step by step, how to use the user-interface and the generic simulation tool. Additionally, it outlines the requirements for appropriate usage.

4
Results

This chapter presents the results from the project. First, findings from the literature study are presented, followed by the qualitative results regarding current practices and specifications from VCC. Lastly the result from the development of the generic simulation tool and user-interface are presented, as well as the proposed standard work procedure for knowledge transfer.

4.1 Literature findings

This section present the literature findings, which will be partially used to answer some of the research questions. First, four general methodologies of input data management are outlined. Then, different applied generic structures and automatic generation approaches are described more in detail. The purpose of this is to establish a theoretical framework for how generic simulation objects may be designed and how process-related data may need to be structured for greater compatibility. Lastly, potential features worth considering for the generic object are presented.

4.1.1 Input Data Management Methods

In a publication by Robertson and Perera [7], a general overview of input data management methods, particularly for DES was presented. Four methodologies were proposed, derived from observations and interviews, representing different levels of automation. The proposed methodologies were as follows:

(A) Manually data input - In this approach, information is collected manually from different sources. One advantage of this methodology is that the simulation engineer can verify the data. However, it has several disadvantages, including being highly time-consuming and inflexible when updates are required.

(B) Hybrid data input - Similar to the first methodology, data are collected manually from different sources but are stored in a centralized external source, such as a spreadsheet. This allows for automated integration into the simulation model. While the data collection process remains time-consuming, this methodology offers greater flexibility increases the flexibility for modifying data and updating the model. In a survey with simulation users from 86 different companies, it was found that this method was the most commonly used [38].

(C) Automatic data input - This methodology utilizes an external source (normally an intermediary simulation database) that is directly integrated with a company's different systems, such as enterprise resource planning (ERP) or PLM. As a result, data can be automatically extracted into the simulation model, potentially reducing the time-consuming process of data collection. However, challenges such as ensuring high-quality data must be properly addressed.

(D) Complete integration with data sources - In this fully automated approach, the simulation model is directly linked to for instance a PLM system, with data integration occurring in real time. This level of automation can significantly reduce the time required for data collection while minimizing errors, given that the required data are accurate and accessible. However, this approach also presents challenges, such as data redundancy across multiple locations, which can impact data quality and reliability. Additionally, limited accessible to certain data may pose further difficulties.

4.1.2 Generic Data Structure and Automatic Generation in DES

Research on generic data structures and automatic generation in the development of simulation models aims to reduce the time spent on data management, which is currently a time-consuming phase [4]. Additionally, by providing standardized approaches, this research could potentially facilitate greater modularity and interoperability between DES applications and other data sources, while enabling a general data structure that supports multiple DES applications [39]. Furthermore, the expertise required for simulation can be reduced through automatic generation, while simultaneously being cost-efficient. This is beneficial for SMEs that often have more economical constraints [40]. Next, a few examples of similar research within this field are provided.

4.1.2.1 Generic Data Structure

Based on the ISA-95 standard, Kardos et al. [5] proposed a generic data structure standards including the relevant enterprise levels as a foundation for automatic generation. According to them, a well-defined structure is essential to enable automatic generation of simulation models. Additionally, having a generic data structure facilitates and enables the integration of different input interfaces and simulation applications, as long as appropriate automatic generation algorithms are implemented. To describe a manufacturing system within this structure, the following key elements - equipment, process segments and product segments are specified by:

- *Classes*: Define the types and quantities of different equipment.
- *Process segments*: Represent time-based processes executed on equipment.

- *Specifications*: Describe how different elements are linked, such as which processes are assigned to which equipments.

- *Dependencies*: Describe the sequential relationship between process segments and how they depend on one another.

- *Attributes*: Assign key parameters to equipment and define the behavior (control logic) of both equipment and the entire manufacturing system .

In a similar manner to this structure, an alternative approach divides the data into distinct categories, such as facility data, manufacturing data also including product data and execution data [41], [42]. Each sets include the following:

- *Facility data*: Describes the production layout, including processes, workers, transport systems and buffers.

- *Manufacturing data*: Represent the behavior of the manufacturing system, encompassing process times, work schedules, products routing and other process-related parameters.

- *Product data*: Contains information such as product geometries, product variants, and assembly information.

- *Execution data*: Specifies how the model will be executed, including parameters such as warm-up time, products schedule and completion time.

As presented, by considering product and execution data in this structure, important aspects such as product variants and warm-up period are addressed within the simulation models [41].

4.1.2.2 Intermediate Database

One of the more commonly used methods for the automatic generation of simulation models is through the use of intermediate databases [38]. As presented by Robertson and Perera [7], data can either be added manually or potentially automatically integrated from a company's systems, such as ERP and PLM, into the intermediate database.

The main idea behind using an intermediary database is to gather all necessary process-related data and store it in a centralized external location [7],[41]. An general example of this is the generic data structure based on ISA-95, where Kardos et al. [5] developed a solution called "EasySim system". The system operates as follow: first data is collected from various sources, preprocessed, and then stored in a intermediary database. Algorithms within the DES application then generate the simulation model and calculate the predefined Key performance indicators (KPIs), with the results presented through an output interface shown in Figure 4.1. In this particular case Microsoft Excel was used as an intermediary database. However,

the authors mention that for more complex manufacturing systems involving larger volumes of data, Structured Query Language (SQL) and XML structures may be required for data pre-processing and storage, which in turn requires the use of a separate interface. What makes approaches like this beneficial is that they can be applied to and are compatible with several different DES applications.

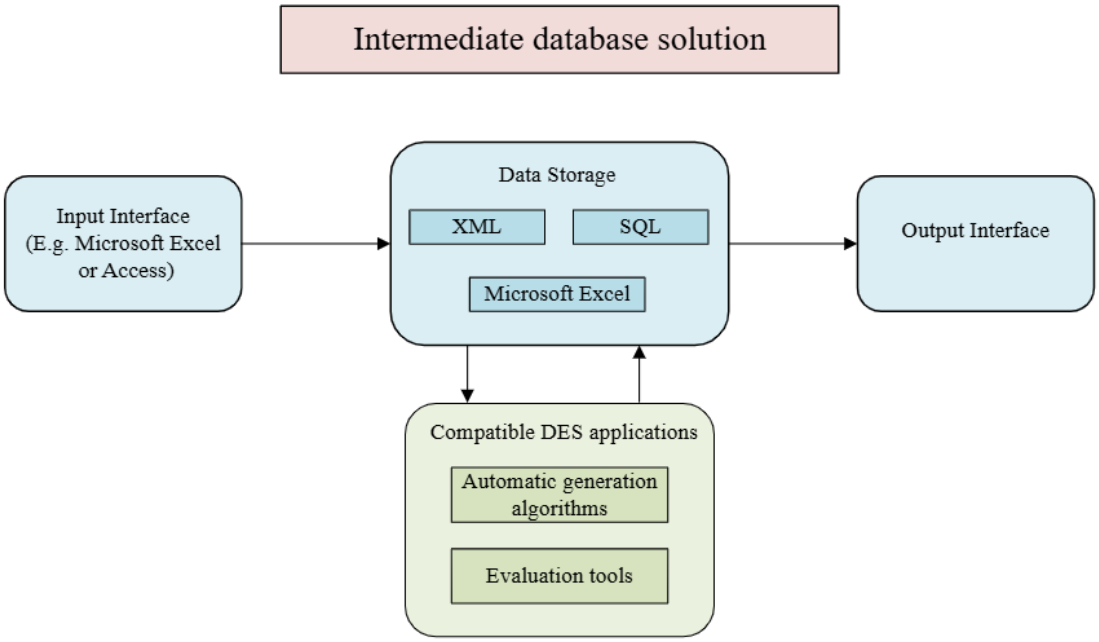


Figure 4.1: Intermediate database solution (adopted by [5])

In a study by Vik et al. [42], Microsoft Access and SQL queries were utilized for automatic generation and the storage of CAD and simulation data. The authors emphasized the importance of storing all data in an intermediate database, enabling centralized access to all data and supporting further optimization at a comprehensive system level. Burnett et al. [41] also supports the idea of storing all data in a single location. However, in contrast Microsoft Excel is used as an intermediate database, with data manually retrieved from a PLM system and structured across multiple sheets depending on type of data. Additionally, Microsoft Excel supports template-based representation of manufacturing systems, which has been shown to be effective in simulation modeling. This approach reduces modeling errors, as input data is already defined within the PLM system and can be extended to include additional data types if needed. Lastly, the general advantages of Microsoft Excel are its simplicity, accessibility and well-adopt, among SMEs as well [5]. It allows users to go through, modify, import and export input data without requiring any expertise in simulation.

4.1.2.3 Applications for Data Input Management and Conversion

Several publications have developed applications or interfaces that utilize standard formats to represent manufacturing systems. For instance, Jain and Lechevalier [40] used an interface supported by JavaScript to parse CMSD files in order to collect process-related data. While being effective in environments where data is already organized to a standard format, this approach may face limitations in real-word manufacturing settings. Since a common challenge in raw data extraction is the inconsistency of standardized data structures or that the data may originate from customized applications [4]. This variability complicates the data extraction process.

In response to these challenges, Skoogh et al. [4] introduced a tool called the Generic Data Management (GDM) Tool as an automation solution for input data management in DES. Instead of relying on existing standard format files such as CMSD, the GDM-tool does the opposite. It collects raw data directly from multiple sources, transform it into simulation-ready data, and then exports it as a CMSD file as demonstrated in Figure 4.2. Building upon this solution, Barlas et al. [43] developed an Open Source tool called the Knowledge Extraction Tool, which works in a similar way.

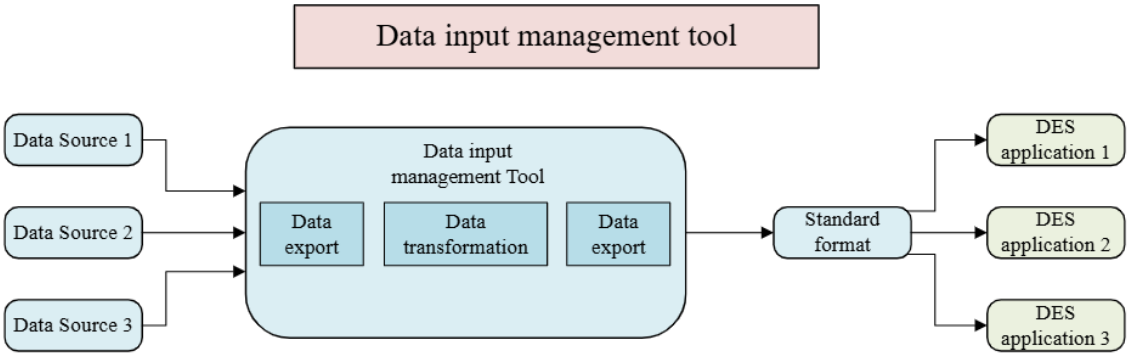


Figure 4.2: Data input management tool (adopted by [4])

Approaches like these offers a number of advantages. They automate large parts of the data management processes, reducing manual work and improving the data reliability [4],[43]. Additionally, by allowing collection of raw data it enables higher configurability with various company systems, which may also be customized for each company. This could be achieved, as demonstrated by D. Krenczyk et al. [44], by using a mapping technique to directly obtain data from systems such as MES and PLM. Moreover, by producing CMSD standardized outputs, it ensures compatibility with a wide range of DES applications, adhering to established simulation interoperability standards [4].

