Tampereen ammattikorkeakoulu

Evaluation of SimMaker for Use in Wärtsilä Land and Sea Academy

Mikko Rajander

Thesis
February 2024

Degree Programme in Electrical Engineering

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Sähkö- ja automaatiotekniikan tutkinto-ohjelma

RAJANDER, MIKKO:
SimMakerin soveltuvuuden arviointi Wärtsilä Land and Sea Academyn käyttöön

Opinnäytetyö 51 sivua
Helmikuu 2024

Tässä opinnäytetyössä tutkittiin Simulator Maker -ohjelmapaketin (myöh. SimMaker) soveltuvuutta Wärtsilä Land and Sea Academyn käyttöön digitaalisten prosessisimulaattoreiden luomisessa. Opinnäytetyössä selvitettiin, millä tavoin SimMaker voisi parantaa WLSA:ssa käytettyjä kehittämistapoja. Tätä selvitystä varten luotiin esimerkkisimulaattori, jonka kehittämisestä saatua kokemusta verrattiin WLSA:ssa luotujen fyysisten prosessisimulaattoreiden kehityskokemuksiin. Esimerkkisimulaattorin kehittämisen aikana kerätyn kokemuksen lisäksi työssä hyödynnettin Wärtsilä Marine R&D -yksikön simulaattorikehitystiimin kokemuksia ohjelmapaketista. Tutkimuksen toimeksiantaja oli Wärtsilä Finland Oy.

Työn alussa esiteltiin työn tavoitteet sekä eteneminen. Lisäksi esiteltiin Wärtsilä yrityksenä, käsitellen yhtiön roolia teollisuudessa sekä merenkulussa. Wärtsilän lisäksi esiteltiin Wärtsilän koulutusosasto – WLSA. Esittelyn jälkeen siirryttiin kirjallisuuskatsaukseen, jossa käsiteltiin simulaation käsitettä, sekä tietokonesimulaation historiaa. Kirjallisuuskatsauksen lopuksi käsiteltiin simulaation roolia koulutuksessa. Tämän jälkeen esiteltiin SimMaker-ohjelmapaketti. SimMakeriä esiteltäessä kerrottiin sen käyttötarkoituksesta, ominaisuuksista sekä Wärtsilä Marine R&D -yksikön simulaattorikehitystiimin kokemuksista ohjelmapaketin kanssa. SimMakerin esittelyn jälkeen käsiteltiin opinnäytetyötä varten tehtyä esimerkkisimulaattoria sekä sen rakennusprosessia. Tämän jälkeen verrattiin WLSA:n kalustoperäisten prosessisimulaattoreiden kehitysprosessia SimMakerillä tehtyjen digitaalisten prosessisimulaattoreiden kehitysprosessiin. Tämän vertailun perusteella tutkittiin, mitä hyötyjä SimMakerin käyttö toisi WLSA:lle.

Vertailun tulokset osoittavat, että SimMaker tarjoaa parannuksia sekä kustannusten että simulaattorikehityksen tehokkuuden kannalta. Esimerkiksi digitaalisia simulaattoreita voidaan kehittää oppimistavoitteiden mukaan. Lisäksi digitaalisissa simulaattoreissa ei tarvitse käyttää fyysisiä komponentteja, jolloin kalustokustannukset vähenevät merkittävästi. Lisäksi SimMaker-ohjelmapaketin sisäiset standardisoidut työkalut tehostavat kehitys- ja huoltoprosesseja. Jatkokehityksen kannalta on merkittävää, että WLSA:n kouluttajat eivät itse ole vielä käyttäneet SimMaker-ohjelmapakettia. Soveltuvuuden selvittämiseksi suositellaan jatkokehitykseksi tuoda ohjelma koekäyttöön ja hankkia kouluttajien mielipiteitä ohjelmapaketin soveltuvuudesta.

Asiasanat: simulaatio, prosessiautomaatio, koulutus

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Electrical Engineering

RAJANDER, MIKKO:
Evaluation of SimMaker for Use in Wärtsilä Land and Sea Academy

Bachelor's thesis 51 pages
February 2024

The purpose of this thesis was to evaluate the efficacy of adopting the Simulator Maker software package (later referred to as SimMaker) for use at Wärtsilä Land and Sea Academy (WLSA). The software package is used to create digital process simulators that could be used to supplement or replace existing, hardware-based process simulators. To achieve the thesis objective, a comparison was made between the development methods used in building physical process simulators in WLSA and those used when building digital process simulators using SimMaker. To understand the development methods used with SimMaker, a demo simulator was built. This development experience was also supplemented with SimMaker user experiences from the Wärtsilä Marine R&D team. The combined information was then compared to the development methods used at WLSA.

The findings of this thesis show that SimMaker offers many advantages to WLSA in terms of effectiveness in simulator development and cost efficiency. The development process is enhanced by advantages like learning objective-based development and standardised tools, while the lack of physical components reduces equipment costs. Additionally, the ability to share digital content through cloud make the simulators accessible from several locations, cutting logistical expenses.

Whilst the study shows clear advantages in using SimMaker, further studies should be conducted with training instructors in WLSA. This way the suitability of the software package can be further evaluated according to the needs of the end users.

Key words: simulator, training, process, PID

SISÄLLYS

ABBREVIATIONS AND TERMS

WLSA	Wärtsilä Land and Sea Academy
SimMaker	Simulator Maker software package
TechSim	Technological Simulator training platform
FMU	Functional Mock-up Unit
FMI	Functional Mockup interface
MVS	MV Studium
MCE	Model Configurator editor
DE	Datainfo Editor
MD	Model Desk
PE	Panel Editor and Controls
UI	User Interface
R&D	Research and development
PID	Porportional-Integral-Derivative Controller
LNG	Liquified Natural Gas
Man-week	Work completed by one person in a 40-hour week
MethanolPac	Methanol engine fuel supply system created by Wärtsilä
Fidelity	Measurement of simulation accuracy compared to real-world systems or scenarios.
Abstraction layer	Method of organizing a system into separately manageable parts to be developed independently without affecting each other.
Fidelity	Measurement of simulation accuracy compared to real-world systems or scenarios.

1 INTRODUCTION

The objective of this thesis is to evaluate the practicality of adopting the Simulator Maker software package (later referred to as SimMaker) for training simulator development purposes in Wärtsilä Land and Sea Academy (WLSA). This thesis work will include an introduction to SimMaker as well as instructions the installation and use of SimMaker. This thesis will also discuss the basics of simulation as well as the use of simulators for training purposes.

In the maritime and energy industries, training plays a vital role in ensuring safety and efficiency during the operation of machinery and industrial processes. To provide sufficient training for workers, simulators are used to provide a safe and controlled environment to train in. Simulators provide trainees with consistent and repeatable training scenarios to help improve their skills. In WLSA, these process simulators provide high fidelity, but their physical nature limits them logistically. Trainees need to travel to specific training centers in order to access the simulators. SimMaker is a simulation creation software package that enables the creation of fully digital process simulators that can be shared across the world via cloud sharing, making them easily accessible from multiple locations.

In this thesis, I compare the development process of digital process simulators created using SimMaker to those currently used to build hardware-based process simulators in WLSA. Through this comparison, I address the question “How could SimMaker benefit WLSA?” In addition to the comparison, I have gathered literature on the nature of simulators and their use in training, examining their advantages and disadvantages. As the thesis focuses on the development of training simulators, the scope of this thesis is focused on the development process of the simulator. The thesis does not include analyzing the training effectiveness of simulators made with SimMaker, as WLSA currently does not use SimMaker to build training simulators.

In chapter 2 of this thesis, I will begin with introducing Wärtsilä as a company, examining its history, strategy and portfolio. Chapter 3 is the literature review, where I discuss the nature and history of simulators as well as their use in training. Chapter 4 will introduce SimMaker and the training platform that utilizes simulators made with SimMaker, TechSim. Chapter 5 will showcase the creation process of a demo simulator built with SimMaker. Chapter 6 will compare the development processes of current day hardware-based process simulators in WLSA to the digital simulators developed with SimMaker. Chapter 7 concludes the thesis by examining the findings, discussing their implications, and providing recommendations for the adoption of SimMaker in WLSA and suggestions for future research.

2 COMPANY INTRODUCTION

Wärtsilä Abp Oyj is a global technology company that focuses on creating lifecycle solutions for energy and marine industries. Wärtsilä has decarbonization and sustainability as its core values. The company strives to continuously create greener future technologies in the effort to create a decarbonized future in the marine and industrial fields. (Wärtsilä Official Website n.d.)

2.1 History

Wärtsilä was founded in 1834 in the former Finnish municipality of Tohmajärvi. The company's history began as a sawmill and iron works and was formally named Wärtsillä Ab in 1898. The company expanded into the marine markets in 1935 when it acquired Hietalahti shipyard in Helsinki and Crichton-Vulcan shipyard in Turku. Soon after this change Wärtsilä moved its headquarters to Helsinki. Wärtsilä started manufacturing its first diesel engines under license in 1942 and started its own designs in 1954. Modern day Wärtsilä has expanded considerably from its origins as a sawmill. Wärtsilä Abp Oyj is an international technology company focused on creating lifecycle solutions for the marine and energy markets. Wärtsilä is a leader in green energy solutions and power storage technology. (History n.d.)

2.2 Company strategy

Wärtsilä focuses heavily on decarbonization in the marine and energy industries. To achieve this, Wärtsilä's strategy centers around two key themes: Transform and Perform. Wärtsilä aims to transform energy production in the marine and energy fields with greener solutions. This decarbonization is driven by Wärtsilä's advancements in technology and alternative fuels. Wärtsilä's advancements in technology strive to cut emissions in the energy industry by optimizing power generation with energy generation and storage solutions. In the marine industry, decarbonization is seen in advancements made in electric drivetrains and carbon capture solutions among others. (Strategy & Purpose n.d.) Alternative fuels, such as methanol, are good alternatives for diesel and can be used to gradually lessen emissions created by engines. Green methanol, or methanol made from either biomass or captured carbon, along with green hydrogen have the potential to be carbon neutral. (Wärtsilä 2023)

The second key theme of Wärtsilä’s strategy is Perform. This aims to make use of recovering markets and growth. Wärtsilä’s growth is helped by new partnerships and acquisitions of Wärtsilä products. Service growth also helps in the growth of Wärtsilä as service support is very profitable. Wärtsilä’s decarbonization efforts also provide profit opportunities from green retrofit solutions and conversions. (Strategy & Purpose n.d.)

2.3 Wärtsilä Land and Sea Academy

WLSA or Wärtsilä Land and Sea Academy is Wärtsilä’s Global training network that offers a wide range of training programs for its customers in the energy and marine industries. The courses offered by WLSA include exclusive product training for Wärtsilä products and for everything to do with the management and operation of customer installations. WLSA training centers are located across the globe and offer many types of training methods. Training can be completed fully online via e-learning courses, face-to-face in classroom teaching and practical training. WLSA also makes use of simulators to train professionals using real equipment and components in simulated environments. (WLSA n.d)

2.4 Wärtsilä Marine and Energy

Wärtsilä Marine focuses on creating efficient, reliable and environmentally friendly solutions for marine power and propulsion systems. The company’s maritime expertise includes extensive experience in engine manufacturing as well as advanced hybrid fuel technology and digital technology. Wärtsilä Energy focuses on providing greener solutions for the energy market. These solutions include future fuels as well as power storage. Wärtsilä Energy uses the company’s extensive knowledge in engine manufacturing to create greener and more efficient engine powerplants. Wärtsilä Energy also creates power storage solutions that allow customers to store power reserves created with renewable energy or with the help of engine powerplants. (Wärtsilä Official Website n.d.)

3 SIMULATION – LITERATURE REVIEW

Simulation is a technique used to model real life processes and systems numerically in a test environment. This is often done using a computer program. (Simulation Modelling and Analysis 2015, 1) Simulation differs from other modelling methods in its nature of operation. Simulation is a time-based modelling method that responds as the real system would under certain conditions. Other methods of modelling (such as statistical models) might only model a system at a specific point in time. Mathematical models might only use deterministic formulas to represent modelled systems without the temporal or random aspects used in simulation.

3.1 History of computer simulation throughout the decades

The first computer simulators were used in the 1940s when the first programmable computers were created. These first computers, such as the ENIAC (Electronic Numerical Integrator and Computer), were initially designed for military purposes during the Second World War. One such military use of computer simulation was at the Los Alamos National Laboratory as part of The United States' nuclear program, the Manhattan Project. In the Manhattan project, scientists used the Monte Carlo simulation method to study the behavior of neutrons. (University of Houston 2000) The Monte Carlo simulation method uses random sampling to simulate different outcomes in systems. The method uses a predefined range of numbers in a random order to simulate a large range of scenarios. (What is Monte Carlo Simulation? n.d.) The Monte Carlo method can be applied to various purposes in various fields. In engineering, this method can be used to test the reliability of a system.

In the 1950s, the use of early analog and digital computers marked a significant evolution in simulation. Analog computers were employed to solve differential equations in fields like aerospace, while digital computers, though still limited in power, began to handle complex calculations. The decade was characterized by experimentation and the initial application of these technologies in industry, despite challenges like slow processing and ambiguous results. (Roberts and Pegden 2017)

The 1960s saw the development of specialized simulation tools and languages, most notably the General Purpose Simulation System (GPSS) developed by IBM. These advancements made simulation more accessible and practical, especially in industries such as manufacturing and aerospace. The establishment of conferences and workshops focused on simulation helped standardize practices and fostered collaboration among professionals, further advancing the field. (Roberts and Pegden 2017)

The 1970s were marked by rapid expansion and the increasing standardization of simulation practices. As computing power grew, simulations became more complex and widely used across various sectors, including manufacturing, transportation, and telecommunications. The decade also saw the formalization of simulation in industrial engineering and the creation of user-friendly simulation languages like SLAM and SIMSCRIPT. (Roberts and Pegden 2017)

The 1980s brought simulation into the mainstream, driven by advances in personal computing and the development of more accessible software such as SLAM II, SIMAN, and CINEMA. Simulation became a critical tool across industries, used not only for design and analysis but also for strategic decision making. This decade marked the transition of simulation from a specialized tool to a widely adopted technology. (Roberts and Pegden 2017)

In the 1990s, simulation became increasingly integrated with other technologies, such as enterprise resource planning (ERP) systems, and saw significant advancements in visualization. Tools like GPSS/PC, EXTEND, and Micro Saint offered features like real-time data collection, 3D animation, and optimization, making simulation more powerful and versatile. These developments broadened the appeal and application of simulation, enabling it to become a fundamental part of modern industry and science. (Roberts and Pegden 2017)

3.2 Simulation in modern time

In the modern era, computer simulations are integral to a wide range of scientific fields like natural sciences or social sciences etc. The development has been marked by a diverse and fragmented evolution, leading to specialized methods tailored to the needs of different disciplines. In finance, simulation is used in activities such as risk assessment and algorithmic trading. In risk assessment, the Monte Carlo method previously mentioned in the Manhattan project can be used to assess probable outcomes of different business strategies. This is a good example of how different techniques used in the past are still very useful to this day. In the aerospace industry, simulation is used in flight simulation, mission planning and training.

3.3 Simulation training

The use of simulation in training has become more widespread as technology has improved. Simulators are very useful tools in training in the modern day because they offer a safe environment for trainees to practice their job roles in. Simulators provide trainees with a controlled environment to practice essential skills in without the risks that come with mistakes made in the field. Simulators also provide trainees with reliable feedback on their performance, allowing them to focus on specific areas of improvement throughout training. (Kim et al. 2021)

3.3.1 Advantages of simulator training

Bouhelal, Patel H. and Patel B. highlight that exposure and repetition are what determines skill acquisition and efficiency in training the most (Bouhelal, Patel H. & Patel B. 2012, 43). This is one of simulations strong suites. Simulators offer trainees consistent and repeatable training scenarios, where they can make use of repetition to better learn required skills. Simulator training also enables trainees to train for scenarios not often encountered in the field. This is very helpful in preparing trainees for rare but important events such as emergency procedures. Bouhelal et al. (2012) also state that properly trained individuals directly lead to marked decrease in costs resulting from errors (Bouhelal et al. 2012, 42-43).

As well as offering trainees an opportunity to acquire experience in even rare events, simulators can be beneficial in a logistical sense. Physical simulators (e.g. simulators with actual hardware) are still limited by their location. Ship bridge simulators for example often comprise of a lot of hardware that cannot be relocated easily if at all. Virtual simulators (e.g. simulators that run completely virtually without hardware other than a computer) on the other hand, can be relocated simply by installing it in a different location. Virtual simulators also save costs compared to physical simulators by not incurring the cost of purchasing the physical components required by physical simulators.

3.3.2 Disadvantages of simulation training

Simulation also has its downsides when comparing it to practical training. According to Krishnan D., Vasu Keloth A. and Ubedullah S., building simulators with sufficient fidelity can add considerable costs in the development phase. They also state that insufficient fidelity in a medical training simulator can lead to malpractice in actual cases. Students may for example neglect checking certain physical signs in patients if these are not modelled into the simulator. (Krishnan D., Vasu Keloth A. & Ubedullah S. 2017, 85-86) In engineering, this could translate into executing wrong procedures in process automation because certain parts of the automation process were not modelled into the training simulator.

4 SIMMAKER

To create digital simulators, Wärtsilä acquired Transas, a UK based global company in May of 2018. Along with Transas, Wärtsilä acquired simulation tools created by Transas — the TechSim platform and SimMaker. In a press release (Wärtsilä 2018), Wärtsilä describes Transas as a “global market leader in marine navigation solutions that include complete bridge systems, digital products and electronic charts. It is also a leader in professional training and simulation services, as well as ship traffic control systems”. After the transition to Wärtsilä, today the “Transas team” is known as the Wärtsilä Marine business R&D simulation team.

Simulator Maker or SimMaker is a simulation software package used to create simulators for the Technological Simulator training platform (TechSim). The SimMaker software package can be used to create simulation controls and panels as well as mathematical models for simulations and their connections to the panels and controls.

The SimMaker software package includes the following programs:
MV Studium (MVS)
Model Configuration Editor (MCE)
Datainfo Editor (DE)
ModelDesk (MD)
Panel Editor and Controls (PE)
Simulator Maker (SM)

MV Studium (MVS) is mathematical modelling program used to build mathematical models of various simulated equipment, processes and systems. It allows users to define equations and relationships that represent the behavior of real systems. MVS uses the modelica modelling language. Modelica is an object-oriented modelling language that is relatively easy to grasp. It uses visual objects as well as text to create mathematical models. Model Configuration Editor (MCE) is used to configure or edit parameters of the simulator model. This can be thought of as tuning the simulators behavior. Datainfo Editor (DE) is used to create, set and edit network variables. These variables are essential for the data exchange within the simulation environment, allowing different components to communicate and interact. ModelDesk (MD) is used to create and link network variables to control panels. It is integrated with Panel Editor and Controls (PE), Datainfo Editor and Model Configuration Editor, ensuring that the creation and management of network variables are consistent across these different programs. Panel Editor and Controls (PE) is used to create control panels for the simulator using different control element libraries. These libraries include visual elements such as controls, indicators, pumps etc. Simulator Maker (SM) is used to create and edit simulator folders. SM brings together the different parts of the simulator built with the programs of the SimMaker software package. It allows developers to add menus, control panels and mimics used in simulators.

One of the big advantages of SimMaker is that it enables developers to use libraries of reusable components in new projects to decrease development costs. This enables the use of components in older projects to be reused in new projects. A key benefit of this is that the more tool is used, the bigger library grows, further increasing efficiency of the development process. As SimMaker is not yet in heavy use, the library is nonexistent at the moment.

Another advantage of the SimMaker software package is its use of abstraction layers to enhance project workflow. In the context of SimMaker, the abstraction layers are the model level, UI level and data exchange level. In the model level, MVS (or other FMU supporting programs) are used to develop the mathematical level. In the UI level, Panel Editor and controls is used to develop the control panels, or UI. Finally, in the data exchange layer, Datainfo Editor is used to manage the data exchange between the mathematical model and the UI. This abstraction allows the different levels to be developed simultaneously, improving the efficiency of the project.

4.1 TechSim platform

Technological Simulator or TechSim is a simulator training platform developed by Wärtsilä and is used by WLSA to provide a simulator training varying in resolution from simple virtual panel simulators to detailed physical simulators that use a combination of hardware and software. TechSim allows instructors to teach trainees efficiently with its built-in suite of training software components. This means that simulators themselves do not need to accommodate simulator training, as the platform itself has a ready infrastructure for this purpose.

TechSim has dedicated software components for instructors and trainees. The instructor component can be used by the instructor to prepare exercises in editor mode and allows playback of the trainee's exercise performance in debriefing mode. The trainee component allows trainees to complete exercises provided by the instructor. TechSim software can be used in a single computer or in a network of computers. Using TechSim in a network of up to 50 computers is useful in a class setting. The instructor station sets up exercises and records the performance of the trainee workstations.

4.2 Simulator development

The digital simulators created by the simulation team at Wärtsilä Marine R&D using SimMaker range in scale. Smaller projects include the liquified natural gas (LNG) bunkering simulator and the MethanolPac simulator. The LNG bunkering simulator is meant to train workers on the process of bunkering. Bunkering is a term used in the marine industry to convey the supplying of fuel onto a ship for use in the ships machinery — in essence, refueling of a ship (Anish Wankhede 2019). MethanolPac is Wärtsiläs methanol fuel supply system for Wärtsilä engines designed to use methanol as fuel (MethanolPac n.d). The largest project built by the simulation team was the entirety of a cruise vessel. In an interview with Lasse Nikkanen, the product lead in engine and cargo simulators at Wärtsilä (Interview 2024), he estimated that simulator projects created by the simulation team can last from around 50 man-weeks to 600 man-weeks. Small projects can last around 50 – 100 man-weeks, mid-sized projects around 100 – 300 man-weeks and large projects around 300 – 600 man-weeks.

4.3 SimMaker software interaction

The SimMaker software package consists of a suite of different programs that are designed to work together to build simulators. The process of building a simulator with SimMaker starts with building the mathematical model of the simulator using MV Studium or other FMU compatible mathematical modeling tools. FMUs use the Functional Mock-up Interface (FMI) standard to enable the exchange of dynamic models between different simulation tools (The FMI Standard 2023). Effectively this means that SimMaker can accept mathematical models made with other FMU supported modelling tools. The creation of the mathematical model is a logical starting point for building a simulator because it defines the functioning and logic of the simulator. It is important to note that when using FMUs to build simulators in SimMaker, the end product uses only the FMU included with the simulator file. If there is something wrong with the FMU, changes need to be made to the original mathematical model that the FMU created from. Therefore, it is important that users store the mathematical model to be accessed later if needed.

Thanks to the abstraction layers used in SimMaker, the control panel can be developed simultaneously with the mathematical model using Panel Editor and controls. The control panel and the mathematical model created are then connected to each other using Model Desk (MD), which connects the mathematical model to the controls via network variables. These network variables allow for data transfer between the mathematical model and the controls. Effectively this means that when the simulator is in use, the changes made in the control panel affects the simulation accordingly. After the creation of the network variables in MD, they can be edited and rearranged using DataInfo Editor (DE). Network variables can also be created in this program. DE collects all the network variables used in the simulation into data info files that can then be edited. MD and DE are integrated with each other so that changes made in either program is present in the other.

Further tuning to the parameters of the simulator can be done using Model Configuration Editor (MCE). MCE plays an important role in determining the accuracy or fidelity of the simulator. Changes in parameters such calculation step can provide more accurate results in the simulator.

All the afore-mentioned programs (MVS, PE, MD, DE and MCE) are tied together using Simulator Maker (SM). SM allows the compiling of all the work done in the previous programs to be compiled together to produce a simulator. SM allows the user to create and edit the structure of simulator folders that organize the files and resources used during the development process. After finalizing the simulator with SM, it is compiled into a cabinet file. A cabinet file or a CAB file is a compressed archive format that is used to store multiple files. The produced CAB file can then be added to the TechSim simulator environment and is ready to be used.

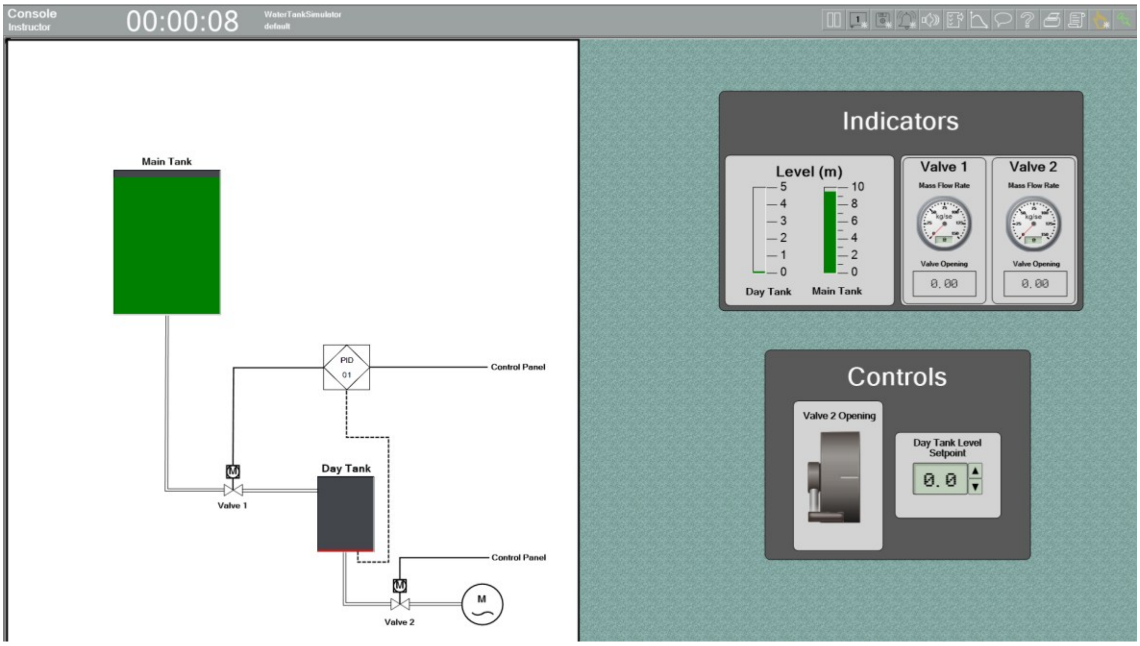
4.4 Installing and launching SimMaker

The installation of SimMaker software tools require the user to have Local administrator rights to their PC. First, the user must log in as local administrator and open the Technology simulator installation package. Then the Setup.exe must be run as administrator. The installation wizard will guide the user in the installation process. The user must check the “Install Simulator Desktop” box to allow the install of SimMaker. If the TechSim software has already been installed on the user’s device, SimMaker will be installed in a subfolder within the TechSim folder. If not, the wizard will prompt the user to create an installation folder for SimMaker. After the installation is complete, a shortcut should be created in the desktop.