When working with CMSD it is of high importance to consider both the intended application and what the standard actual include. Bergmann and Strassburger [45]

used the CMSD format to model a job shop and identified several improvement potentials. They identified that attributes such as capacity, availability and mean time to repair (MTTR) are not included initially. Due to this, these attributes were added to represent, for instance, buffer sizes and the critical behavior of equipments during maintenance and breakdowns of equipments. Furthermore, dynamic behavior such as control logics are crucial in simulation models, but are not covered. Therefore, data regarding decision and routing rules were added as potential future extensions to the CMSD standard.

4.1.2.4 Direct integration

Very few publications on the direct integration between DES applications and company systems were identified. Kirchhof [46] developed a solution using a custom made data extractor to retrieve data directly from the company's ERP and MES system. However, there is limited information available on how the tool actually work and whether it can be utilized for an universal purpose.

The solution presented by Oppelt et al. [47] represents one of the closest approaches to direct integration between a simulation application and a company planning system that was found. They addressed previous challenges such as high complexity, manual input data management and utilization of proprietary mapping tools that require customization. The core idea behind the solution is to establish the planning tool as the only source of information. Then the aim is to enable the end-user to modeling the simulation model, with automatic model generation already configured, without requiring any setup by the end user. The methodology for direct integration can be divided into the following five steps as demonstrated in Figure 4.3:

Step 1: Create predefined simulation objects placed in a simulation component library, which is a one-time task. The author emphasizes the importance of close collaboration between the simulation engineer and the end users during this step, ensuring the needs are fulfilled.

Step 2: The objects in the planning tool are extended with essential simulation attributes, it is a one-time task. A close collaboration between the planning tool engineer and simulation engineer is recommended. Its due to that the simulation engineer knows better what attributes is needed and thereby facilitate the integration when right kind of data is added.

Step 3: A one-time mapping of existing process-related data from the main source to the simulation attributes in the planning tool. This establishes the planning tool as the single source of information. The mapping ensures that changes made by other engineers are immediately captured. It guarantees that simulation objects are configured with the latest updated and correct parameters. This approach enables automatic model generation, maintains data consistency, and minimizes errors.

Step 4: Implemented once, data exchange involves a mechanism within the planning tool that collects the simulation data and converts it to a standard simulation

format. When new models have to be generated, due to changes, this mechanism is utilized to automatically generate the information needed for the updated model.

Step 5: The model is generated by the end user. Some manual work is typically required to make the model executable, which is standard practice

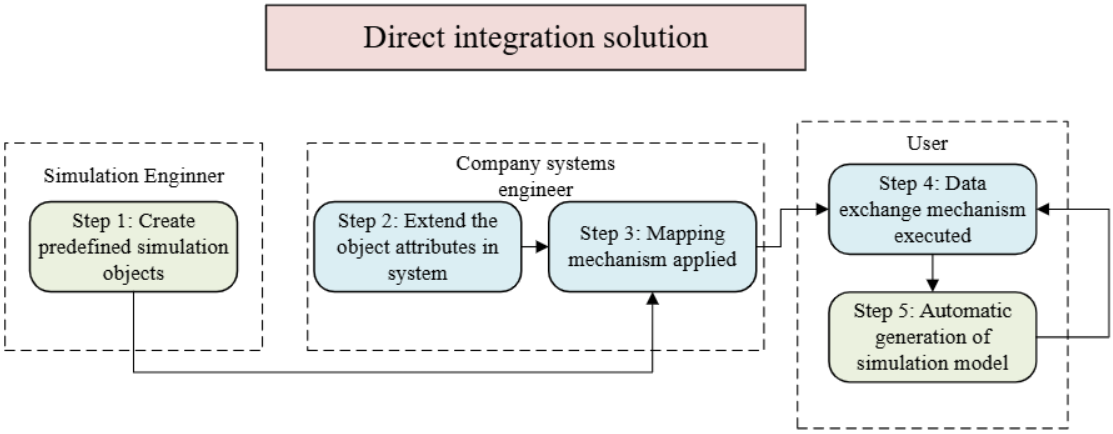


Figure 4.3: Methodology for direct integration (adopted by [47])

4.1.2.5 Alternative Approach: Data Derived From PLC

Several studies have investigated the use of Programmable Logic Controller (PLC) for the automatic generation of simulation models. One such approach involves utilizing the symbol table from a PLC program [48]. By having standardized naming rules for the PLC symbols, it becomes possible to include not only the control logic but also essential information about the plant. In another example, both a PLC program and a Manufacturing Execution System were used to gather input data [49]. The PLC program provided the control logic and the topology of a manufacturing system. Meanwhile, the MES complemented this with essential equipment parameters and "Scenario dependent dataset" Shown in Figure 4.4.

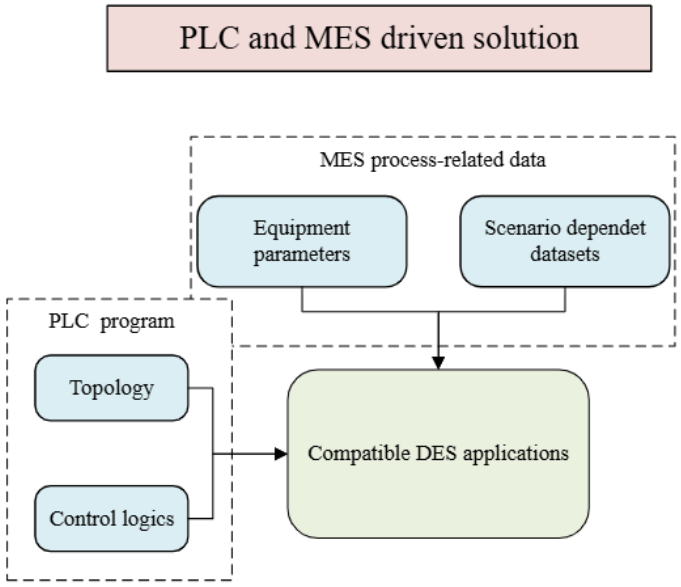


Figure 4.4: Data gathering supported by PLC and MES (adopted by [49])

4.1.3 Potential features

This section present inspiration for potential features of the generic objects are presented. For instance, R. Hughes et al. [50] described several internal functions using programming code in DES that are useful for automatic generation of simulation models. Specifically, they used functions for reading all data into predefined variables, building the layout with the included equipment, and routing the flow between stations.

In contrast Burnett et al. [41] uses internal tables to store data, each corresponding to specific data such as processes parameters, part routing, included processes etc. These tables are then refereed to during the model generation, and additional libraries could be included such as custom objects and custom model code. Including a structure for model code can reduce the manual work of defining control logic, which is typically required after simulation model generation. Bergmann and Strassburger [45] also emphasizes the importance of considering the dynamic behav-

ior. They suggests an extension of CMSD including decision and routing rules as a representation of dynamic behaviors.

Having different buttons can also be an additional function, enabling users with no expertise to automatic generate models. The implemented buttons can represent different actions such as resetting a model, importing data, and running the simulation [41].

4.2 Results from Qualitative Study

This section presents the current state of DES practices at VCC. The insights from the different engineers highlight established approaches, suggestions on alternative approaches, and also what limitations that exists. Additionally, it covers what input data are required, how it is collected, and how it is structured. These findings formed the design specification for the development of the generic simulation object and, the user-interface.

4.2.1 Current State of DES

DES Plays a significant role at VCC for the analysis of manufacturing flows and their resulting outputs. It is also used for the verification and validation of different concepts, encompassing both the development of entirely new production lines and plants (greenfield) and the optimization of existing manufacturing lines (brownfield). Today, the modeling of the simulation models are done manually, but there is potential for more efficient methods.

One potential alternative for increasing efficiency in modeling could be a centralized and structured data environment offered by PLM systems. However, the adaption and maturity of PLM structures vary considerably across different departments and plants within VCC. While some have integrated processes and equipment data, essential process-related parameters are often lacking. Furthermore, many manufacturing systems are not comprehensively and consistently set up. Another approach to reduce some of the manual effort involves the utilization of custom objects within simulation. Two of the simulation engineers use it to some extent, but it depends on the use case. Like for instance, if the system has a tendency to change a lot, or if it is an already installed manufacturing system (brown field) or a new manufacturing system (green field). The complexity of the simulation model also matters. If it is more complex and changeable, it is usually easier to start from scratch since adaption of the custom objects has to be done anyway. However, for faster analysis and faster modeling, it can be beneficial to utilize custom objects.

It was identified that all the simulation engineers seem to follow a similar abstraction level structure for their simulation models, using a "process level" and an "equipment level". This can be comparable to the BOE and BOP structures in a PLM system. For example, a specific step in the manufacturing process, such as a robot cell or an assembly station, is represented in the simulation model as a station object with

all important parameters assigned to it. Then the equipments in that process are considered by having a table of equipment failures (breakdown, maintenance, etc.) linked to the process object.

Regarding the representation of transport system within simulation models, the most common approach is that one buffer space corresponds to a single station object in the simulation model. Sometimes, a standard simulation buffer object is used, and then the capacity has to be set. Specifically, two of the engineers emphasize the importance of including transport systems in the simulation models, especially when their analysis focus on the quantity of pallets/skids and buffer dimensions. However, it is noteworthy that transport systems are currently excluded from VCC PLM system.

The approach to visualization of the simulation model varied among then engineers. The engineer from the ME Propulsion components department have more illustrative models, utilizing 3D and a layout design that replicates the real or planned manufacturing line. This is done to facilitate understanding for customers and make it look more appealing. While some other engineers generally considers it a waste of time, arguing that the most important part is the output of the simulations.

4.2.2 Input Data Parameters

For new models, a clear definition of the layout is necessary, requiring prior definition of the process steps and the flow. If these are defined within PLM, the structure of the process steps could be extracted from the BOP. Regarding the input data, different time variables are required, such as TCT, CT, process times and transport times are needed. What was identified was the inconsistent definition of these times throughout the company, leading to potential confusion and inaccuracies. To address this, it was proposed that time could be divided into TT and DT. Where DT includes, for example, when an elevator returns to a pickup location or when a turntable has to turn back to pick up the next part, causing a delay. Beyond TT and DT, technical availability, and different failure times has to be included, also potentially scrap rate, product variants, SSIT and SSIP. In terms of output data, jobs per hour (JPH) is considered the most essential parameter to evaluate potential cost savings or the necessity for further optimization and investments.

There are some difference between various processes and objects. Transport objects have very high availability (this parameter is often negligible), and achieving an optimal SSIT with good buffer utilization is also desirable. Additionally, clarifying whether a transport object operates as a “train” or “stop and go” system is important, as this potentially requiring different control logic. Furthermore, buffer objects need to have a defined capacity, which is not the case for processes handling one part at the time. In contrast, process objects are typically more detailed, and the parameters values included within the process steps, such as different failure data, scrap rate etc., needs to be defined. Failure data is more significant for process objects than transport objects, due to transport objects high availability. Despite

these differences, a process structure can generally be generic, allowing parameters to be skipped if not essential for a specific process or object.

4.2.3 Input Data Management

Currently, input data for particularly greenfield projects can be collected from VCC libraries. Unfortunately, these libraries are not regularly updated, especially not the failure data. Instead, it was concluded that a system based on real process-related data from similar lines, with frequent updates, would be preferable. An alternative approach suggested by one of the engineers was to directly get all essential parameters from the customer/supplier via a standardized Excel sheet. This approach also shifts the responsibility for input data away from the simulation engineer, as it is predefined and no longer requires manual implementation. It is noteworthy, particularly for greenfield projects in an early phase, that it can be a tendency there is a lot of changes. Due to this, layouts and parameters values has to be updated continuously. In contrast brownfield projects, a lot of existing information can be collected and converted into functional data. However, this process requires knowledge of where to find the information, by knowing the right people across different plants that can provide the data needed. This difficulty arises from the current lack of a centralized data source, resulting in a time-consuming input data gathering phase and significant challenges in accessing all required data.

In general it is the lack of updated data appears to be one the most significant limitation. All engineers highlighted the particular need for current and continuously updated failure data. Beyond failure data, control logic is typically missing. Control logic can be for instance, the requirement for a process to wait for a signal before being able to entry/leave a station. Implementing control logic requires proper understanding of the processes and is to a high extent a manual effort. An idea emerging from a workshop, was to link generic objects to predefined control logic. However, control logic are not universally applicable and significant manual efforts remains necessary. An example of such control logic could be the pull principle, where parts moves at the same time. Availability and scrap rate are also usually inaccurate and require updates when real data becomes available. While specific times are often unavailable at the beginning of a project, some estimations is often made with safety margins.

A common understanding is that it would be desirable to gather all process-related data in a centralized PLM system. This requires an organizational structure to function effectively, which is not currently in place. The engineer from the plant business office department believes that while a PLM system is desirable, Siemens Teamcenter might not be the optimal solution. Instead it is suggested that a data source via sharepoint could potentially be easier to maintain and keep updated. It would also present a lower barrier to entry for individuals who do not regularly use Teamcenter. In contrast, the main users are positive about using Teamcenter, in their view, the structure particularly the BOE and BOP are suitable. Overall PLM could serve not only as the source for modeling specific simulation model, but could

also be a useful database for future projects when making assumptions based on prior situations. However, one of the engineers argues that for certain projects, a PLM system is not necessary. The reason is that the project would not proceed faster, as the customer would still need to fill in the data, whether in Teamcenter or an Excel sheet. There is an agreement that gathering all the information from different projects in one centralized location could be beneficial for future projects.

4.2.4 Design Specifications

Based on the result from the qualitative study, design specifications including needs and wishes for the generic simulation tool were summarized in Tables 4.1 & Table 4.2 respectively. The needs represents mandatory features that have been incorporated into the generic simulation tool. The user-interface has to some extent, taken these needs and wishes into consideration, particularly regarding the generic data structure, the requirement to consider transport systems, and the necessary input data parameters.

Table 4.1: Design specifications, needs

Needs	Reason
Automated input data import to DES application from predefined data file.	Shifts the responsibility for input data away from the simulation engineer, also reducing errors made by the simulation engineer.
Modeling of simulation model through automatic generation.	A way to reduce manual effort.
Generic data structure including process level and equipment level.	Structure used by all simulation engineers, also similar to Teamcenter's structure of BOP and BOE.
Transport systems must be considered and included.	Important aspect to consider, especially for analyses done for the quantity of skids/pallets and buffer dimensions.
Each transport step/buffer location should be defined as one simulation station object.	More useful when using the generic simulation object approach, removes the need for a "capacity" parameter.
Inclusion of the following input data parameters for both generic objects and the user-interface: TT, DT, Availability, MDT Connections.	Most important key parameters for accurately representing a manufacturing system, enabling possibilities of having an output of critical KPIs such as JPH and resource utilization.
Global tables in the DES application containing different types of input data.	Structure the data in a clearer way into different tables. It will be easier to find information regarding certain data and facilitate the debugging process.
Output tables in the DES application presenting the generic simulation objects and the objects contained within them.	This enables the validation that all systems and processes have been implemented as intended.
Cataloging of input data files with a proper naming convention, that should be selectable from a drop-down list within the DES application.	Ensures the correct files are loaded and gather all relevant files simultaneously.

The wishes represents desired features, and only few of them was included due to time limitations. The ones included were the local tables and the translation table.

Table 4.2: Design specifications, wishes

Wishes	Reason
Inclusion of the following input data parameters: SSIT, optimal SSIT, scrap rate and conveyor speed	Additional information valuable for the simulation models, like inclusion of conveyor speed for more illustrative models.
Local tables in the different generic simulation objects.	Allow changes/experimentation without affecting the main input data and the entire model.
Further cataloging: Correct sheet names should be included; if not correct, they will be disregarded. The column names must match.	This is done to minimize errors for input data management.
Translation table from a selected object type to a object used in the DES environment.	Enables the user to freely chose what object type they want. Then the chosen object is compared against a table to determine the corresponding DES object, into which it will then be translated.
Have predefined control logic (Methods in Simtalk) linked to different generic simulation objects.	Reduce some of the manual effort for implementing control logic.
Output data on JBH and utilization.	Present what is of actual interest.

4.3 Results from Development of Generic Simulation Objects

This section presents the outcomes of developing the generic simulation objects, including a description of the utilized generic data structure. Furthermore, it details the building of the automatic generation tool, including its functions and objects.

4.3.1 Description of the Utilized Generic Data Structure

For the generic simulation object a hybrid input data management method is utilized. As real data are unavailable and the focus is on demonstrating a methodology for effectively structuring data to facilitate the integration between a PLM system and DES applications, the parameters values are made up. In this solution, process-related data is defined (manually using the user-interface) and structured in an Excel project file serving as an intermediate database. This can then enable for the automatic generation of a simulation model.

Identifying that all engineers followed a similar abstract level structure, and building upon findings from earlier utilization of the ISA-95 standard, a generic data structure was designed accordingly. This main structure shown in Figure 4.5 was defined as follows: first, having generic simulation objects representing either transport systems or the manufacturing processes. For transport system the type of transport object and buffer dimensions were included, where each transport object and each buffer slot correspond to one simulation object. For processes, typically there is only one simulation object for the actual process, along with other simulation objects representing for instance, the loading/unloading of parts. Additionally, parameters such as TT, DT, MDT, and availability are assigned. The availability and MDT represents equipment failures, where it include the impact of the actual equipments involved in a process without the need for one simulation objects for each of the equipment. Furthermore, additional structures are presented in the same document but in separate Excel sheets. These include the description of the manufacturing flow (connections) and translation tables that convert an object type into an appropriate DES object.

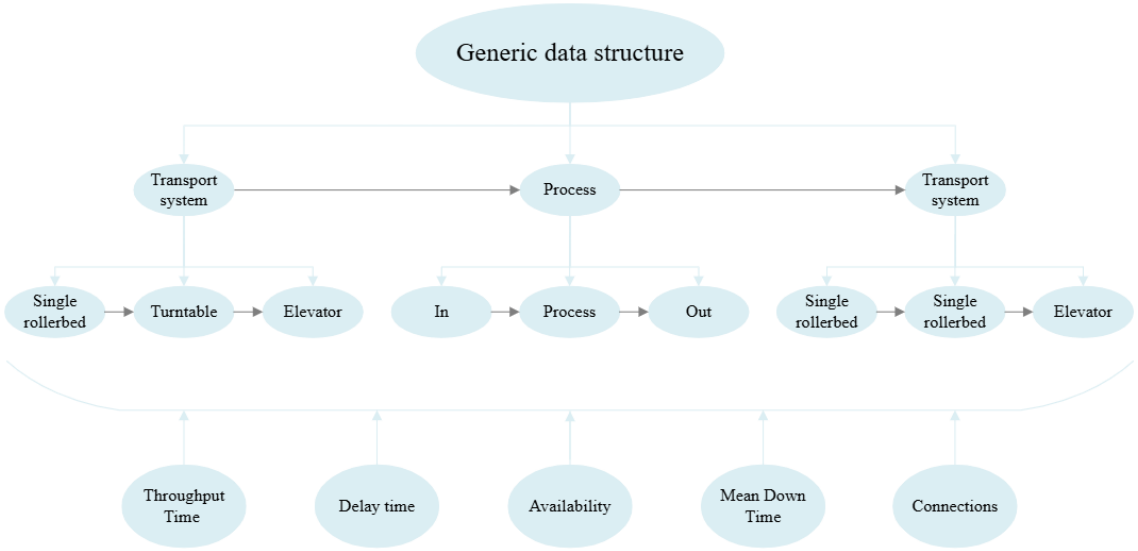


Figure 4.5: The designed generic data structure

4.3.2 Structure of the Built Generic Simulation Tool

The generic simulation tool is programmed inside a siemens plant simulation model file. The generic simulation tool is organized into several folders and simulation Model Frames, the names of which follows a consistent naming structure. Folders and frames used for the simulation tool are prefixed with an underscore, followed by a descriptive name e.g., the folder "_methods" and the frame "_modelMethods". Similarly, the different simulation elements, Methods, tables, buttons and variables are named with an initial letter (representing type of simulation object), an underscore, and a descriptive name e.g. m_generateModel and t_objectMethods. The reason for this naming structure is to recognize what parts of the simulation model is custom made and also easily understand the function of each different objects.

The different parts of the simulation model are divided into three folders: "_simulationModels", "_methods", and "_tables", all located within the Class Library. Each of the folders have different frames which include the simulation model, Methods and tables. The frame "_generatedModel", which is shown in Figure 4.6, is were the simulation model will be automatically generated. The generic simulation objects will appear as frames representing the different transport systems and processes and their connections. Additionally, a drop-down list with the different project files and the buttons for triggering actions are included. Furthermore, two key variables also exists, one indicating the most recently loaded input data file, and the other showing the most recently generated input data file. These variables are included in order to reduce errors and ensure the intended input data file is used.

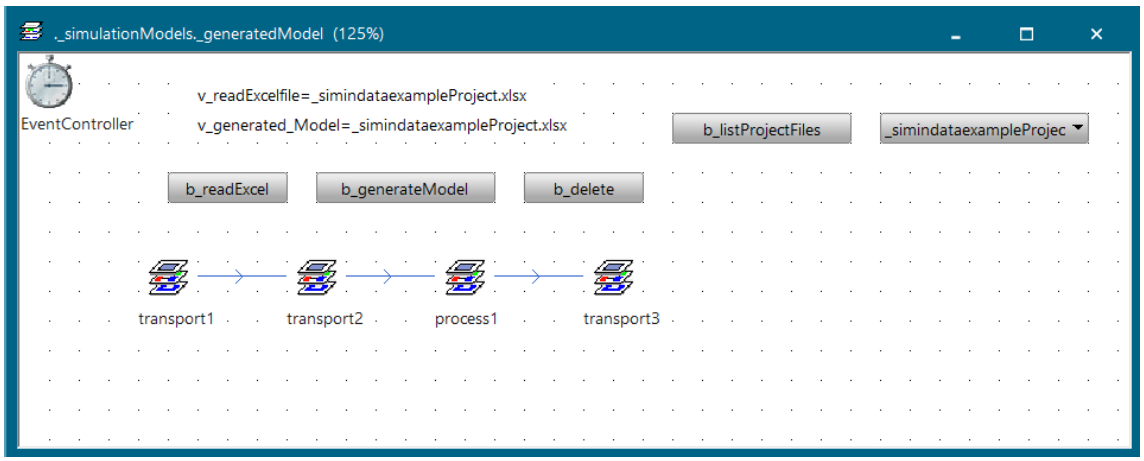


Figure 4.6: The frame "_generateModel" with an example project generated

The other two folders contains frames for all the Methods and tables. The Methods consists of all the programming code used for all features and conditions in the generic simulation tool. The code is commented so that it would be easier for a future developer to understand, reuse features, and be able to make changes. The tables are used to store different input data and information about which simulation objects that have been created. The generic simulation tool with the frames "_generateModel", "_modelMethods", "_genericObjectMethods" and "_allInputData" can be seen in Appendix C.