To launch SimMaker, the user must right click on the desktop shortcut and select “Run as administrator”. All SimMake components must be launched as administrator.

5 SIMULATION EXAMPLE

To demonstrate the use of SimMaker, an example simulator was made using the software package. The simulator is meant to demonstrate the functions of the different components of the SimMaker software package. The simulation model is a water tank system comprising of two tanks: the main tank and the day tank. The names for these tanks come from ship fuel tanks. The water tank system is modelled after ship fuel systems, where the main tank of the ship feeds fuel into the day tank. The day tank then feeds fuel into the engine of the ship.



PICTURE 1. Water tank simulator. Day tank almost empty at the start of simulation.

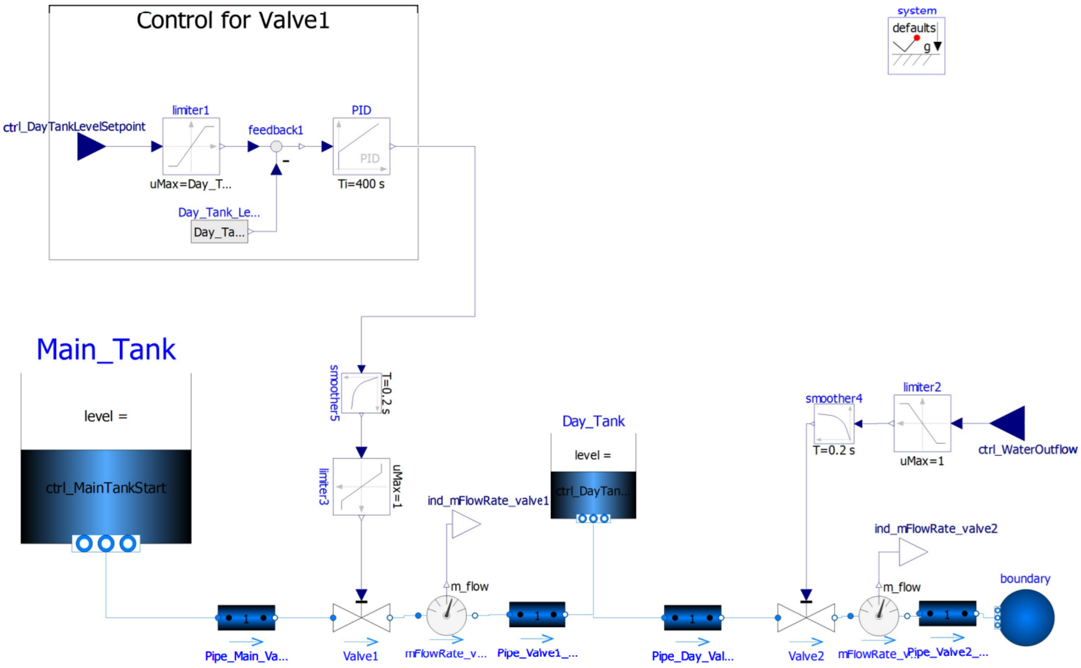
In the demo simulator, the main tank is filled with water while the day tank is nearly empty, as shown in picture 1 with the almost empty red level bar. When the simulation starts, the main tank starts feeding the day tank with water via pipes. The flow of water from the main tank to the day tank is controlled by a valve. The valve (Valve 1) is controlled with a PID controller. This controller is given a setpoint (Day Tank Level Setpoint) that tells the controller how high the level of the day tank should be. The PID controller measures the level of the day tank and opens Valve 1 accordingly. The day tank connects to the engine via pipes, with the flow of the water being controlled by another valve (Valve 2). The engine in the demo is only for visual representation. It has no controls in the simulation. In addition, Valve 2 would be an ON/OFF valve in an actual system. The valve would be meant to cut off the flow of fuel to the engine. To simplify the simulator, the opening of Valve 2 represents the fuel consumption of the engine. The more Valve 2 is opened, the more fuel is consumed. Users can open Valve 2 as desired, and Valve 1 will adjust the level of the day tank accordingly. Because the simulator uses water as its medium, the word “fuel” is not totally accurate. Fuel is used as a term only to relate the simulator to the fuel systems it is modelled after.

To create the simulator, I first started with the creation of the mathematical model as it defines how the simulator operates and is the base that the entire simulator is built on. After completing the mathematical model, I built the control panel or UI that is used to control and visualize the simulator. The control panel and mathematical model were then bound to each other, so that inputs made in the control panel affect the mathematical model. After binding the two together, I compiled the simulator into a cabinet file that could then be used in the TechSim platform as a simulator. The steps for building the simulator are further expanded upon in the following subchapters.

5.1 Building the mathematical model

5.1.1 Mathematical model overview

The mathematical model of a simulator is the bedrock of the simulator and thus, is a good place to start the building process. MV Studium, the integrated mathematical modelling software in SimMaker, would ideally be used in the creation of the mathematical model. In my project, I decided to use wolfram system modeler to create my mathematical model. This was due to issues with the use of MVS. The two programs use the same modelica visual modeling language, making the development experience somewhat comparable. The process of making the mathematical model using modelica was intuitive due to the visual aspect of the modeling language.



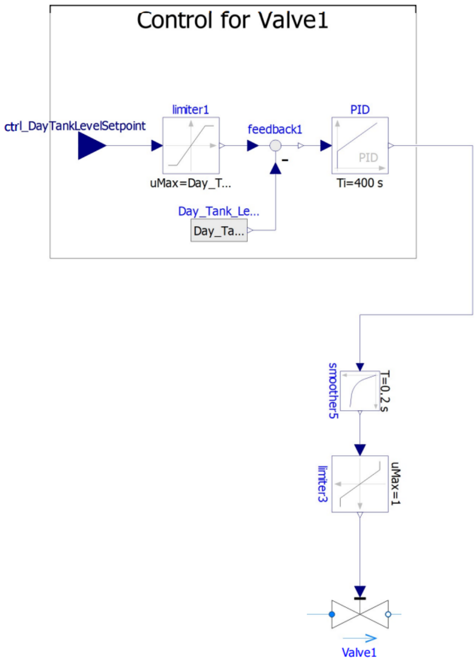
PICTURE 2. Mathematical model for water tank simulator.

As seen in picture 2, the overall visual design of the model is comparable to a piping and instrumentation diagram (P&ID). In the mathematical model, a similar structure to the visual representation in the control panel (picture 1) can be seen. The controls for the mathematical model will be expanded upon in later subchapters. In this subchapter, I will go over the main features of the rest of the system.

The main tank placed on the far-left side of the mathematical model is linked to a pipe component with a solid line. Connecting components like this enables them to interact with each other. From the first pipe component, a link is established with valve 1, and from there to another pipe via a mass flowrate meter. In practice, this means that the pipeline coming from the main tank is fitted with a valve (valve 1) and a mass flowrate meter. A mass flowrate meter measures how much fluid flows through the pipe per second. This information is then forwarded from the meter to an output that is later used in the control panel. The pipeline connects to the day tank. The day tank then connects to another pipeline that is fitted with valve 2 and another mass flowrate meter that also sends its output to the control panel. This pipeline then connects to a boundary element. This boundary element can be thought of as a void or a vessel of infinite capacity. The boundary element represents the ship engine in the final model.

5.1.2 Valve controls

The controls of the simulator are also created in the mathematical model and are connected to valve 1 and valve 2. The PID controlling valve 1 receives a setpoint value for the level height of the day tank from the user. This signal is represented with a solid blue arrow on the left side of the area marked as “Control for Valve1” as seen in picture 3.

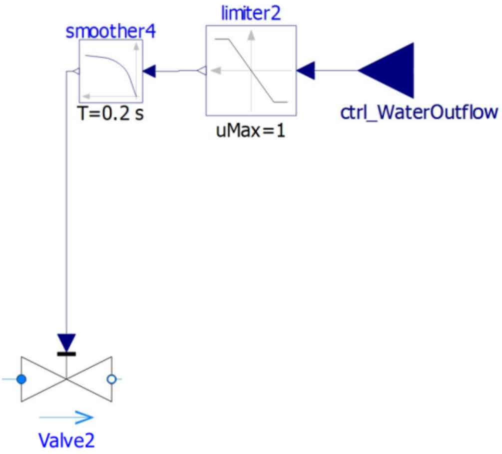


PICTURE 3. Control circuit for valve 1 in mathematical model.

The signal given from the input is then passed through a limiter. The limiter sets the maximum level height to be as tall as the height of the day tank. This prevents overflow. The signal then passes through a feedback point to the PID controller that controls valve 1. In the feedback point, the level measurement of the day tank is subtracted from the input signal (represented with the arrow with a minus symbol next to it). The result of this subtraction is the error of the controller. In practice, this feedback loop compares the wanted level height to the actual level height by subtracting the current level from the desired level. As the level in the day tank rises, the error in the controller gets smaller until the desired level is reached. At this point the controller shuts off valve 1 so the level does not rise above the wanted level. As the PID controller is the most essential part of the control system for valve 1, it will be further examined in chapter 5.1.3.

Before the signal reaches valve 1, it goes through a first order block (smoother5) and a limiter. The first order block is named a smoother because it is configured to act as a low-pass filter to the control signal given by the PID controller. In the mathematical model, the low-pass filter acts to slow the opening and closing of the valve. This is because a valve in an actual system cannot open and close instantly. This also enables further changes to be made in the opening characteristics of the valve (e.g. the opening can be further slowed down or sped up to simulate different opening times of different valves). The final part of the control loop for valve 1 is a limiter. This limiter limits the signal to real number values between 0 and 1. This is because the opening of valve 1 is measured in percentage from 0% to 100%. 0 is 0%, 0.5 is 50% and 1 is 100%.

In comparison to the first valve, valve 2 has simple controls. Valve 2 is designed to be fully controlled by the user as it acts as a “throttle lever” in the final model.

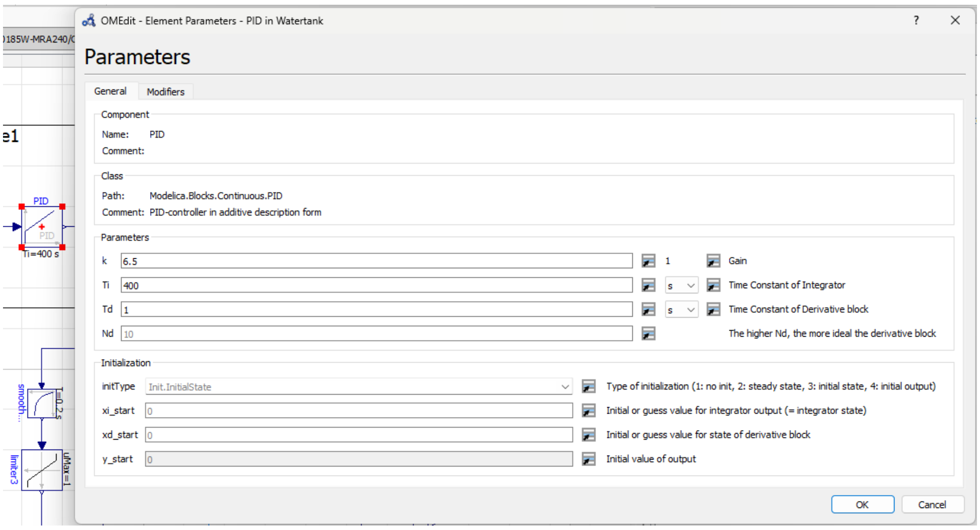


PICTURE 4. Controls for valve 2.

As seen in picture 4, valve 2 receives an input signal like valve 1. The user changes the value of the input in the control panel, which is then sent to valve 2. The control signal of valve 2 has the same limiter and first order block as valve 1. The limiter works the same way that the limiter in valve 1 controls just before reaching the valve. It limits the control signal to real values between 0 and 1. The first order block or “smoother” also serves the same purpose as the “smoother” block in valve 1.

5.1.3 PID controller

The water tank system is controlled and managed by a proportional-integral-derivative controller or PID controller seen in picture 3. The PID controller receives a setpoint signal for the desired level in the day tank from the user and operates valve 1 to increase the level of the tank when required.



OMEdit - Element Parameters - PID in Watertank

Parameters

Component
Name: PID
Comment:

Class
Path: Modelica.Blocks.Continuous.PID
Comment: PID-controller in additive description form

Parameters
k 6.5 1 Gain
Ti 400 s Time Constant of Integrator
Td 1 s Time Constant of Derivative block
Nd 10 The higher Nd, the more ideal the derivative block

Initialization
initType Init.InitialState Type of initialization (1: no init, 2: steady state, 3: initial state, 4: initial output)
xi_start 0 Initial or guess value for integrator output (= integrator state)
xd_start 0 Initial or guess value for state of derivative block
y_start 0 Initial value of output

PICTURE 5. PID controller properties

The tuning of the PID controller was done by trial and error. Better ways of tuning exist but for the purposes of this demo simulation, approximate values were acceptable. The parameters given to the PID can be seen in picture 5. These values will next be examined in detail with the help of figures 1 and 2. To illustrate the function of the PID controller, I simulated a scenario where the setpoint of the day tank was first increased from 0 to 2. After this I opened valve 2 halfway (opening 0.5). In response to the opening of valve 2, the system compensates by opening valve 1 to keep the level of the day tank steady.

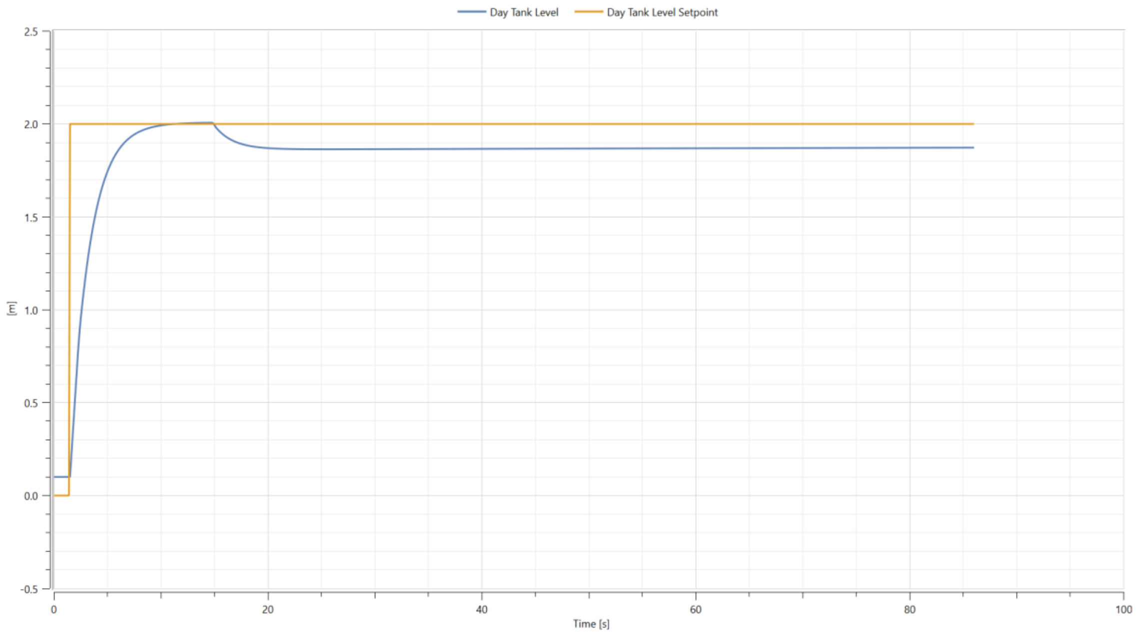


FIGURE 1. Water tank system plot window 1. Day tank level and level setpoint. Setpoint increased at 1.5s mark, valve 2 opened at 15s mark.

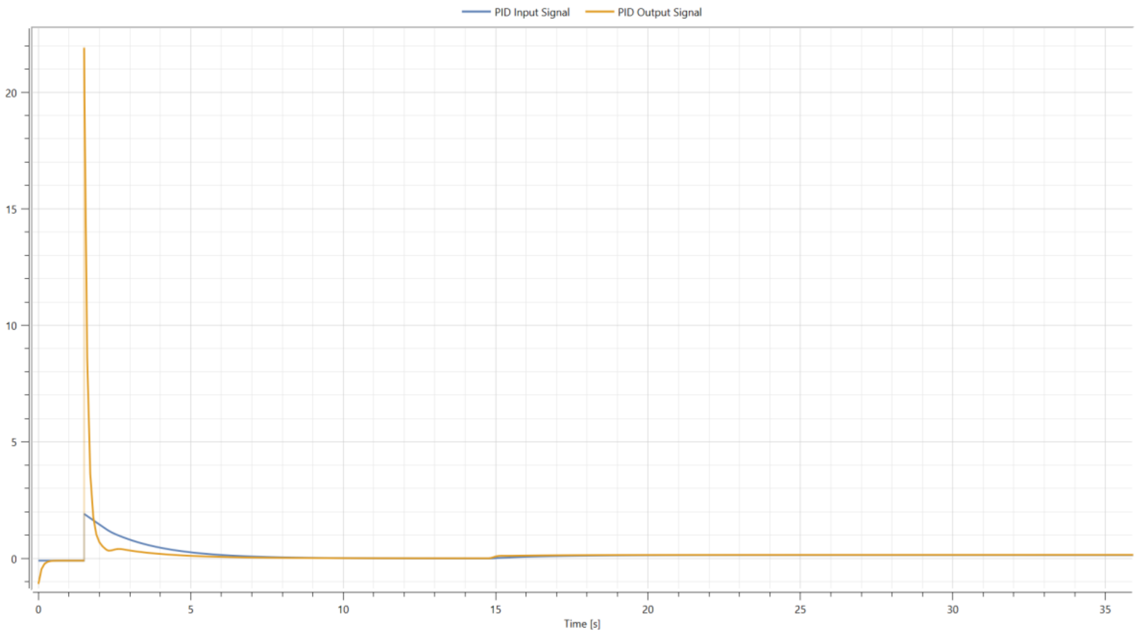


FIGURE 2. Water tank system plot window 2. PID input (Process variable) and output (Controller output). PID input signal is the error of the system.

Picture 5 shows the parameters of the PID controller. The first parameter is the gain value. The gain value affects the proportional action of the PID controller. The proportional action of the PID works directly proportionally to the process variable error. The larger the process variable error is, the stronger the controller output. The proportional gain value dictates the strength of this output. The value for gain in the water tank system is 6.5. This means that the controller has a strong reaction to errors in the process variable. This makes the controller fast acting but increases the chance of overshoot. This will be compensated for in the other control actions. In figure 2, the combination of a strong gain parameter and the derivative action, examined later, combine to create a big spike in the controller output in reaction to the error in the process variable. (Visioli A. 2006, 3–5)

The integrator time constant affects the integrative action in the PID. The integrative action is meant to eliminate steady state error. Steady state error is caused when the proportional action leaves behind a small error in the process variable. Because the proportional action is directly proportional to the error in the process variable, as the error gets smaller, so does the proportional action. Eventually the proportional action is too weak to correct the smallest error, leaving a permanent error in the process variable. The integrative action reacts to accumulated error. This means that as long as there is error in the process variable, the integrative action will work to correct it. In the water tank system, the integrator time constant (Ti) is high at 400s. This time constant has a stabilizing effect on the integrative control. A low integrative time constant makes adjustment faster but runs the risk of accumulating too much error. As the integrative part of the PID controller adjusts based on accumulated error, a high integrative time constant prevents excessive accumulation of error, known as controller windup, by increasing the reaction time to accumulated error. This allows the controller output to correct the process variable without overshooting. The effect of a high integrator time constant can be seen in figure 1. The strong reaction to error in the process variable created by a large gain variable and the derivative action is stabilized by the integral action. It prevents overshooting of the level but also takes longer to correct the level valve 2 is opened. In figure 2, the level does not seem to return at all. This is because the level is increasing at a slow pace. (Visioli A. 2006, 5)

The derivative action of the PID controller serves as an “anticipatory” action. The derivative action predicts errors in the process variable and counteracts them. (Visioli A. 2006, 6) Derivative action works by measuring the slope of the error in the process variable, or how fast the error happens. The effects of the derivative action can be seem in figure 2. When the setpoint (seen in figure 1) is changed to 2m instead of 0m, the instant change in error causes the derivative action to spike. The derivative action produces a vertical control input. This is because the action measures the slope of the error. The error happens instantly so the slope is vertical. When valve 2 is opened at the 15s mark, the slope of the error is such that the resulting controller output is relatively small compared to the setpoint change. To define how far in the future the derivative predicts error, a derivative time constant (Td) is used. The derivative time constant used in the water tank system is 1s. This makes the derivative action balanced. It is neither too fast, causing instability, nor too slow to react. The method of measurement used by the derivative action lends itself to be easily disrupted by noise in the process variable. (Smuts J. 2010)

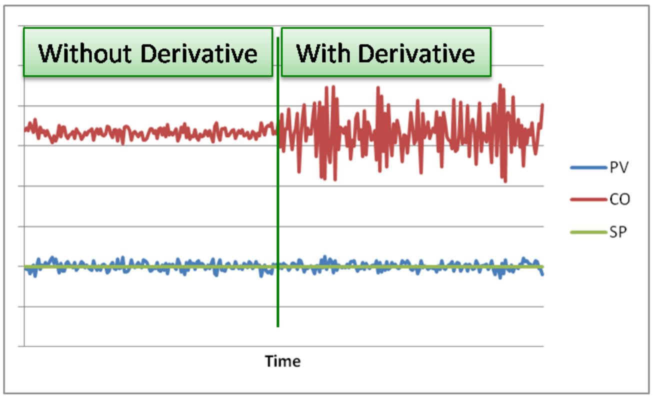
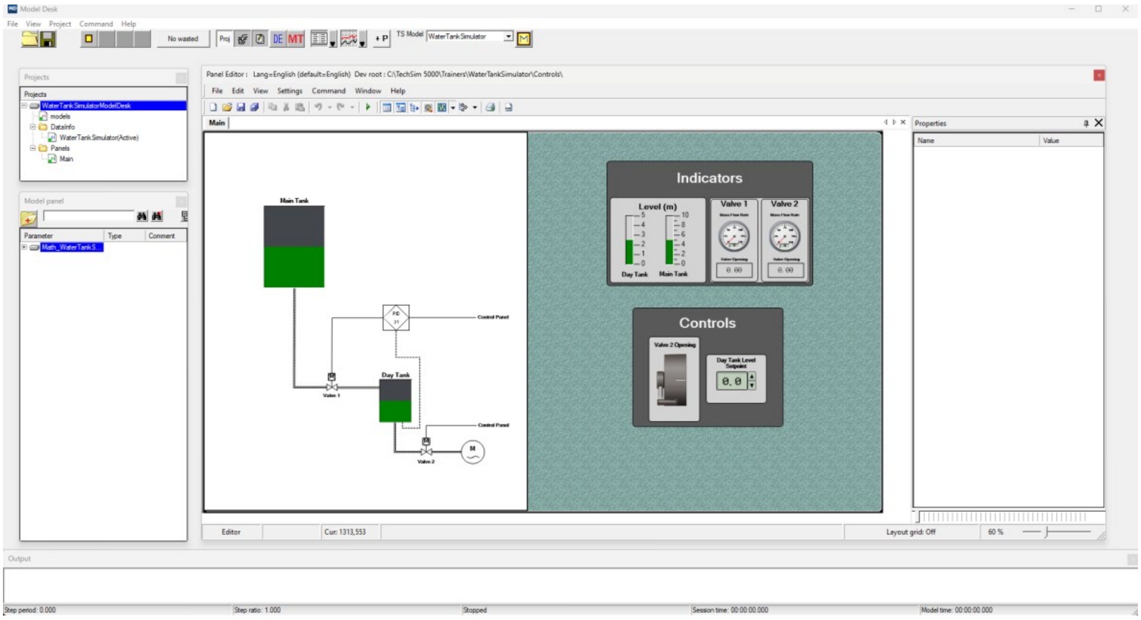


FIGURE 3. The affect of noise in the derivative action. PV = Process variable, CO = Controller output, SP = Setpoint. (Smuts J. 2010)

Smuts (2010) describes noise as ”small, random, rapid changes in the PV, and consequently rapid changes in the error”. The derivative action reacts to these small and rapid changes and amplifies their effects. This is because the derivative action reacts to the slope of the process variable. No matter how big the error, if the slope of said error is steep, the derivative action will predict a sharp increase in error and will equal the corrective action. Even if the steep signal change in the process variable turns out to be due to noise, the derivative control action will still be large. This leads to unwanted adjusting of the process variable and in a real system, to premature wearing out of mechanical components. This is why derivative action is often accompanied by a filter, for example, a low-pass filter. A low-pass filter attenuates high-frequency noise in the process variable, stabilizing the controller output. Smuts (2010)

5.2 Building the control panel

After building the mathematical model for the simulator, I moved on to build the control panel. Because the control panel is the visual side of the simulator that users interact with, the modeling of the control panel was relatively easy compared to the mathematical model, as it is only an interactive element of the simulator. It does not hold any control logic nor define the dynamics of the simulator.