4.3.3 Generation Methods

There are two groups of Methods developed to generate the simulation model. There are the ones located in the frame "_modelMethods", which are the global Methods. They are responsible for inserting project file names in to the drop-down list, reading the project files and filling the input data tables, generating the simulation model, and one for deleting the generated simulation model. The other group are local Methods they are located in the frame "_genericObjectMethods". They are responsible for the modeling within the "generic simulation objects". These Methods are

developed for creating standard simulation objects, tables for the input data and connections, and insert the relevant data from the global tables. There is also the Method which utilizes all this data to generate all of standard object which is defined in the "generic simulation object". All of these Methods are automatically inserted into the generated object frame and executed, and later deleted. Which result in a generic simulation object which is generated from the input data. An example of a generated generic simulation object can be seen in the Figure 4.7.

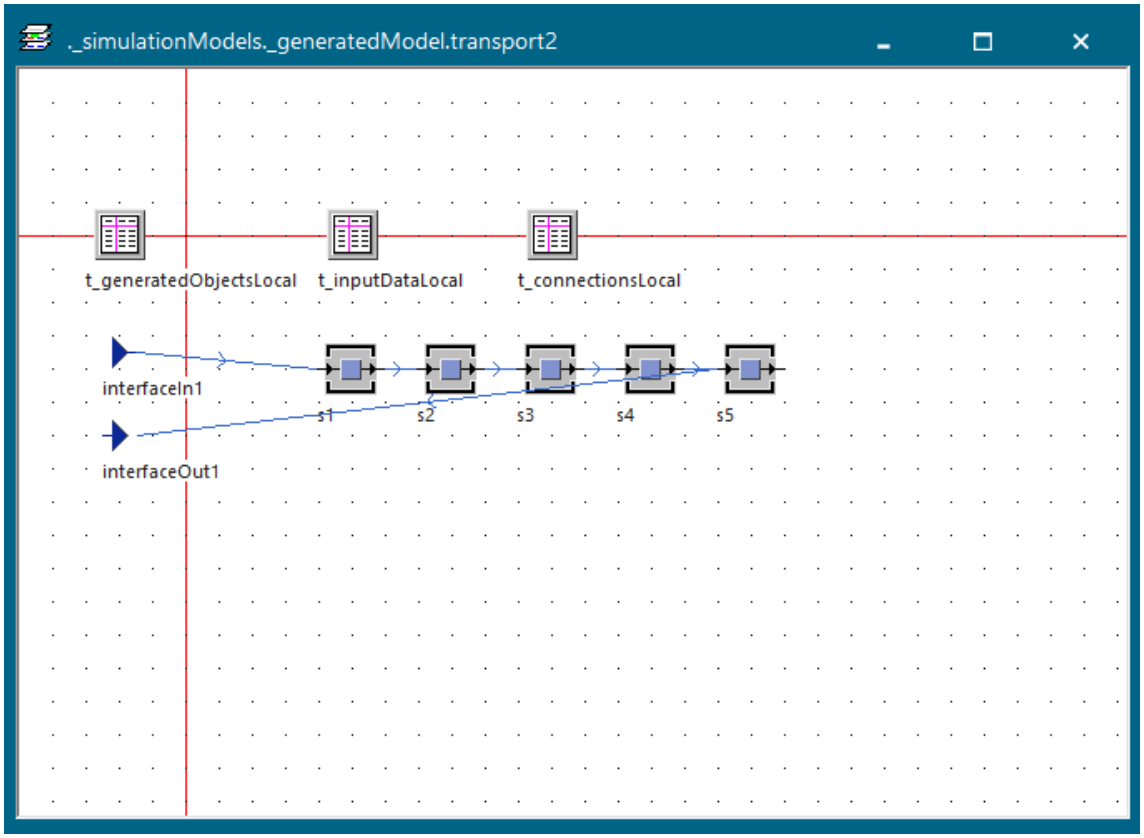


Figure 4.7: Example of Generated Generic Simulation Object

4.3.3.1 Global Methods

The main functions of the simulation tool is programmed into the global Methods in "_modelMethods". Each global Method specifically contributes as follow:

- *m_listProjectFiles:* This Method initially starts with clearing any previously loaded files from both the drop-down list "d_DropDownListExcelFiles", and the DataTable, "t_excelFiles". It then scans the working directory for files with the correct file format and name. If such project files are found, they are added to both the drop-down list and the DataTable.

- *m_readExcel:* Loads the data from the selected project file into the different global DataTables. It includes the following data: main input data, manufacturing flow data, required local Methods and a translation table.

- *m_generateModel:* This is the primary generation Method in the tool. Initially the Method checks the DataTable, "t_generatedObjects", if any simulation object have been previously generated. If there is a generated simulation object, it gets deleted and the DataTable gets cleared. Thus deleting the simulation model. The Method then starts generating the new simulation model by creating each new generic simulation objects. At first an empty frame is created and placed in the defined spot, the interfaces are later added which represents the entries and exits. Furthermore, a table is generated within the frame to specify which simulation objects are included in the generic simulation object. The Method then proceeds to insert and execute the local Methods relevant to the generic simulation object. Once all of generic simulation objects have been generated, connections between them are created.

- *m_deletemodel:* This Method only deletes the generated simulation model. It checks if any simulation object have been previously generated. If there is a generated simulation object, it gets deleted and the DataTable gets cleared. Thus deleting the simulation model.

4.3.3.2 Local Methods

The local Methods are used to generate everything included within a specific generic simulation object. Each contributes specifically as follows:

- *m_createDataTable:* Creates the local input data table for the generic simulation object. An additional column is included for "standardDesObjects".

- *m_readGlobalInputData:* Reads the global main input data into the local input data table. The defined object type is later compared with the DataTable "t_objectTranslation" to define what standard DES object should represent it in the DES application. This result is defined in the column "standardDESobject" in "t_inputDataLocal".

- *m_createConnectionTable:* Creates the local connection table for the specific generic simulation object.

- *m_readGlobalConnectionTable:* Loads the global connection data into the local table

- *m_createObjects:* Creates all the standard simulation object included, with names and input data assigned to each object. It also creates the connections between the objects.

4.3.4 Tables

Similar to the Methods, there are both global and local tables, with most of the local tables inheriting information from the global tables. First, the "t_excelFiles" table lists all data input files with the correct prefix "_siminput" from the project directory. Then, the "t_objectData" global table shown in Appendix D contains the main input data, presenting the generic simulation objects and the objects within them that are intended to be automatically generated. Additionally, the table includes critical parameters for ensuring correct manufacturing system behavior. Next, there is the "t_connections" global table, which describes the flow both between transport systems and processes, as well as the connections between transport objects shown in Table 4.3. The interfaces included in the tables represent the entries and exits of a system (which there can be multiple of).

Table 4.3: Example of a connection table

FromObject	ToObject	FromFrame	ToFrame
interfaceIn1	e1	transportsystem1	transportsystem1
e1	s1	transportsystem1	transportsystem1
s1	s2	transportsystem1	transportsystem1
s2	interfaceOut1	transportsystem1	transportsystem1
interfaceOut1	interfaceIn1	transportsystem1	process1
interfaceIn1	in1	process1	process1
in1	welding	process1	process1
welding	out1	process1	process1
out1	interfaceOut1	process1	process1
interfaceOut1	interfaceIn1	process1	transportsystem2
interfaceIn1	s1	transportsystem2	transportsystem2
s1	s2	transportsystem2	transportsystem2
s2	t1	transportsystem2	transportsystem2
t1	interfaceOut1	transportsystem2	transportsystem2

Furthermore, there are two more global tables where one of them includes what Methods needs to be executed for a certain generic simulation object. The other one is a translation table "t_objectTranslation" where the selected object type is translated into an appropriate DES object shown in Table 4.4 as an example. Lastly an global output table exists presenting what generic simulation objects has been made.

Table 4.4: Example of translation table

objectTypes	standardDesObjects
elevator	station
turntable	station
singlerollerbed	station
in	station
marriage	assemblystation
unmarriage	dismantlestation
out	station

Regarding the local tables, there are three in total. These are located within the generic simulation object. Similar to the global tables, there are tables for the local input data, local connections and what objects that has been created in the specific generic simulation object.

4.3.5 Drop-down list

A projects input data file can be selected through the drop-down list. This drop-down list contains all project files that is located in the Plant simulation working folder, which is the folder where the simulation model is located. The Method for reading and creating this drop-down list does so by checking the current directory folder for project files files with correct naming structure. The filenames are later inserted into the drop-Down List so that the user can select which project input data file they want to generate a simulation model with.

4.3.6 Buttons

As shown in Figure 4.6, there are four buttons, all of which are connected to different Methods. The button object is used to simplify the users experience when wanting to execute a specific Method. It eliminates the need to find the appropriate Method and execute it manually. The buttons are connected to the Methods with the same descriptive name so for example the button "b_readExcel" is connected to "m_readExcel".

4.4 Developed User-interface

The user-interface was based on the input data format in the project file. The user-interface is an Excel file comprised of five worksheets. The first is the "object-Template", shown in Appendix E.1. This worksheet has two parts the first part has several named lists, similar to Table 4.5, which are referenced and used throughout in the user-interface. Firstly, "objectType" is a list of the different types of generic simulation object that can be defined with the User-interface. The other lists include what objects that can be included inside the different generic simulation objects. The other lists are named the same as the items in the list "objectType". This is because these lists will be used as drop-down lists and the first one will reference

the others.

Table 4.5: ObjectTypes

objectTypes	transportObjects	processObjects
transportObject	elevator	in
processObject	turntable	marriage
	singlerollerbed	unmarriage
		out

The other part of the "objectTemplate" worksheet is the generic simulation object template. This template include the name, object type, x and y coordinates, and number of input and outputs for the generic simulation objects. It also includes the name object type, TT, DT, Availability and MDT for the simulation objects inside the generic simulation object. Some of the columns/cells uses Data Validation in order to ensure correct input parameters. For the cells which defines the object type for the generic simulation object, an drop-down list is available. This selection decides what options will be available when defining the object types for the simulation objects inside the generic simulation objects. The other Data Validation are in the column MDT and restricts the data input to time (HH:MM:SS).

The next two worksheets are used to define the simulation object and the connections in the simulation model. The worksheet "objects" is where the generic simulation objects are defined by the user, shown in Appendix E.2. At the top of this worksheet there is a field for defining the project name and two buttons, as shown more clearly in Figure 4.8. The first button is labeled "Add Object". This button will execute a macro which copies the generic simulation object template and adds it to the first empty columns. A user can add as many objects as needed and define each object. The second button is labeled "Generate Project File". This button executes a macro which creates an project file. If the project name is "exampleProject, then the name for the project file will be "_simindataexampleProject". The macro will copy all of the data defined in the user-interface and paste it in the project file. The "connections" worksheet, shown in Appendix E.3, is where the flow of the entire simulation model is defined. The connections are defined with the columns Fromobject, toObject, FromFrame, ToFrame.

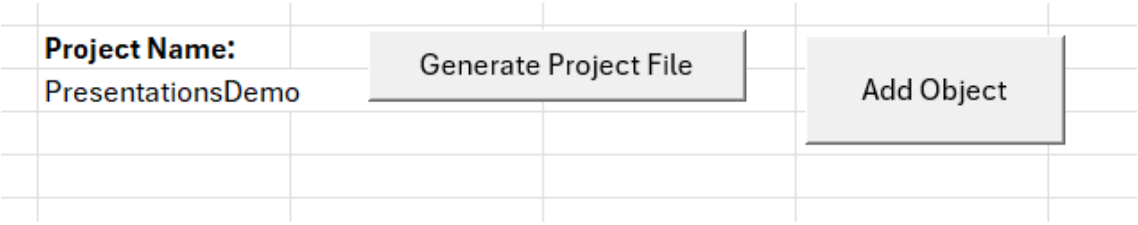


Figure 4.8: User-interface: buttons

The last two worksheets are used in the simulation object to enable the simulation model to generate. They are included inside the project file in case any of their corresponding dataTables in the simulation model needs to be updated. The "objectMethods" worksheet, shown in Appendix E.4, lists all of the Methods which is needed to be included to generate the different types of generic simulation objects. The "translation" worksheet, shown in Appendix E.5, include defines what standard simulation objects the different simulation object should translate to.