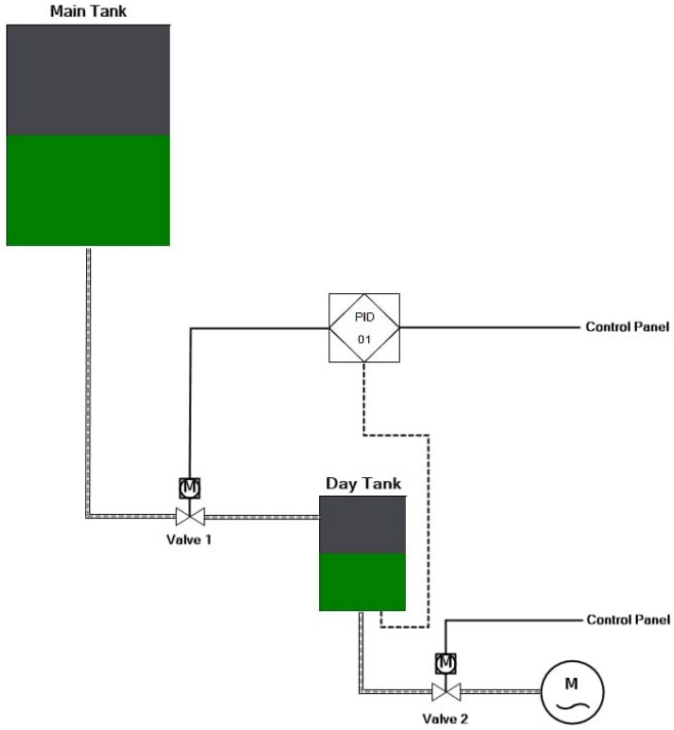


PICTURE 6. Water tank simulator control panel in Model Desk.

The programs used to build the control panel were Model Desk, Panel Editor & Controls and Datainfo Editor. As described in the chapter on SimMaker, the programs in the software package are integrated with each other. In this case, PE and DE are both operating inside MD. In picture 6, the layout of Model Desk can be seen. The projects window is seen in the left top corner of the of the program window. Here the user can access crucial information related to simulator like datainfo files and panels. Beneath the project window is the model panel window. The variables from the mathematical model can be accessed here. In the center is the panel editor window.

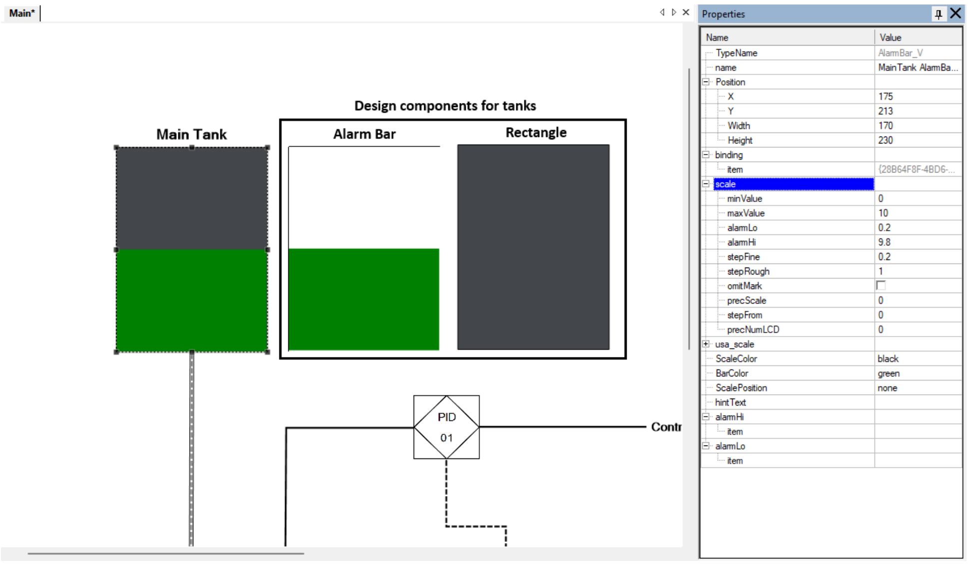
5.2.1 Visual diagram

In the control panel shown in picture 6, a visual representation of the simulated system can be seen on the left side of the control panel. This visualization can be thought of as the visual diagram for the system.



PICTURE 7. Visual diagram for Water tank simulator.

The diagram is mostly comprised from noninteractive elements such as the pipes, valves and controller. The only interactive elements in the diagram are the main tank and day tank. The levels of these tanks are actively represented by level bars with alarm functions. Because these alarm bars do not have their own backgrounds but are just colored bars, dark grey rectangles were added behind them to give an impression of a tank with a rising or falling level (picture 7).



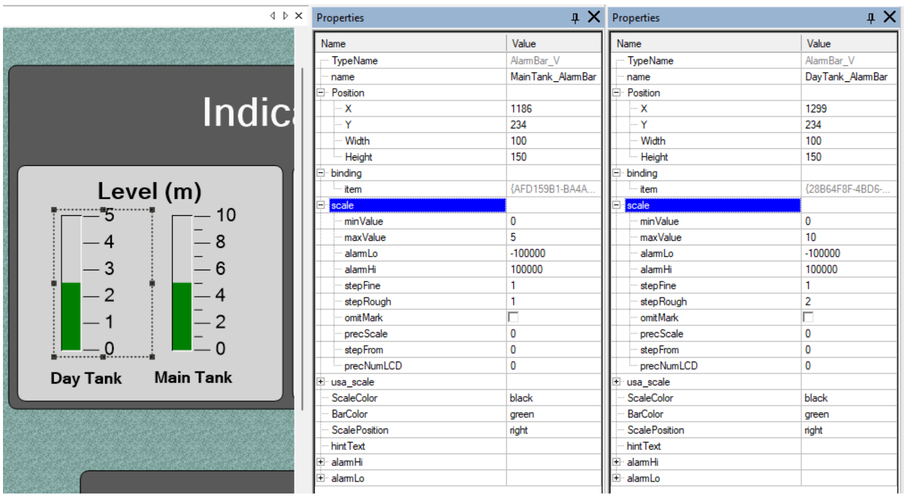
Name	Value
TypeName	AlarmBar_V
name	MainTank AlarmBa...
Position	
X	175
Y	213
Width	170
Height	230
binding	
item	{28B64F8F-4BD6-...
scale	
minValue	0
maxValue	10
alarmLo	0.2
alarmHi	9.8
stepFine	0.2
stepRough	1
omitMark	☐
precScale	0
stepFrom	0
precNumLCD	0
usa_scale	
ScaleColor	black
BarColor	green
ScalePosition	none
hintText	
alarmHi	
item	
alarmLo	
item	

PICTURE 8. Main tank design components and properties.

To represent the tanks in the mathematical model correctly, the maximum values of the alarm bar level indicators were set at the maximum height of the tanks – 10m for the main tank height and 5m for the day tank. This way, the alarm bar is at the same level as the mathematical model. The rest of the visual diagram are noninteractive visual design elements, thus I will not go to detail on their design.

5.2.2 Indicators

The controls and indicators are situated on the right half of the control panel in their own panels. The indicator panel has labeled indicators for levels, mass flow rates and valve openings; with levels in their own cluster and everything else together (picture 6).

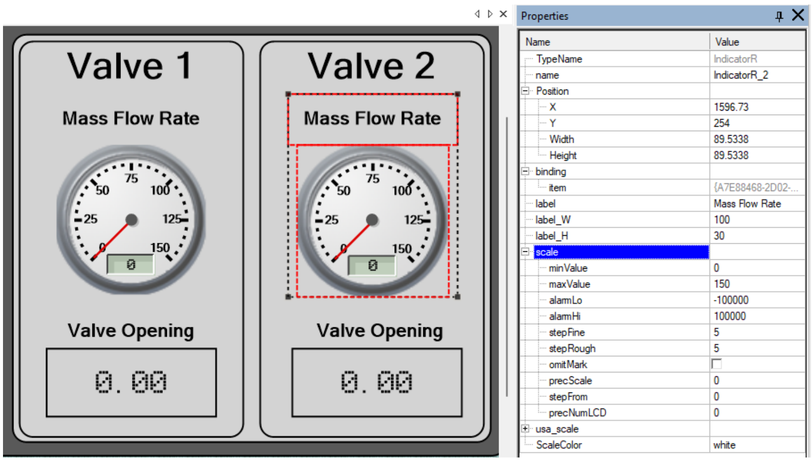


Name	Value
TypeName	AlarmBar_V
name	MainTank_AlarmBar
Position	
X	1186
Y	234
Width	100
Height	150
binding	
item	{AFD159B1-BA4A...
scale	
minValue	0
maxValue	5
alarmLo	-100000
alarmHi	100000
stepFine	1
stepRough	1
omitMark	☐
precScale	0
stepFrom	0
precNumLCD	0
usa_scale	
ScaleColor	black
BarColor	green
ScalePosition	right
hintText	
alarmHi	
alarmLo	

Name	Value
TypeName	AlarmBar_V
name	DayTank_AlarmBar
Position	
X	1299
Y	234
Width	100
Height	150
binding	
item	{28B64F8F-4BD6-...
scale	
minValue	0
maxValue	10
alarmLo	-100000
alarmHi	100000
stepFine	1
stepRough	2
omitMark	☐
precScale	0
stepFrom	0
precNumLCD	0
usa_scale	
ScaleColor	black
BarColor	green
ScalePosition	right
hintText	
alarmHi	
alarmLo	

PICTURE 9. Level indicator properties in indicator panel.

Picture 9 shows that, as in the design of the water tanks in the visual diagram, the level indicators in the level cluster were set with the maximum height being the maximum height of the tanks in the mathematical model. The scale next to the indicator bar had to also be set to the correct parameters using stepFine and stepRough. Rough step indicates how many steps are left unmarked. In the day tank, rough step is 1. This means that every step is numbered. In the main tank, rough step is two and thus every other step is numbered. Fine step dictates the scale of each step. A fine step of 1 means that the scale is marked at every whole number. When the parameter is 0.5, the scale is marked at every 0.5 interval.



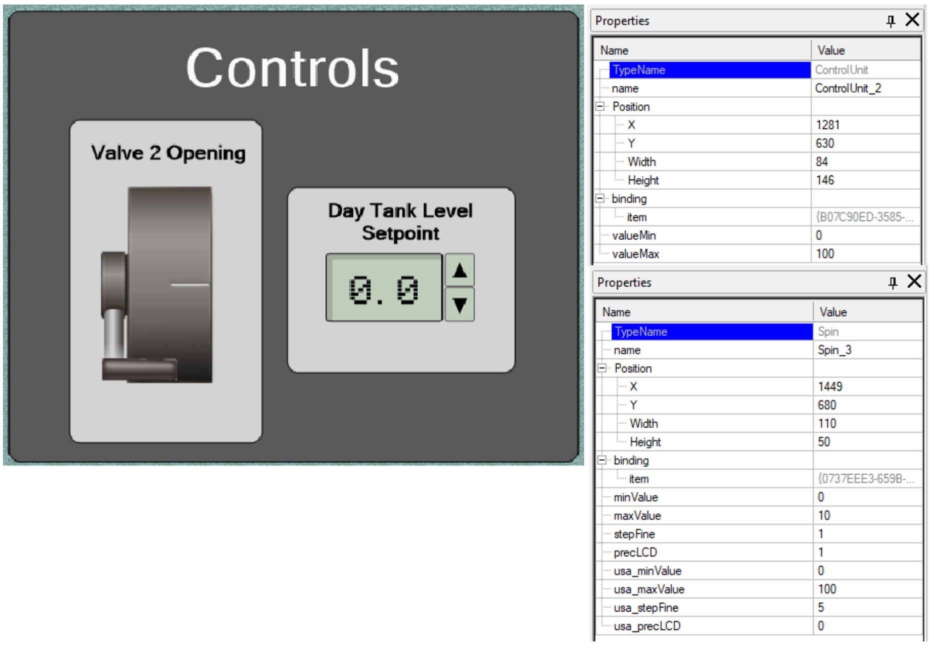
Name	Value
TypeName	IndicatorR
name	IndicatorR_2
Position	
X	1596.73
Y	254
Width	89.5338
Height	89.5338
binding	
item	{A7E88468-2D02-...
label	Mass Flow Rate
label_W	100
label_H	30
scale	
minValue	0
maxValue	150
alarmLo	-100000
alarmHi	100000
stepFine	5
stepRough	5
omitMark	☐
precScale	0
stepFrom	0
precNumLCD	0
usa_scale	
ScaleColor	white

PICTURE 10. Valve indicator and properties.

The valve indicator cluster situated next to the level cluster shows the mass flow rates and valve opening percentages of the valves (picture 10). The scales for the indicators showing the mass flow rates had to also be set up to show correct range of flow rate. The fine step used is 5. This means that every fifth whole number is marked. Rough step being 5 means that every fifth mark is numbered. The range for the meters were set at 150 kg/s. The valve percentage scales didn't need adjusting, because they are simple numeric indicators. In these indicators, a fully opened valve shows as 1.00, and fully closed valve as 0.00.