4.5 Standard Work Procedure

A step by step standard work procedure were done for how to appropriately use the user-interface and the generic simulation tool. Initially, the different generic simulation objects are defined in the user-interface.

4.5.1 Standard Work Procedure for user-interface

The work procedure for the user-interface is conducted as follow:

Step 1: Open the user-interface and make sure that the worksheets "objectMethods", "translation" and "objectTemplate" are updated to latest version to define the intended simulation model.

Step 2: In the worksheet "objects" add simulation object by pressing the button "Add object". (You either add and define the objects one at a time or all of them at once).

Step 3: Define all of the simulation objects with process related data.

Step 3.1: Define the name for the generic simulation object.

Step 3.2: Select the object type for the generic simulation object from the drop-down menu that will appear. This will decide what object types will be available to select when defining the object type for the objects inside the generic simulation objects.

Step 3.3 Define the rest of the generic simulation object with x and y coordinates, and number of input and outputs.

Step 3.4: Define the names of the objects within the generic simulation objects.

Step 3.5: Select the object type for the objects from the drop-down menu that will appear.

Step 3.6: Define the rest of the object with TT, DT, Availability and MDT.

Step 4: Go to worksheet "connections". Define the flow by filing in the columns Fromobject, toObject, FromFrame, ToFrame, for each connection.

Step 5: Click the button "Generate Project file" to generate a new project file with the input data defined in the user interface.

4.5.2 Standard Work Procedure for Generic Simulation Tool

The work procedure for the generic simulation tool is conducted as follow:

Step 1: Initially, ensure that the project file is placed in the correct project directory. It should be in the same as the simulation tool.

Step 2: Then press the button "b_listProjectFiles" to load the compatible project input data files in to the drop-down list "d_DropDownListExcelFiles".

Step 3: Select what project file you want to generate, in the drop-down list "d_DropDownListExcelFiles".

Step 4: Press the button "b_readExcel" to read all the different input data into the different global tables within the simulation model. Verify that the correct file has been loaded, by checking the value of "v_readExcelfile" variable.

Step 5: Press the button "b_generateModel" to automatically generate the simulation model with generic simulation objects. Confirm that the intended model has been generated by examining the value of the "v_generated_Model" variable.

5
Discussion

In this chapter, the research questions are addressed. Then the methodology used for this master thesis is discussed, why it was used and how it may have affected the results. In addition, future work and future research are presented. Lastly, the ethical, societal, and ecological aspects are discussed in terms of how they are taken into consideration.

5.1 Research Questions

RQ1: How can a generic simulation object be designed so that it can accurately represent multiple manufacturing processes and transport systems, and enable automatic generation of simulation models?

RQ1 was addressed by considering the design specifications and current practices identified from the qualitative study, as well as insights gained from developing the generic simulation tool. Building on the observation that all engineers followed a similar level of abstraction, and drawing inspiration from an example of prior utilization of the ISA-95 standard, a generic simulation object was developed. The VCC practices was taken into account to ensure usability and relevance, which was also supported by the literature.

The abstraction levels "process" and "equipment" levels were used. Where a generic simulation object could represent either a manufacturing process or a transport system, each with assigned key parameters. To facilitate for a generic approach, the actual process and each transport step or buffer slot was represented by a single object in the DES application. This enabled a flexible and generic approach, where the same set of parameters could be applied to any simulation object or left out if not relevant. Furthermore, different processes or transport system could be presented using different DES objects, but this was clarified through a translation table ensuring correct mapping. Equipment considerations were integrated by assigning relevant equipment-related data directly to each process or transport step/buffer slot.

The VCC engineers highlighted the need for defining the layout, process and transport steps and the manufacturing flow. The key parameters included TT and DT for comprehensive time-based operations behavior, as well as Availability and MDT to account for equipment failure data. Parameters regarding the connections for the

manufacturing flow was also required. Control logic was identified as a limitation, often requiring significant manual effort and deep knowledge of processes.

Beyond actual things directly connected to the generic simulation objects, several critical and value-adding features were implemented to facilitate automatic generation and enhance user-friendliness. These are summarized below:

- *Intermediate database (Microsoft Excel):* User-friendly database with centralized data, but with clear distinction of different input data in separate sheet such as main input data, manufacturing flow and a translation table. Increased flexibility as it is applicable to different DES applications.

- *Generic simulation tool structure:* Organized folders for simulation models, Methods and tables with clear naming conventions. This enhances the clarity of the tool, makes it easier to navigate.

- *Global generation Methods:* Used for the main actions in the tool, such as reading the input data, automatically generating the simulation model, and providing the ability to delete the generated simulation model.

- *Local generation Methods:* Handling different parts of the automatic generation and is divided into different Methods to improve clarity.

- *Global and local tables:* Used for having a structured way of storing the input data, manufacturing flow data, and translation table.

- *Drop-down list for input files:* Enables an easy way to select between different project input data files.

- *Buttons for actions:* Simplified execution of the main actions of the tool, where the buttons are linked to the global Methods.

RQ2: How can process-related data from a PLM system be effectively structured and integrated into DES applications to ensure accurate system behavior?

RQ2 was answered to much extent based on the literature, but is supported by current practices at Volvo and the solution provided in this thesis. To effectively integrate process-related data from a PLM system into DES applications and ensure accurate system behavior, a structured and standardized approach is essential. A structure adapted from the ISA-95 standard was identified, serving as a framework for data exchange between systems [5]. This structure aligns closely with how simulation engineers at VCC organize their simulation models. Both approaches emphasize the use of distinct abstraction levels for processes and equipment, and prioritize the representation of time-dependent operations over simulating each individual equipment included in a process.

With this structure consisting of process segments, specifications, dependencies, and attributes, several similarities were identified between the ISA-95 adapted structure and the simulation approach at VCC [5]. These shared principles form a strong foundation for determining what process-related data that should be included and extracted from a PLM system:

One of the main similarities lies in the concept of process segments, which represents time-based operations, including both manufacturing processes and transport steps [5]. These segments are closely aligned with the generic simulation objects developed in this study, and form the foundation blocks of the simulation model. At VCC, a common current practice is to modeling on a "process" level rather than inlcluding each individual equipment as a simulation object, which reflects the ISA-95 adapted structure. Additionally, if processes (and transport steps) is properly structured within the PLM system's BOP, there is a potential to extract the structure directly from PLM to define the layout of a simulation model

Specifications describe how different elements are linked to each other, particularly how equipment is assigned to processes [5]. At VCC, instead of modeling the equipment, it is considered for by assigning relevant equipment-related data such as failure data (availability and MDT) directly to the associated processes.

Dependencies refers to how processes depend on each other [5]. This is comparable to the logical flow modeling at VCC, which includes both process and transport steps. There is also potential to extract the manufacturing flow directly from the PLM's BOP structure, if well-developed. Accurately capturing the manufacturing flow is essential for realistically replicating system behavior.

Finally, attributes define the key parameters of each process or transport step [5]. In VCC case, this includes the required input data such as TT, DT, availability and MDT. Another important and more complex attribute is the control logic. That also to a higher extent, define the system behavior. Implementing control logic usually requires manual input and deep process knowledge, making it one of the more challenging aspects of simulation modeling. This challenge has been acknowledged in earlier research as well [45],[47].

Given VCC emphasis on a defined layout, well-defined process steps, along with the ISA-95 adapted structure, a generic data structure was developed in this thesis. The utilization of several distinct substructures ensures clarity and enhances the practicality of automatic generation of simulation models. The structure was divided as following: the first structure includes the process steps and transport steps represented by generic simulation objects, along with all key parameters. The second defines the manufacturing flow. The third serves as a translation table, translating object types into corresponding DES objects.

Regarding the integration of a PLM system and DES applications, several approaches are identified in the literature: complete manual input, utilization of an

intermediate database (as utilized in this thesis), conversion tool, or direct integration [4],[5],[7],[47]. As highlighted in the literature and supported by interviewed VCC engineers, a commonality across these approaches is the importance of centralizing the data. Furthermore, having parameters assigned to the process and transport object is emphasized, whether through mapping, extensions of data formats, or by adopting a similar structure as this thesis solution [44],[45]. While the Methods differ in the level of automation, the underlying principle remains, to establish a single, reliable source for all information to be used in DES.

Having a completely manual approach is very time consuming and therefore not a preferred alternative. While intermediate database have been used to a high extent in previous cases [38], their effectiveness relies on the inclusion of automatic extraction from the PLM system. This automation is crucial in order to gain the benefits of the approach and avoid the time-consuming manual data input phase. Furthermore, when utilizing standard formats, it is important to consider what data is included and what potential extensions that may be necessary. Regarding direct integration, close collaboration between different departments and a well-established planning system are essential. This is also emphasized by VCC engineers, who highlighted the importance of organizational structure, in order for such an integration to work, also noting that it may be a significant barrier for not daily users, which further emphasizes the importance of strong collaboration between different departments.

RQ3: How can a user-interface be designed to enable the configuration of transport systems for a DES environment?

RQ3 was addressed by considering the data structure used in the generic simulation tool, as well as results from the literature and qualitative study.

In the absence of having process-related data in a centralized PLM system the user-interface was designed to serve the role of an intermediate database. The user-interface would be used to define a project file which would be used in the simulation tool to generate a simulation model. The input data management method of collecting data and using a user-interface to store it in an Excel file makes this an Hybrid data input method.

Burnet et al. [41], presents benefits of using Microsoft Excel as an intermediate database were data is manually retrieved from a PLM system. Microsoft Excel supports template-based representation of manufacturing systems which has been shown to be effective in simulation. This approach has shown to reduces modeling errors, as input data is already defined within the PLM system and can be extended to include additional data types if needed. The authors, recognizes that the structure can facilitate the future development in automated integration with PLM systems. Lastly, the general advantages of Microsoft Excel is simplicity. Despite the need for manual data input, this solution enables non-expert users to create simulation models more efficiently It allows users to go through, modify, import and

export input data without requiring any expertise in simulation.

Other approaches than using intermediate databases were found in the literature study such as using the GDM-tool to transform raw data into simulation-ready information as an CMSD-file. As highlighted by Skoogh et al. [4], one of the key advantages of these approaches is their ability to maintain direct connections to original data sources rather than relying on intermediary databases, ensuring that simulation models are fed with accurate and current data. Taken together, these capabilities position the authors approach as a more appropriate and future-oriented solution for managing input data in complex, dynamic manufacturing environments. CMSD lacks however several key parameters, including buffers, capacity, and Mean Time to Repair (MTTR). Therefore, an extension of the CMSD-file format would be necessary [45].

The user-interface have different worksheets which enables the user to define transport systems with generic simulation object. It is structured to be compatible with the simulation tool. It utilizes a template for the generic simulation object to enforces the structure which is essential for automatic generation. The user-interface enables the user to define parameters and process-related data such as names, object type, TT, DT, Availability, MDT, coordinates, entrances, exits, and connections.

In addition to the data structure, other value-adding features were implemented into the user-interface for an easier user experience and to enforces the structure. These features were:

- *A button to add objects*: A button which automatically adds an object to the input worksheet, according to the established template.
- *Data validation*: Data Validation which restricts the user to define parameters with correctly formatted values.

5.2 Contributions and Limitations of the Thesis

The methodology used for this master thesis was a simplification of DSRM, as it normally requires several years of research work. The reason it was chosen as the method is that it follows a systematic procedure, with well-defined steps that facilitate thorough and organized research work. Additionally, it focuses on designing practical and innovative artifacts, which in this case can be compared to the generic simulation tool and user-interface, and thereby make it relevant, as its aim is to improve current practices.

As for the literature study, it presents the current state of knowledge, identifies remaining research gaps regarding comprehensive interoperability, and provides several concepts for input data management and automatic generation. The identified publications included a mix of more recent and older literature, where it can be argued that some literature may be outdated given the rapid digitalization transformation. However, it was identified that the fundamentals of DES have remained consistent and are therefore still relevant [28]. The scope was narrowed down to

DES, excluding other types of simulation, and primarily focused on the manufacturing sector to enhance relevance. It can be argued that this may exclude relevant publications and limit the generalization of the findings.

For the qualitative study consisting of interviews, it provided insights regarding current practices and different perspectives. Simulation engineers from different departments were interviewed to capture potential differences in approaches, thus broadening the perspectives. These interviews also provided valuable input on how data should be structured at different levels and which types of data are considered essential. However, a limitation of this approach is that only internal interviewees were included, including external interviewees could have yielded further insights and perspective. However, it can be argued that since the project was for VCC, adaptions should be made to the company to ensure usability and relevance. It is however important to recognize that if the qualitative study included a bigger group of simulation engineers from different companies the simulation tool and user-interface might have other input data and function differently when generating the simulation model.

Combining the literature study, the qualitative study and the development of the generic simulation tool was beneficial for comparison. This approach allowed for triangulation when addressing research questions, enabling findings and information to support or contradict each other. This strengthens the credibility of the conclusions, as they are supported by multiple sources. However, having to conduct all three of these in the same project may have limited to which extent the studies could be performed, which could have an impact on the overall result.

As mentioned, considerations has to be made regarding the development of the simulation tool and the user-interface. VCC requested that the project file had to be an Excel file and that the user-interface serve as a stand-in for an intermediate database. However, the advantages of this solution are that Microsoft Excel is user-friendly and not limited to a specific DES application. This flexibility enhances interoperability and increases adaptability, making the user-interface applicable to other companies and compatible with various DES applications. Moreover, it could lower the barrier for SMEs with limited resources and expertise, enabling them to benefit from DES without requiring advanced knowledge. Other alternatives for input data management, such as the GDM tool and direct integration with company systems, were identified but remained outside the scope of this project.

The thesis have contributed with a generic simulation tool which utilizes generic simulation objects to automatically generate simulation models. To enable automatic generation, the process-related data have been structured inside a project file. This thesis also included the development of a user-interface and a standard work procedure which a user can follow to define these project files. As the user-interface is a stand-in solution for a future PLM system these project files will contribute as suggestion of what data should be included in a future PLM system and how the data should be structured.

5.3 Future Work and Research

The main purpose for developing the simulation tool was to see if a simulation model could be generated with process-related data. The data would be extracted from a project file (Excel file). The user-interface was developed as a stand in intermediate database since the PLM integration in VCC remains limited. Therefore, future work would include extending VCC's PLM system to include process-related data used to generate simulation models. This would also include defining transport systems as these are not currently defined in VCC's PLM system. This would mean that entire production systems would have to be defined inside the PLM system where each of the process steps would have to be linked with the relevant process-related data. There would also be the need to develop a function to generate a project file like the ones that is defined using the user-interface.

While the simulation tool and user-interface developed have the functionality of including Methods used for generating the simulation object it shows the potential of including control logic which have been one of the main limitations of the different data management methods. Future work could therefore be to see if this can be included in the generation of the simulation model and see if the model generation remains generic and flexible.

Another thing to consider in future work, which is not covered by this thesis project, is variant dependent cycle times. In some cases there are multiple different parts that are being processed in the same manufacturing line and require different cycle times. This is demonstrated by Burnett et al., which then also includes product schedule [41].

One aspect of the future work is the input data management system chosen for the generic simulation object. The chosen method is the one which fit the result for this thesis but a desired version of this would be Automatic data input or Complete integration with data sources, as described in subsection 4.1.1. This would reduce the time-consuming processes of data input management however the problems of these methods would need to be addressed.

Future research includes investigation of a data structure that encompasses all process-related data. While the developed solution includes input parameters and connections to represent the production flow it still lacks the inclusion of control logic. Similarly, Bergmann and Strassburger identified that the CMSD-format, while being very capable, does not include capacity, availability, MTTR, and control logic, and suggests these as potential extensions.

5.4 Ethical, Societal and Ecological Aspects of the Thesis

As the master thesis was conducted at VCC it was important to treat the company information securely. Regarding the selection of interview participants, the primary criterion was their experience with DES simulation, and efforts were made to ensure equality beyond that. Furthermore, all interviewees were provided with a consent form including their rights, emphasizing voluntariness, and guarantee of ending the interview at any time. To further respect the participants, their anonymity has been preserved.

Regarding the ecological perspective this master thesis aligned with the following SDG goals [51]:

- *Goal 9: Industry, Innovation and Infrastructure:* Our solution foster innovation in the manufacturing industry by making DES work more efficient. It also enables SMEs with limited resources and expertise to better utilize DES tools, thereby contributing to a more sustainable industrialization.

- *Goal 12: Responsible Consumption and Production:* With DES in general, concepts can be tested beforehand without having to do a real implementation, saving us from spending unnecessary resources. In addition, this master thesis results can further facilitate this kind of work further with more efficient information and resource use.

- *Goal 13: Climate action:* As mentioned, by striving for more efficient processes, with as good resource utilization as possible, this project can contribute to a more positive environmental impact

6
Conclusion

The automotive and manufacturing industries are experiencing a transformation through the integration of advanced simulation technologies and data management systems. DES are becoming essential to optimize production system processes and reducing the risk of inefficiencies. The current practices at VCC for modeling simulation models involve creating new simulation objects for each case. This approach is not only time-consuming but also requires extensive customization. Simulation projects are generally constrained by factors such as time and expertise, which can lead to significant financial costs. Therefore, it is not considered appropriate or recommended to undertake a simulation study if there are insufficient resources or time.

This thesis has explored the development of generic simulation tool, and a user-interface designed to automatically generate DES models of manufacturing systems. The simulation tool utilizes generic simulation objects which were designed to fit the abstraction level of current practices at VCC. They can be defined to effectively represent transport and manufacturing systems each with assigned process-related data. The user-interface is developed to be compatible with the simulation tool and the template-based representation enforces the structure necessary for automatic generation.

Centralized data are essential to facilitate input data management and enable smooth automatic generation. This requires an organizational structure that functions effectively, utilizing for instance, either a mapping technique or a project input file extraction feature similar to the one in the user-interface. For a future PLM system, it is essential to include all the manufacturing processes and transport systems. Additionally, updated input data should be directly assigned to the different process and transport objects within the PLM system. This will facilitate a smooth extraction of layout data, process-related data and the manufacturing flow data.

In conclusion, while DES is a valuable tool at VCC, its current dependence on manual modeling highlights opportunities for efficiency gains through enhanced integration with PLM systems. The insights and solutions developed from this master thesis demonstrate the potential for improving the utilization of DES. Furthermore, this work lay the foundation for future work and research towards a complete PLM integration and the development of a comprehensive data structure including control logic and other essential parameters. Lastly, the results represent a meaningful step toward more efficient, scalable, and integrated simulation practices during the ongoing digital transformation in the manufacturing sector.

Bibliography

[1] J. Stark, *Product Lifecycle Management*, 5th ed. London, U.K: Springer, 2022.

[2] A. Zvantesson and M. Andersson, "Discrete event simulation as a tool for virtual commissioning," M.S. thesis, Dept. Elect.Eng. Chalmers University of technology, Gothenburg., Sweden, 2018.

[3] E. Babulak and M. Wang, "Discrete event simulation," in *Discrete event simulation*, A. Goti, Ed., Rijeka, Croatia: InTech, 2010, ch. 1.

[4] A. Skoogh, B. Johansson, and J. Stahre, "Automated input data management: Evaluation of a concept for reduced time consumption in discrete event simulation," *SIMULATION*, vol. 88, no. 11, pp. 1279–1293, 2012. DOI: 10.1177/0037549712443404.

[5] C. Kardos, G. Popovics, B. Kádár, and L. Monostori, "Methodology and data-structure for a uniform system's specification in simulation projects," *Procedia CIRP*, vol. 7, pp. 455–460, 2013. DOI: 10.1016/j.procir.2013.06.015.

[6] J. Zhao, E. Aghezzaf, and J. Cottyn, "Interoperability performance evaluation for discrete event simulation models: A step towards multi-level data exchange," in *34th CIRP Design Conference.*, 2024, pp. 72–77.

[7] N. Robertson and T. Perera, "Automated data collection for simulation?" *Simulation Practice and Theory*, vol. 9, no. 6-8, pp. 349–364, 2002. DOI: 10.1016/S0928-4869(01)00055-6.

[8] A. Sudhakaran and V. Solanki, "Data enhancement in teamcenter for des (discrete event simulation)," M.S. thesis, Dept. Industrial and Material Science. Chalmers University of technology, Gothenburg., Sweden, 2024.

[9] J. Banks, J. S. Carson, B. L. Nelson, and D. M. Nicol, "Introduction to simulation," in *Discrete- event system simulation*, 4th, Upper Saddle River, New Jersey, USA: Prentice Hall, 2004, ch. 1.

[10] S. Brailsford, L. Churilov, and B. Dangerfield, *Discrete-Event Simulation and System Dynamic for Management Decision Making*, 1st ed. Chichester, U.K: John Wiley & Sons, Ltd, 2012.

[11] A. Luder, N. Schmidt, and R. Rosendahl, "Data exchange toward plc programming and virtual commissioning: Is automationml an appropriate data exchange format?" In *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, Cambridge, UK: 2015, pp. 492–498. DOI: 10.1109/INDIN.2015.7281783.

[12] T. Czvetkó and J. Abonyi, "Data sharing in industry 4.0 - automationml, b2mml and international data spaces-based solutions," *Journal of Industrial Information Integration*, vol. 33, 2023. DOI: 10.1016/j.jii.2023.100438.

[13] ISA-95, *Isa-95 series of standards: Enterprise-control system integration*, Accessed: Mar. 13, 2025., Jan. 2025. [Online]. Available: https://www.isa.org/standards-and-publications/isa-standards/isa-95-standard.

[14] SISO, *Standard for core manufacturing simulation data – xml representation*, Accessed: May. 6, 2025., Aug. 2012. [Online]. Available: https://www.sisostandards.org/page/StandardsProducts.

[15] *Siemens tecnomatix plant simulation*, Accessed: Feb. 14, 2025. [Online]. Available: https://www.indx.com/en/product/siemens-tecnomatix-plant-simulation.

[16] S. Bangsow, "Simtalk and dialogs," in *Tecnomatix Plant Simulation*, 2nd ed., Cham, Switzerland: Springer International Publishing, 2020, ch. 2.

[17] *Plant simulation help*, Accessed: May. 2, 2025. [Online]. Available: https://www.indx.com/en/product/siemens-tecnomatix-plant-simulation.

[18] S. Bangsow, "Modeling of production processes," in *Tecnomatix Plant Simulation*, 2nd ed., Cham, Switzerland: Springer International Publishing, 2020, ch. 3.

[19] Y. Zhang and H. Cai, "Research and implementation of product lifecycle management platform based on teamcenter," *Advanced Materials Research*, vol. 538-541, pp. 2961–2966, 2012. DOI: 10.4028/www.scientific.net/AMR.538-541.2961.