5.2.3 Controls

The control panel for the water tank simulator is simple as the only controllable objects in the simulator are the two valves. The control methods differ significantly. As mentioned in the design of the mathematical model and illustrated in the visual diagram of the control panel (picture 7), valve 1 controls water flow into the day tank as commanded by the PID controller. The only control given by the user is the wanted height of the level in the day tank.



Name	Value
TypeName	ControlUnit
name	ControlUnit_2
Position	
X	1281
Y	630
Width	84
Height	146
binding	
item	{B07C90ED-3585-...
valueMin	0
valueMax	100

Name	Value
TypeName	Spin
name	Spin_3
Position	
X	1449
Y	680
Width	110
Height	50
binding	
item	{0737EEE3-659B-...
minValue	0
maxValue	10
stepFine	1
precLCD	1
usa_minValue	0
usa_maxValue	100
usa_stepFine	5
usa_precLCD	0

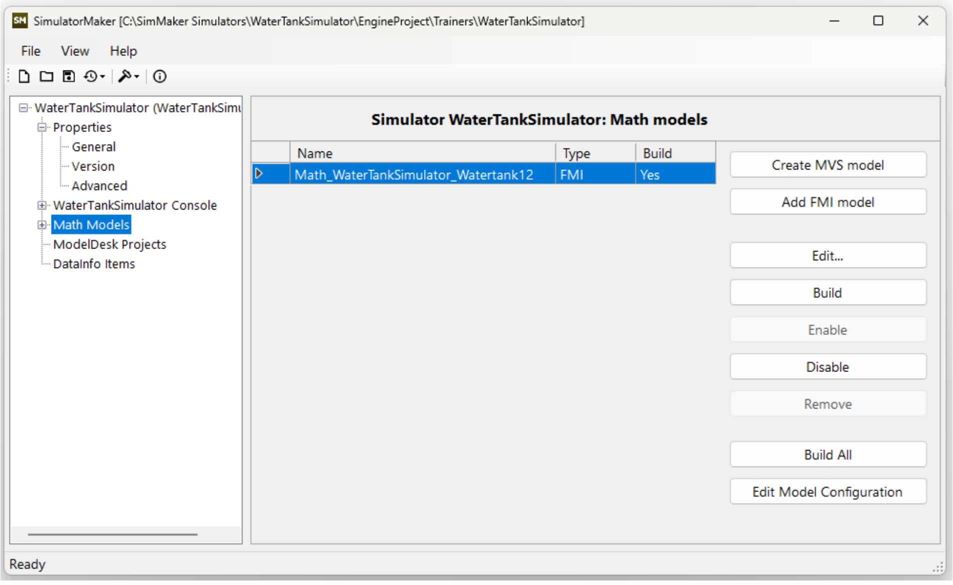
PICTURE 11. Controls for water tank simulator.

As seen in picture 11, the day tank level setpoint is controlled by a ”spin” component (Spin_3 as seen in the properties window). This is a numerical control instrument that can be adjusted with the accompanying up and down arrows. Fine step was set at 1. This means that the level is adjusted a meter at a time. The maximum level for the spin control was set at 10, or 10m. As the day tank is only 5m in height, setting the level to 10m would cause an overflow. This is cancelled in the math model with the limiter set in the PID controller (limiter1 seen in picture 3). This limiter restricts the accepted level height at 5m to prevent user-caused overflow.

In comparison to valve 1, valve 2 is controlled directly by the user. In the visual diagram of the system, valve 2 leads to the “engine”. Because of this design, I decided to visualize the control unit of valve 2 as a throttle unit. This makes visualizing the system easier. The more the user opens the throttle, the faster the day tank drains as more fuel is consumed. The throttle unit was set to a range of 0 – 100. Normally this would make the valve go from closed to fully open instantly as the opening for the valve is only 0 – 1. This was addressed in the binding of the mathematical model to the control panel.

5.3 Connecting the mathematical model and control panel

To connect the mathematical model to the control panel, a functional mock-up unit (FMU) file had to be created from the mathematical model made with Wolfram system modeler. To do this, I received help from the Wärtsilä Marine R&D simulation development team as I didn't have the ability to do it myself. Because of this, I cannot describe the conversion process from math model to FMU. With the original mathematical modelling software integrates into the SimMaker software package, MV Studium, an FMU file does not need to be created, simplifying the binding process. The final FMU file was mounted to SimMaker from the Math Models section in the project tree.

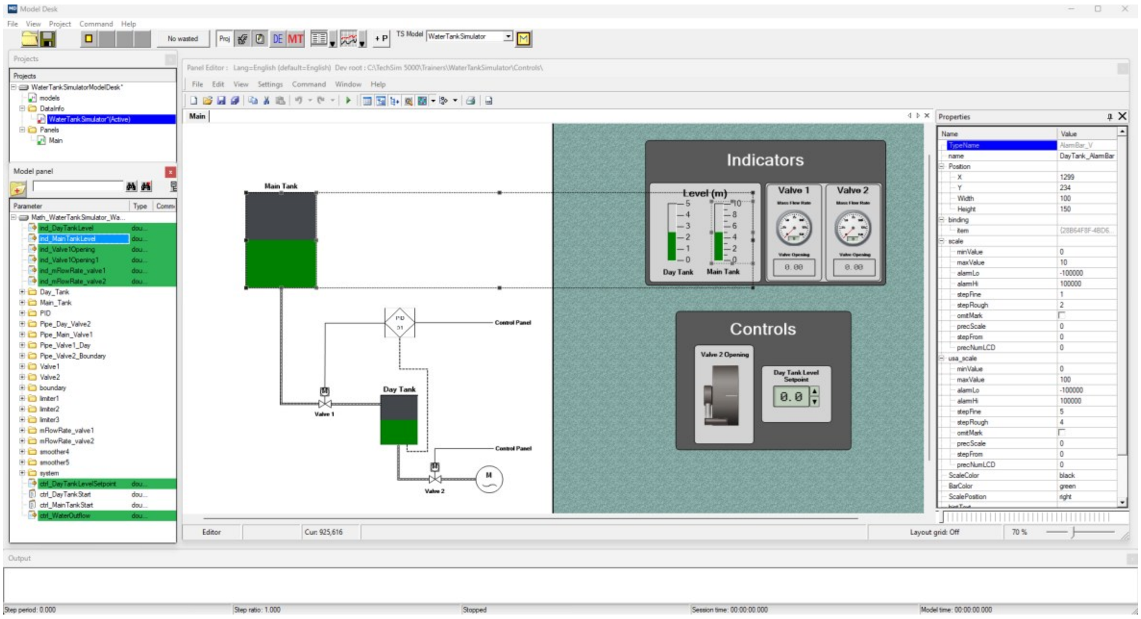


Simulator WaterTankSimulator: Math models

Name	Type	Build
Math_WaterTankSimulator_Watertank12	FMI	Yes

PICTURE 12. FMU file added to SimMaker.

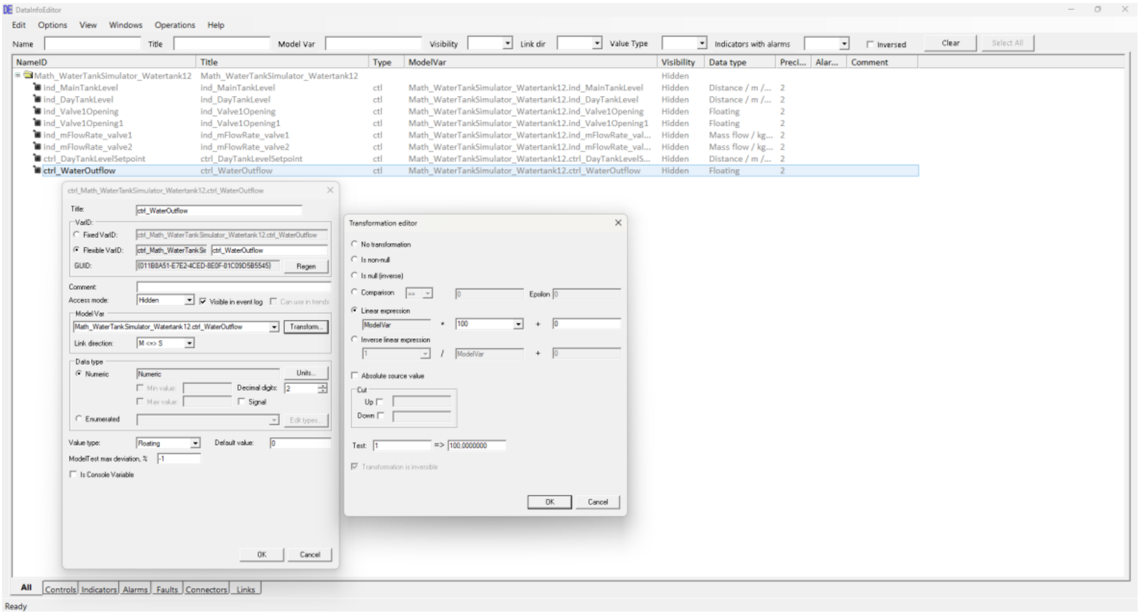
After adding the FMU to the project, the binding can continue in Model Desk. To connect the FMU to the control panel, a datainfo file needed to be created. Datainfo files store the links between the mathematical model and the control panel. To edit these files. Datainfo Editor is used.



PICTURE 13. Binding mathematical model variables in Model Desk. Main tank level variable highlighted.

In Model Desk, variables of the connected mathematical model can be seen in the Model panel window. These can now be dragged and dropped to bind them to the control panel elements. When connected, the variable in the Model panel is colored green. The same link can also be used on multiple elements, as seen in picture 13, where selecting the variable ind_MainTankLevel outlines the main tank in the visual diagram and the main tank level in the instrument panel. The same is done with the day tank. The rest of the instruments and controls were also bound to the mathematical model by dragging the variables from the Model panel window.

After binding the control variable of valve 2 to the control unit in the control panel, issues with the control unit had to be addressed to ensure correct operation. When designing the control panel, the control unit was set to an adjustment range of 0 – 100. This causes an issue, when trying to control valve 2, as the valve is set to an adjustment range of 0 – 1. If the control unit had the same range, it would only have 2 positions, 0 or 1. To solve this, changes needed to be made to the scaling of the variable that controls valve 2 in Datainfo Editor.



NameID	Title	Type	ModelVar	Visibility	Data type	Preci...	Alar...	Comment
Math_WaterTankSimulator_Watertank12	Math_WaterTankSimulator_Watertank12			Hidden				
ind_MainTankLevel	ind_MainTankLevel	ctl	Math_WaterTankSimulator_Watertank12.ind_MainTankLevel	Hidden	Distance / m /...	2		
ind_DayTankLevel	ind_DayTankLevel	ctl	Math_WaterTankSimulator_Watertank12.ind_DayTankLevel	Hidden	Distance / m /...	2		
ind_Valve1Opening	ind_Valve1Opening	ctl	Math_WaterTankSimulator_Watertank12.ind_Valve1Opening	Hidden	Floating	2		
ind_Valve1Opening1	ind_Valve1Opening1	ctl	Math_WaterTankSimulator_Watertank12.ind_Valve1Opening1	Hidden	Floating	2		
ind_mFlowRate_valve1	ind_mFlowRate_valve1	ctl	Math_WaterTankSimulator_Watertank12.ind_mFlowRate_val...	Hidden	Mass flow / kg...	2		
ind_mFlowRate_valve2	ind_mFlowRate_valve2	ctl	Math_WaterTankSimulator_Watertank12.ind_mFlowRate_val...	Hidden	Mass flow / kg...	2		
ctrl_DayTankLevelSetpoint	ctrl_DayTankLevelSetpoint	ctl	Math_WaterTankSimulator_Watertank12.ctrl_DayTankLevelS...	Hidden	Distance / m /...	2		
ctrl_WaterOutflow	ctrl_WaterOutflow	ctl	Math_WaterTankSimulator_Watertank12.ctrl_WaterOutflow	Hidden	Floating	2		

ctrl_Math_WaterTankSimulator_Watertank12.ctrl_WaterOutflow

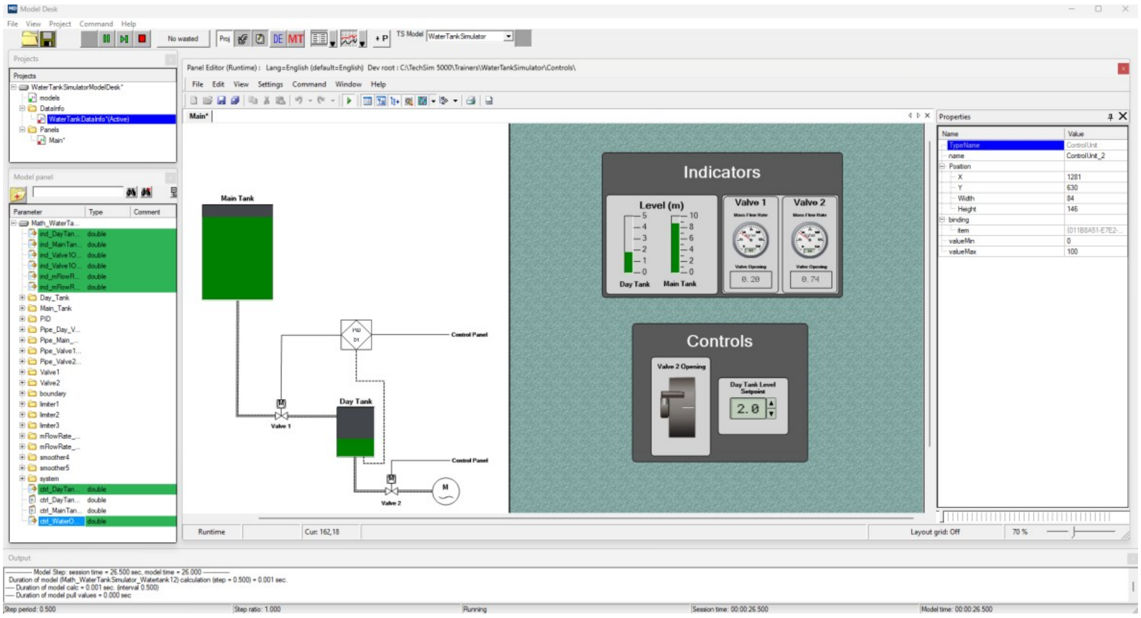
Title: ctrl_WaterOutflow
VarID:
○ Fixed VarID: ctrl_Math_WaterTankSimulator_Watertank12.ctrl_WaterOutflow
● Flexible VarID: ctrl_Math_WaterTankSir ctrl_WaterOutflow
GUID: {011B8A51-E7E2-4CED-8E0F-81C09D5B5545}
Comment:
Access mode: Hidden ☑ Visible in event log ☐ Can use in trends
Model Var: Math_WaterTankSimulator_Watertank12.ctrl_WaterOutflow
Link direction: M <=> S
Data type
● Numeric: Numeric
☐ Min value:
☐ Max value:
Decimal digits: 2
☐ Signal
○ Enumerated
Value type: Floating
Default value: 0
ModelTest max deviation, %: -1
☐ Is Console Variable

Transformation editor

○ No transformation
○ Is non-null
○ Is null (inverse)
○ Comparison == 0 Epsilon 0
● Linear expression
ModelVar * 100 + 0
○ Inverse linear expression
1 / ModelVar + 0
☐ Absolute source value
Cut
Up ☐
Down ☐
Test: 1 => 100,0000000
☑ Transformation is inversible

PICTURE 14. Linear expression transformation for ctrl_waterOutflow.

In Datainfo Editor, a list of variables from the mathematical model are listed (picture 14). The program is used to manage these variables and their connections to the control panel. The variable for controlling valve 2 (ctrl_WaterOutflow) was now modified to address the scaling for control unit of valve 2. To do this, I selected transform from the variable property window. In the Transformation editor, I selected linear expression and multiplied the variable by 100. This way, when receiving signal of 100, the variable showed 1 as seen in the test window in the Transformation editor. Now the control unit in the control panel adjusted from 0 to 100 with a value of 1 in the control unit corresponding to 0.01 in the variable. This ensured that the control unit worked properly. In addition to this scaling change, measuring units were given to other variables from the unit setting in the variable properties. After connecting the mathematical model of the simulator to the control panel, the simulator could be tested. To do this, I entered simulation mode by pressing the yellow rectangle in the top left corner of the Model Desk window (picture 13).



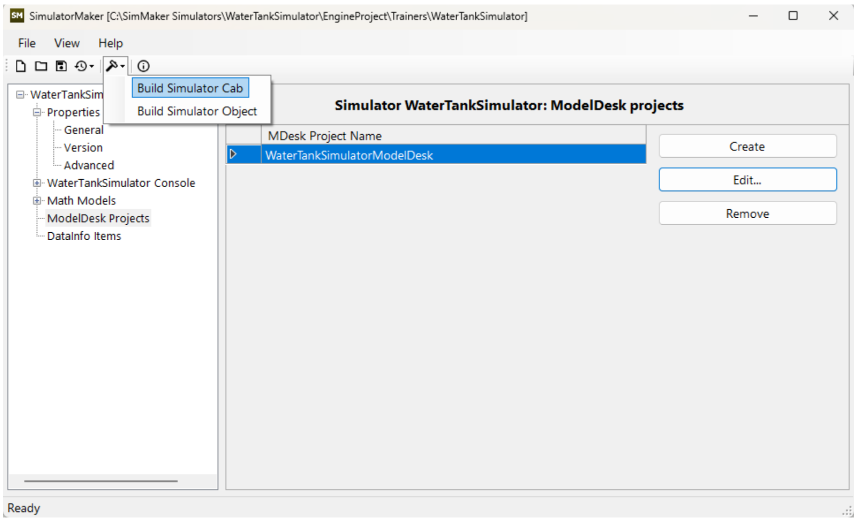
```
------------ Model Step: session time = 26.500 sec, model time = 26.000 ------------
 Duration of model (Math_WaterTankSimulator_Watertank12) calculation (step = 0.500) = 0.001 sec.
---- Duration of model calc = 0.001 sec. (interval 0.500)
---- Duration of model pull values = 0.000 sec
```

PICTURE 15. Testing the simulator in Model Desk.

Controls for the simulation runtime can be seen in the top left corner of the Model Desk window (picture 15). After starting the simulation, the system was operated as intended in the TechSim platform. Testing was successful. After the successful tests, the simulator could be mounted onto the TechSim platform.

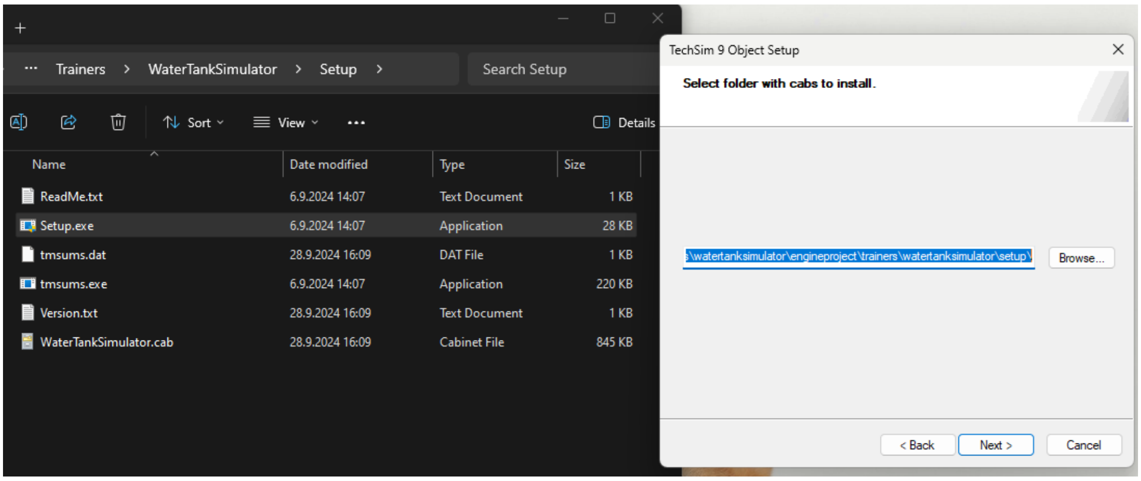
5.4 Mounting the simulator to the TechSim platform

To mount the simulator to the TechSim platform, the simulator was archived into a cabinet file. This cabinet file contains everything needed by the simulator to work in the TechSim environment.



PICTURE 16. Building simulator cabinet file.

After building the cabinet file for the simulator (picture 16), the simulator can be mounted onto the TechSim platform. When the building of the cabinet file was complete, a setup application was created in the file location of the simulator.



Name	Date modified	Type	Size
ReadMe.txt	6.9.2024 14:07	Text Document	1 KB
Setup.exe	6.9.2024 14:07	Application	28 KB
tmsums.dat	28.9.2024 16:09	DAT File	1 KB
tmsums.exe	6.9.2024 14:07	Application	220 KB
Version.txt	28.9.2024 16:09	Text Document	1 KB
WaterTankSimulator.cab	28.9.2024 16:09	Cabinet File	845 KB

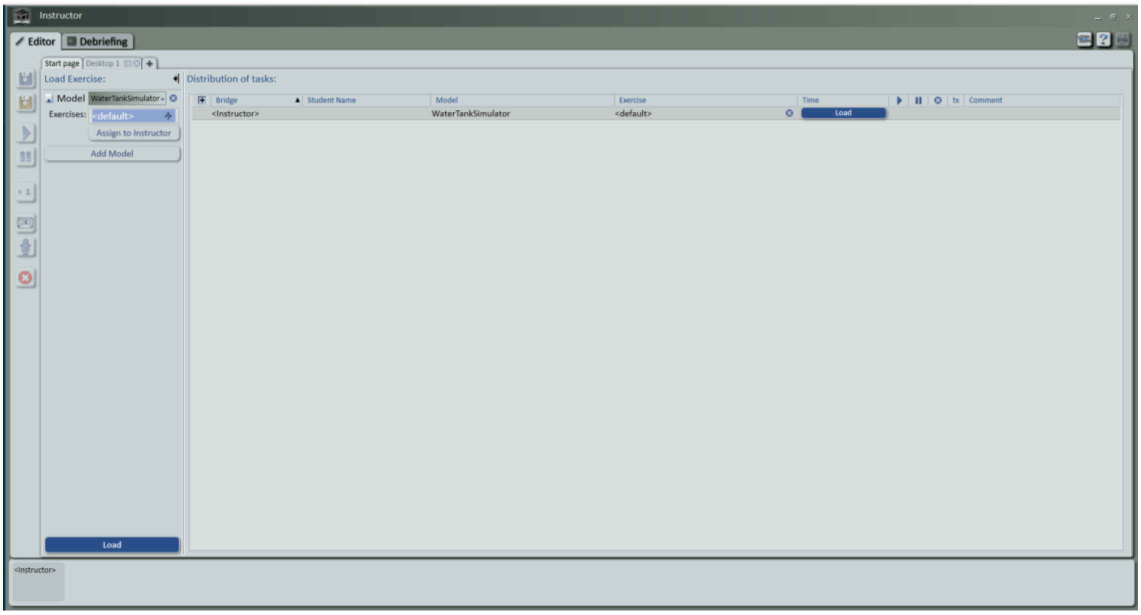
TechSim 9 Object Setup

Select folder with cabs to install.

s\watertanksimulator\engineproject\trainers\watertanksimulator\setup\

PICTURE 17. Mounting simulator into the TechSim platform.

Opening the setup application starts the object setup program for TechSim with the cabinet file selected (picture 17). Following the installation process completes the setup.



Load Exercise:

Model WaterTankSimulator

Exercises: <default>

Distribution of tasks:

Bridge	Student Name	Model	Exercise	Time	Comment
<Instructor>		WaterTankSimulator	<default>	Load	

<Instructor>

PICTURE 18. TechSim 9 instructor environment. Water tank simulator mounted in the training environment.

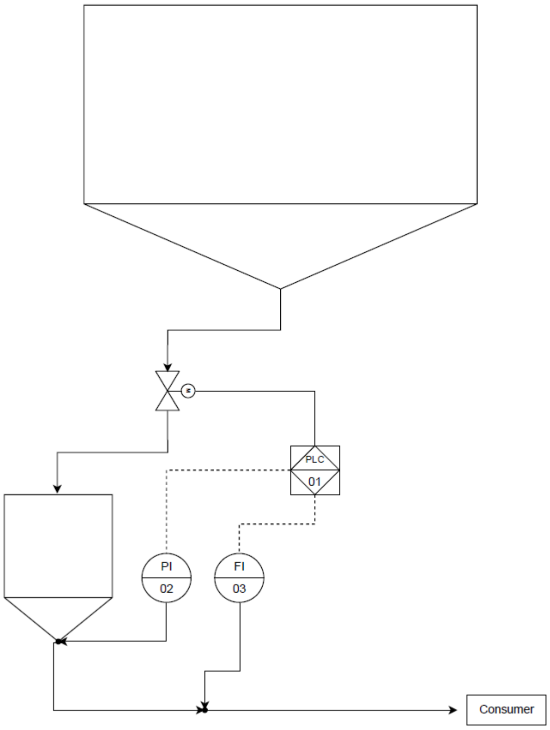
Once the setup was complete, the instructor environment (picture 18) of the TechSim platform could be used to load the simulator into the training space. This was done by selecting the simulator from the Model dropdown menu and dragging the default exercise onto the Distribution of tasks list. The simulator was then loaded into the training space (picture 1).

6 TRAINING SIMULATOR DEVELOPMENT IN WLSA

The training simulators developed by WLSA today differ from those made with SimMaker in composition and development. WLSA creates both digital simulators and hardware-based simulators. The digital simulators include VR-based, mechanical installation training simulators and rudimentary Powerpoint-based simulators while the hardware-based simulators are process simulators. The use of SimMaker in WLSA aims to supplement or replace hardware-based process simulators used in training. In this part of the thesis, I will describe and compare the development of the hardware-based process simulators built currently in WLSA to the digital process simulators built with SimMaker. It is important to note that SimMaker is not yet in use in the WLSA training team. Because of this, I will use my development process alongside the development experiences of the simulation development team at Wärtsilä Marine R&D as a template to describe a possible future development process in WLSA. It is also important to note that SimMaker is not yet fully matured as a product. During my development process, multiple mathematical programs were tested to find a suitable solution for my project. These are subject to change in the future.