[20] J. Niemann and A. Pisla, "Teamcenter data management," in *Life-Cycle Management of Machines and Mechanisms*, M. Ceccarelli, Ed., vol. 90, Switzerland: Springer, 2021, ch. 19. DOI: 10.1007/978-3-030-56449-0_19.

[21] Y. Liu, X. Zhan, and X. Li, "Method for realizing structured process through secondary development of teamcenter system," in *2020 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS)*, Shenyang, China: 2020, pp. 512–515. DOI: 10.1109/ICPICS50287.2020.9202321.

[22] C. Scheffer, *How well do you know your product?* Accessed: Feb. 25, 2025., Sep. 2023. [Online]. Available: https://blogs.sw.siemens.com/teamcenter/teamcenter-structure-management/.

[23] L. Asplund and D. Andersson, "A guide to transport systems," *unpublished (internal document)*, pp. 1–9, 2023.

[24] K. Peffers, T. Tuunanen, M. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of Management Information Systems*, pp. 45–77, 2007.

[25] H. Snyder, "Literature review as a research methodology: An overview and guidelines," *Journal of Business Research*, vol. 104, pp. 333–339, 2019. DOI: 10.1016/j.jbusres.2019.07.039.

[26] O. Paul, "The nature and purpose of a literature review," in *Suceeding with Your Literature Review : a Handbook for Students*, 1st ed., Maidenhead, U.K: McGraw-Hill Education, 2012, ch. 1, pp. 4–21.

[27] C. Parker, S. Scott, and A. Geddes, "Snowball sampling," *SAGE Research Methods Foundations*, 2019. DOI: 10.4135/9781526421036831710.

[28] A. J. Collins, F. S. A. Pour, and C. A. Jordan, "Challenges and the future of discrete event simulation," *The Journal of Defense Modeling and Simulation*, vol. 20, no. 3, pp. 351–369, 2022. DOI: 10.1177/15485129211067175.

[29] S. B. Merriam and E. J. Tisdell, "What is qualitative research," in *Qualitative Research : A Guide to Design and Implementation*, 4th ed., San Francisco, CA, USA: Cambridge University Press, 2015, ch. 1, pp. 3–21.

[30] R. E. Freeman, "The stakeholder concept and strategie management," in *Strategic Management: A Stakeholder Approach*, Cambridge, U.K: Cambridge University Press, 2010, ch. 2, pp. 31–51.

[31] J. Gilstein, "Stakeholder analysis," *Salem Press Encyclopedia*, 2024.

[32] J. M. Bryson, "What to do when stakeholders matter," *Public Management Review*, vol. 6, no. 1, pp. 21–53, 2004. DOI: 10.1080/14719030410001675722.

[33] T. George, *Semi-structured interview/ definition, guide examples*, Accessed: Jan. 28, 2025., Jan. 2022. [Online]. Available: https://www.scribbr.com/methodology/semi-structured-interview.

[34] A. Galletta, "The semi-structured interview as a repertoire of possibilities," in *Mastering the semi-structured interview and beyond*, 1st ed., New York, NY, USA: New York University Press, 2013, ch. 2.

[35] A. Novak, "Anonymity, confidentiality, privacy, and identity: The ties that bind and break in communication research," *Review of Communication*, vol. 14, no. 1, pp. 36–48, 2014. DOI: 10.1080/15358593.2014.942351.

[36] C. Wilson, "Chapter 3 - unstructured interviews," in *Interview Techniques for UX Practitioners*, 1st ed., Burlington, Massachusetts, USA: Morgan Kaufmann, 2014, ch. 3.

[37] A. J. Gil, D. Romero-Daza, L. Rodriguez-Cavides, and C. Tobias, "Learning culture and knowledge transfer: The role of colombian employees attitudes," *Knowledge Management Research Practice*, vol. 22, no. 6, pp. 632–645, 2023. DOI: 10.1080/14778238.2023.2277929.

[38] A. Skoogh, T. Perera, and B. Johansson, "Input data management in simulation – industrial practices and future trends," *Simulation Modelling Practice and Theory*, vol. 29, pp. 181–192, 2012. DOI: 10.1016/j.simpat.2012.07.009.

[39] S. J. E. Taylor and et al., "Grand challenges for modeling and simulation: Simulation everywhere—from cyberinfrastructure to clouds to citizens," *SIMULATION*, vol. 91, no. 7, pp. 648–665, 2015. DOI: 10.1177/0037549715590594.

[40] S. Jain and D. Lechevalier, "Standards based generation of a virtual factory model," in *2016 Winter Simulation Conference (WSC)*, Washington, DC, USA: 2016, pp. 2762–2773. DOI: 10.1109/WSC.2016.7822313.

[41] G. A. Burnett, D. J. Medeiros, D. A. Finke, and M. T. Traband, "Automating the development of shipyard manufacturing models," in *2008 Winter Simulation Conference*, Miami, FL, USA: 2008, pp. 1761–1767. DOI: 10.1109/WSC.2008.4736264.

[42] P. Vik, L. Dias, G. Pereira, and J. Oliveira, "Automatic generation of computer models through the integration of production systems design software tools," *International Journal for Simulation and Multidisciplinary Design Optimization*, vol. 4, no. 3-4, pp. 141–148, 2010. DOI: 10.1051/ijsmdo/2010018.

[43] P. Barlas, C. Heavey, and G. Dagkakis, "An open source tool for automated input data in simulation," *International Journal of Simulation Modelling*, vol. 14, pp. 596–608, 2015. DOI: 10.2507/IJSIMM14(4)3.306.

[44] D. Krenczyk, W. W. Kempa, K. Krzysztof, C. Grabowik, and I. Paprocka, "Integration of manufacturing operations management tools and discrete event simulation," in *IOP Conference Series: Materials Science and Engineering*, vol. 400, 2018. DOI: 10.1088/1757-899X/400/2/022037.

[45] S. Bergmann and S.Strassburger, "On the use of the core manufacturing simulation data (cmsd) standard: Experiences and recommendations," in *Fall Simulation Interoperability Workshop 2015*, Orlando, FL, USA, 2015.

[46] P. Kirchhof, "Automatically generating flow shop simulation models from sap data," in *Proceedings of the 2016 Winter Simulation*, Duesseldorf, Germany, 2016, pp. 3588–3589.

[47] M.Oppelt, G. Wolf, O. Drumm, M. S. B. Lutz, and L. Urbas, "Automatic model generation for virtual commissioning based on plant engineering data," in *IFAC Proceedings Volumes*, vol. 47, Cape Town, South Africa, 2014, pp. 11 635–11 640. DOI: 10.3182/20140824-6-ZA-1003.01512.

[48] H. T. Park, J. G. Kwak, G. N. Wang, and S. C. Park, "Plant model generation for plc simulation," *International Journal of Production Research*, vol. 48, no. 5, pp. 1517–1529, 2009. DOI: 10.1080/00207540802577961.

[49] G. Popovics, A. Pfeiffer, B. Kádár, Z. Vén, L. Kemény, and L. Monostori, "Automatic simulation model generation based on plc codes and mes stored data," *Procedia CIR*, vol. 3, pp. 67–72, 2012. DOI: 10.1016/j.procir.2012.07.013.

[50] R. Hughes, R. Scott, and K. Ridgway, "Automatic simulation model generation for supporting facility planning in smes," in *European Simulation and Modelling Conference, ESM 2013*, 2013, pp. 62–67.

[51] UN, *The 17 goals*, Accessed: May. 14, 2025., 2025. [Online]. Available: https://sdgs.un.org/goals.

A

Appendix 1 - Consent Form

The purpose of the interview is to gather non-numerical data from stakeholders who are considered relevant for the project. The goal is to assess what features and KPIs are necessary to develop a generic simulation object.

Your participation will contribute to building a better understanding of the currents state of how simulation is conducted today and to scope out the requirements needed to develop a generic simulation object.

Your participation in this interview is entirely voluntary. You may choose not to participate or withdraw at any time without any consequences.

All information shared during the interview will be treated with confidentiality. Your answers will be used in a wider collection of data and there will be no reference to you specifically in any published work.

This interview will be audio recorded for the purpose of accurate data collection. If you are uncomfortable with this, you may decline, and we can take notes instead.

By signing below, you acknowledge that you have read and understood the information provided above. You voluntarily agree to participate in this interview and understand your rights as participants.

Participants Name:

Participants Signature:

Date:

Interviewers Name:

Interviewers Signature:

Date:

B

Appendix 2 - Interview Questions

Background:

1. What is your current role and in which area/department are you working?

2. How many years experience do you have within your field (DES)?

Simulation:

1. What do you mainly use DES for today?

2. What do you look at when analyzing the data from the simulations?

3. In which abstraction level do you design and simulate your models?

4. How do you define transport and buffer objects?

5. Do you use custom objects in your simulation models?

Data collection:

1. What data do you use and how do you collect that data today?

2. Is it some kind of data that is usually missing?

3. Is it possible to find the missing data another way?

4. How does the data differs for different objects and processes? (if it does)

5. How would it be preferred to be able to collect the data?

C

Appendix 3 - Generic Simulation Tool

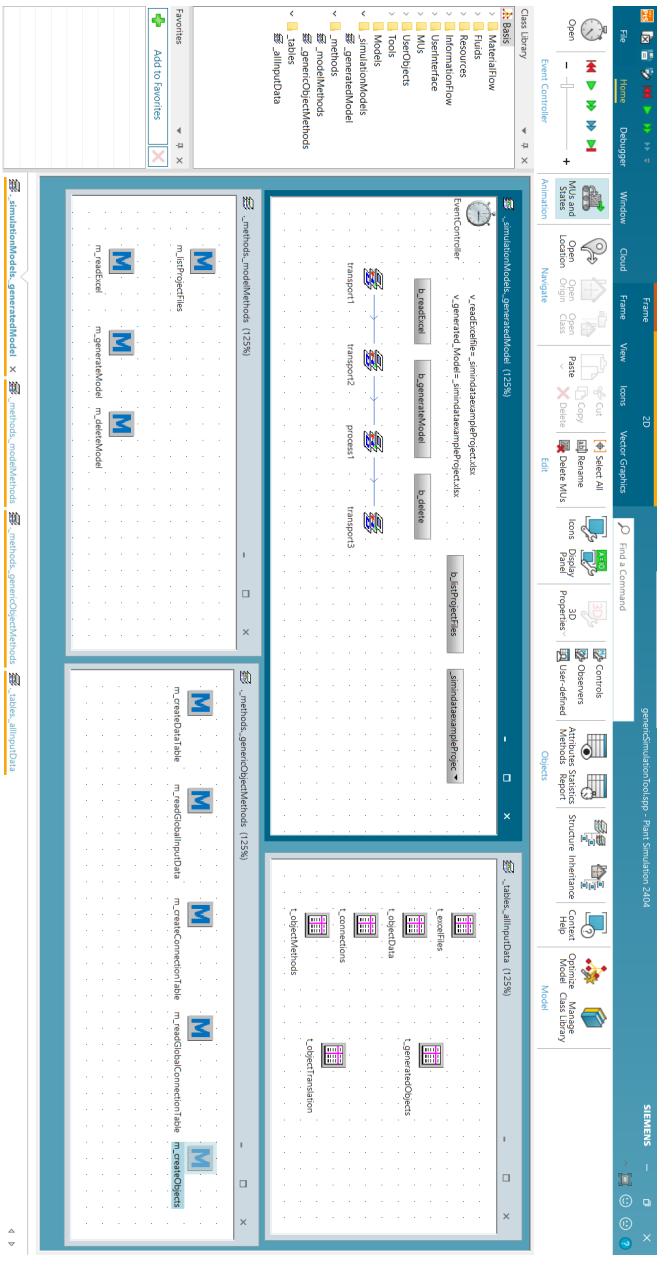


Figure C.1: Generic Simulation Tool

D

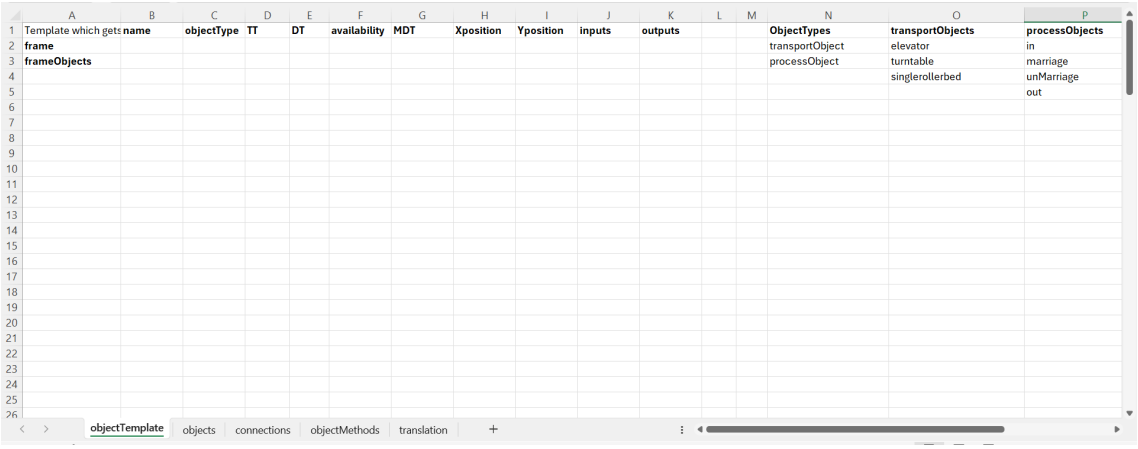
Appendix 4 - Input data table

Table D.1: Example of a main input data table

	name	objectType	TT	DT	availability	MDT	xPos	yPos	inputs	outputs
frame	transport1	transportObject					100	150	1	1
frameObjects	e1	elevator	10	10	96	2:30.00				
	s1	singlecollected	10	10	99	2:30.00				
	s2	singlecollected	10	10	99	2:30.00				
	t1	turntable	10	10	99	2:30.00				
	e2	elevator	10	10	97	2:30.00				

E

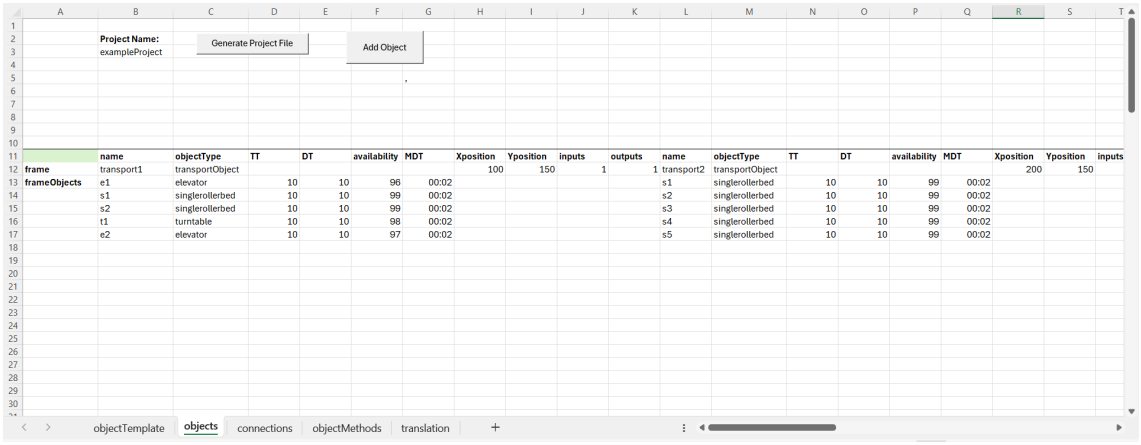
Appendix 5 - User-Interface Worksheets



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Template which gets	**name**	**objectType**	**TT**	**DT**	**availability**	**MDT**	**Xposition**	**Yposition**	**inputs**	**outputs**			**ObjectTypes**	**transportObjects**	**processObjects**
2	**frame**													transportObject	elevator	in
3	**frameObjects**													processObject	turntable	marriage
4															singlerollerbed	unMarriage
5																out

objectTemplate | objects | connections | objectMethods | translation

Figure E.1: User-Interface objectTemplate



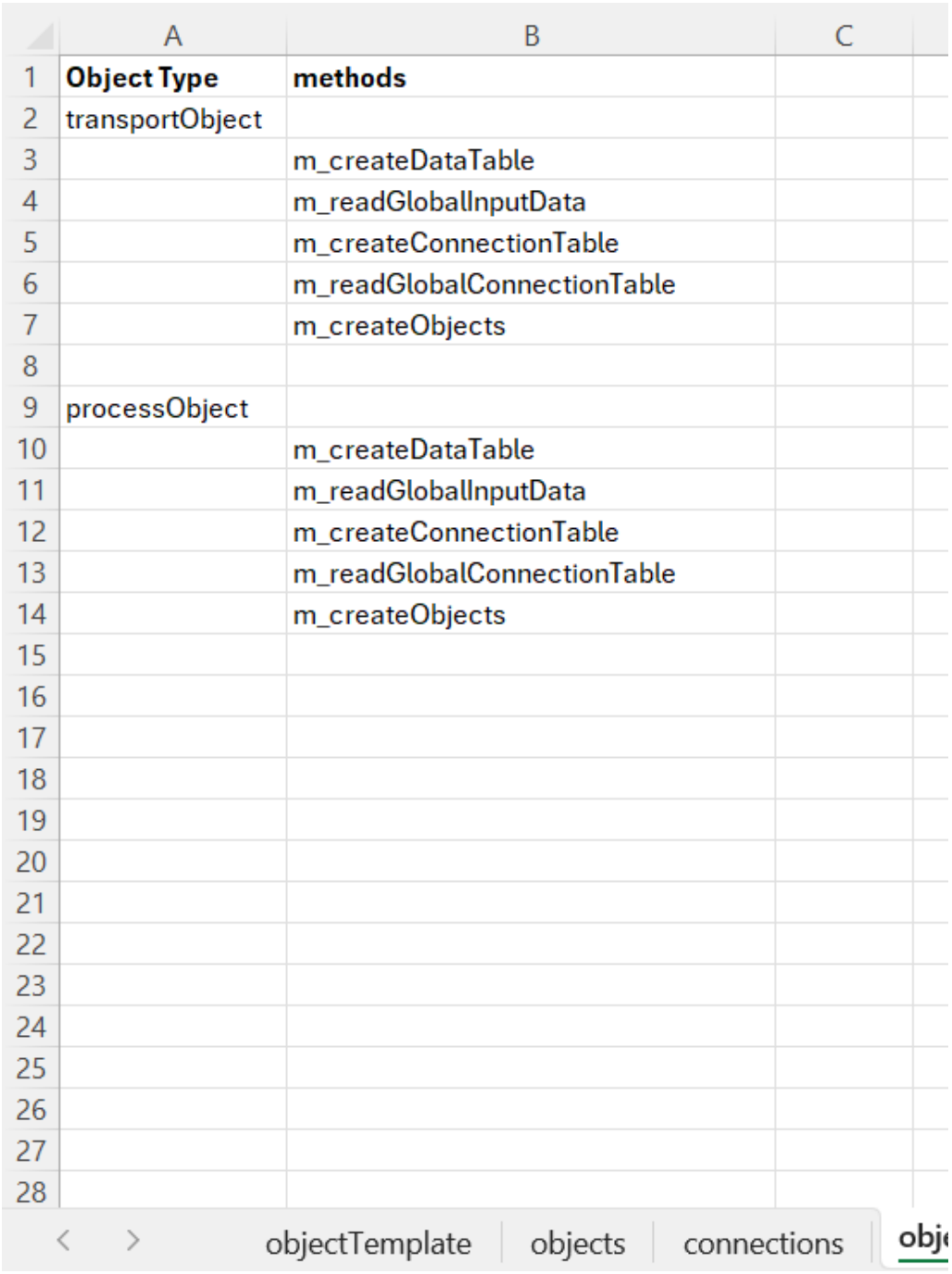
	name	objectType	TT	DT	availability	MDT	Xposition	Yposition	inputs	outputs	name	objectType	TT	DT	availability	MDT	Xposition	Yposition	inputs
frame	transport1	transportObject					100	150	1	1	transport2	transportObject					200	150	
frameObjects	e1	elevator	10	10	96	00:02					s1	singlerollerbed	10	10	99	00:02			
	s1	singlerollerbed	10	10	99	00:02					s2	singlerollerbed	10	10	99	00:02			
	s2	singlerollerbed	10	10	99	00:02					s3	singlerollerbed	10	10	99	00:02			
	t1	turntable	10	10	98	00:02					s4	singlerollerbed	10	10	99	00:02			
	e2	elevator	10	10	97	00:02					s5	singlerollerbed	10	10	99	00:02			

Figure E.2: User-Interface objects

	A	B	C	D	E
1	**FromObject**	**ToObject**	**FromFrame**	**ToFrame**	
2	interfaceIn1	e1	transport1	transport1	
3	e1	s1	transport1	transport1	
4	s1	s2	transport1	transport1	
5	s2	t1	transport1	transport1	
6	t1	e2	transport1	transport1	
7	e2	interfaceOut1	transport1	transport1	
8	interfaceOut1	interfaceIn1	transport1	transport2	
9	interfaceIn1	s1	transport2	transport2	
10	s1	s2	transport2	transport2	
11	s2	s3	transport2	transport2	
12	s3	s4	transport2	transport2	
13	s4	s5	transport2	transport2	
14	s5	interfaceOut1	transport2	transport2	
15	interfaceOut1	interfaceIn1	transport2	process1	
16	interfaceIn1	in1	process1	process1	
17	in1	svetsning	process1	process1	
18	svetsning	out1	process1	process1	
19	out1	interfaceOut1	process1	process1	
20	interfaceOut1	interfaceIn1	process1	transport3	
21	interfaceIn1	s1	transport3	transport3	
22	s1	s2	transport3	transport3	
23	s2	s3	transport3	transport3	
24	s3	e1	transport3	transport3	
25	e1	t1	transport3	transport3	
26	t1	interfaceOut1	transport3	transport3	
27					
28					

objectTemplate | objects | **connections** | objectMethods | translation

Figure E.3: User-Interface connections



	A	B	C
1	**Object Type**	**methods**	
2	transportObject		
3		m_createDataTable	
4		m_readGlobalInputData	
5		m_createConnectionTable	
6		m_readGlobalConnectionTable	
7		m_createObjects	
8			
9	processObject		
10		m_createDataTable	
11		m_readGlobalInputData	
12		m_createConnectionTable	
13		m_readGlobalConnectionTable	
14		m_createObjects	
15			
16			
17			
18			
19			
20			
21			
22			
23			
24			
25			
26			
27			
28			

Figure E.4: User-Interface objectMethods

	A	B	C	D
1	**objectTypes**	**standardObjects**		
2	elevator	station		
3	turntable	station		
4	singlerollerbed	station		
5	in	station		
6	marriage	assemblystation		
7	unMarriage	DismantleStation		
8	out	station		
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				
23				
24				
25				
26				
27				
28				

objectTemplate | objects | connecti

Figure E.5: User-Interface translation

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

CHALMERS
UNIVERSITY OF TECHNOLOGY