6.1 SimMaker simulator development

The development of a training simulator with SimMaker begins with assessing the requirements of the simulator. Training simulators are created to fulfill a certain learning objective. By assessing the requirements of the simulator from the perspective of the learning objective, estimations for the simulator scope and fidelity can be made. The requirements for the water tank simulator were to showcase the development steps in creating a simple simulation model using SimMaker. To achieve this, the simulator needed to be large enough to demonstrate the use of the different parts of SimMaker, such as the mathematical modeling, Model Desk and Panel Editor and Controls. After assessing the requirements, a description of the simulator is created. This serves to give the developers a picture of the desired product. To support this, a preliminary sketch of the simulator is made.



PICTURE 19. Preliminary sketch for water tank simulator.

To build the proposed simulator, required tools need to be defined. Tools such as a mathematical modelling program and possibly a 3D modelling program. For the water tank project, only a mathematical modelling tool was needed in addition to the SimMaker software package. This is one of the strong suites of SimMaker. It gives developers a standard set of tools to use in the development process, minimizing the need to define required tools. In addition, standardization provides uniformity across different projects.

After the simulator model has been defined and the necessary programs have been acquired, the workload is divided between the team members. This workload involves tasks such as scheduling, prioritization, data collection, modelling, testing, documentation, user data collection and improvement implementation. The division of the workload is dependent on the team size. In the water tank project, there was no division of the workload, as I worked on everything with the assistance of the Marine R&D simulator development team. With the workload divided, the creation of the simulator can begin. The development is similar to the development of the water tank model described in chapter 5. In the water tank simulator, the steps were developed in succession as opposed to an actual simulator development. In an actual simulator project, due to the team size and the abstraction layers in SimMaker, these steps would be divided between team members and developed simultaneously, provided that the available data on the simulator is sufficient. Sometimes it is not possible due to the unique dynamics of certain systems. Problems can rise along the way, that can halt the project and force changes to other parts of the system. A mathematical modeling issue can, for example, require large changes to the model that lead to changes required in the data exchange and UI layers. After the initial version of the simulator is compiled, testing is conducted to find faults and improvement needs for the simulator. The simulator is shared with a testing group and feedback is collected. The improvements are implemented, and the official version of the simulator is published. As opposed to current process simulators in WLSA that use a combination of hardware and software, virtual simulators built with SimMaker can be shared through cloud, making distribution of the simulators fast. After the official publication, the simulator can be implemented into training. Lifecycle support is often provided for further improvements.

6.2 Current state of training simulator development in WLSA

The simulators currently developed in WLSA are either hardware-based or software-based. The digital simulators in Wärtsilä are mechanical installation simulators that are used to practice the installation of different mechanical parts in a VR environment. The focus of this chapter is the hardware-based process simulators, as the simulators built with SimMaker aim to supplement or even replace them. As with the simulator development process described in the previous chapter, the development of a hardware-based process simulator should start with evaluating the requirements of the simulator using the learning objectives of the training course it’s developed for. After the requirements for the simulator are defined, an investment proposal is made. After the investment proposal is accepted, the building process can begin. Due to the diversity of simulators in WLSA, there is no unified building method and so I will not describe these methods in detail. In WLSA, the development of process simulators often does not follow the before mentioned steps. Usually, the simulators are built first and the suitable training purposes are thought of afterwards. The simulators are then used in training that match the capabilities of the simulators. This is not an optimal method.

With the current method of developing hardware-based process simulators, a streamlined development process can be hard to achieve. One reason for this is because WLSA uses actual process equipment. Using actual process equipment in simulators can be beneficial, as the trainees get to familiarize themselves with the features of the equipment before operating them in actual processes. The significant downside of this method is that it poses significant challenges in development due to the extensive requirements for their operation. This equipment demands numerous safety and operational signals be satisfied before they can function correctly. Consequently, to integrate such equipment into a simulator, it becomes necessary to develop additional hardware systems to fulfill these requirements. This, in turn, leads to an exponential increase in the simulator's size and complexity. The process of building extensive hardware infrastructure to accommodate the operational needs of actual equipment consumes substantial resources, both in terms of time and cost. The resources needed to build a simulator can be the equivalent of a full man-week to 100 man-weeks. This resource-intensive approach may not be proportionate to the training benefits gained. Additionally, the upkeep of the simulators can be difficult, as the development methods and tools are not standardized. This means that methods used in the development of simulators differ simulator to simulator. To provide upkeep for one simulator, one must commit significant time resources to understand the tools used in building the simulator and how the simulator itself operates.

In WLSA, there are currently no dedicated simulator development teams. The simulators developed in WLSA are built by the instructors themselves. One benefit of this is that the instructors are well versed in the simulator, having had a part in the development process. This also makes the instructors deeply knowledgeable on the broader subject matter. In my interview with Nikkanen (2024), he stated that:

> I built the first LNGPac simulator for WLSA as a thesis work. After the simulator was finalized, I could immediately use it in training. Other instructors needed years to become proficient in using it. I also gained very valuable information regarding the LNGPac product. I could immediately start training the ship crews. (Nikkanen interview 2024)

The lack of dedicated simulator development teams at WLSA also causes several issues. one of these is that instructors have added workload on top of their training assignments. This limits the available time resources to dedicate to the development of simulators, which is a reason why simulators are not standardized. To save time, instructors work with resources most convenient and available to themselves, making the development of simulators in one region different from those of another. Another downside of this development method is that the amount of people working on the simulators are typically one to two, thus, the development time of a simulator is longer. Replication of the simulators is also not taken into much consideration, complicating the process if a similar simulator is needed in another location.

6.3 Comparison

In this chapter, I will compare the pros and cons of the simulator development process with SimMaker to the current day methods. The pros and cons of the SimMaker development process are listed in Table 1, while those of the current development process are detailed in Table 2.

TABLE 1. Pros and cons of simulator development using SimMaker.

	Pros	**Cons**
SimMaker development	Learning goals-based projects	SimMaker not yet mature as product
	Larger, dedicated team	Task coordination
	Standard tools	Tools require learning
	Growing library	Current library nonexistent
	Cloud sharing	Total virtualization
	Parallel development	Parallel development not always available
	Easier to limit in scope	
	Support available	
	Ready infrastructure for training in TechSim	

TABLE 2. Pros and cons of simulator development using current methods in WLSA.

	Pros	**Cons**
Current development	Actual process equipment	Build first, training objective later
	Increased instructor proficiency	Small development team
	Hardware	Hardware
		Hard to limit in scope
		Non-reusable parts
		Non-standard tools
		Sequential development
		Resource intensive
		Replication difficult
		Training infrastructure need to be built into simulator

One of the biggest improvements to simulator development when comparing current methods to those used with SimMaker is the possibility to develop simulators based on learning objectives. This is hard to do with the current development methods in WLSA as the full capabilities of a simulator are not realized until the simulator itself is finished. The development methods used in SimMaker simulator development also enables the limiting of the simulator scope beforehand. This is mainly due to the nature of digital simulators. As digital simulators do not use actual process equipment, the requirements to run the simulator are lower and the simulator can be limited to parts actually needed in the training, with the other components being either simplified or left out if possible. An example on the simplification of models is in the water tank simulator. The consumer of the “fuel” in the model is an engine. As the simulator does not require anything else but the modelling of the tank, pipes and valves; the engine in the mathematical model is just a boundary element that is fed via a valve. Simplifications like these are enabled by digital simulators. Complex systems like engine modules can be simplified or if possible, removed entirely to make the process less resource intensive. It could still be beneficial to continue using actual equipment in simulator training because as stated in chapter 3.3.2, a certain fidelity should be achieved in simulators to ensure correct practice in actual processes. Using actual equipment could be a good way to ensure that the process simulator works as similarly to the actual process as possible. The importance of keeping the actual equipment should be examined on an individual basis. Using digital simulators when possible, would keep the equipment costs lower compared to hardware simulators. This is because hardware simulators require the use of supporting hardware and software, which drive costs even higher. Additionally, training infrastructure does not need to be built into the simulators made with SimMaker as the simulators are run in the TechSim platform, which has its own built-in training infrastructure.

Another significant improvement is the dedicated simulator development team as well as the team size. With the current size of the development teams, the efficiency of the development process is low. A bigger, dedicated team would enable the tasks to be divided between different members of the team, decreasing the workload of each team member. Additionally, the team would only focus on the development of simulators, not training. Current simulator development methods benefit from having instructors develop the simulators. This allows the instructors to gain detailed knowledge on the subject and how the simulator works, enabling them to use the simulator in training immediately. This level of proficiency in the subject and simulator might take other instructors significantly longer to achieve. With a sizable development team, the abstraction layers in SimMaker facilitate the possibility of developing parts of the simulation at the same time. Although possible, this parallel development is not always possible depending on the system.

The development process using SimMaker is more efficient due to the standardized tools provided with the software package. Currently, WLSA instructors use various tools based on personal preference. Standardized tools would help development efficiency as instructors would no longer have to evaluate what tools to use in each simulator as they already have a pre-defined set of tools. Standardized tools would also help future upkeep as the structure of the simulators are consistent across all simulators, simplifying the upkeep. Instructors performing the upkeep would also be familiar with the tools used when building the simulator. Although developers need to learn these tools, support is available from the Wärtsilä Marine R&D simulation development team, who are experienced with SimMaker. Another advantage of SimMaker is that components from previous projects can be stored in component libraries for future use, reducing development time by eliminating the need for repeated development. This is not as easy with current simulators, as replicating the hardware itself means that new components must be purchased/manufactured. Digital simulators also have faster distribution, as they can be shared through cloud services and can be put to use quickly, assuming the receiving end has the equipment required for the specific simulator.

7 CONCLUSION

In this chapter I will summarize key research findings made in previous chapters as they pertain to my research objective as well as discuss their value and future contribution. The objective of this thesis was to evaluate the efficacy of using the Simulator Maker software package in Wärtsilä Land and Sea Academy to develop virtual process simulators for use in simulator training. To achieve this objective, my research question was “How could SimMaker benefit WLSA?” To provide an answer to this question, I started by creating a demo process simulator using SimMaker. Using my development experience with the demo simulator alongside the experiences of the simulation development team at Wärtsilä Marine R&D, I compared the development process of digital process simulators using SimMaker to that of current hardware-based process simulators in WLSA. In this comparison, the results indicate that the digital process simulators built with SimMaker offer significant advantages in comparison to current hardware-based process simulators in terms of efficiency and cost-effectiveness. This is due to advantages like the lack of physical hardware, use of standard tools and the possibility of using streamlined development methods such as designing the simulator according to the learning objective.

The use of actual physical process equipment in hardware simulators currently operational in WLSA increase the fidelity of the simulators. This also forces the building of complex and expensive systems due to the working requirements of the equipment. Digital simulators made with SimMaker enable developers to limit the scope of the simulator, simplify the system where possible as well as lower development costs due to less labor-intensive development methods. Standardized tools help in development and upkeep. During development, sufficient tools do not need to be evaluated before building the simulator as they are pre-defined. During upkeep, as the tools are standardized, the structure of simulators are consistent across models. Mathematical models for example, use the same modeling language. This simplifies upkeep as the tools used are consistent in all simulators. Hardware-based simulators are hard to design according to learning objectives as, due to their complexity, the full capabilities of the simulators are only revealed when the simulator itself is finished. Digital simulators enable the developers to define the scope and fidelity requirements of the simulator according to the learning objectives. In this way, the capabilities of the simulator are defined during the design phase.

In the literature review chapter, I cited Krishnan et al. (2017) stating that building simulators with sufficient fidelity can add considerable cost to the development phase. This was supported in my findings by the cost of process simulators currently built in WLSA. The simulators offer high fidelity due to using actual process equipment. Although the equipment enhances the fidelity of the simulator, this process equipment adds considerable resource costs. Increasing costs were also evident during the development of my demo simulator. As more details were added to the system, the more labour intensive the project became, increasing cost. These increased costs also highlight the importance of being able to determine the required fidelity and scope of the simulator. Not every simulator needs high fidelity and large scope.

When evaluating the reliability of this thesis, a key issue becomes apparent. Advantages in efficiency and cost effectiveness were observed during the development of the demo simulation. These advantages were also reported by the simulation development team at Wärtsilä Marine R&D during their projects. Despite these findings, it should be noted that my findings were based on one project, while the advantages experienced by WLSA instructors may differ from those observed by the simulation development team. To determine the full capabilities SimMaker can offer to WLSA instructors along with advantages and disadvantages of the software package, further research and testing should be conducted by WLSA instructors.

The objective of this thesis was to evaluate the efficacy of using SimMaker in WLSA to develop process simulators for use in training. To achieve this, I examined the specific advantages SimMaker could bring by using my experience with the software package and those of the Wärtsillä R&D simulation development team. However, this method of examining the efficacy of SimMaker lacks the user experience of WLSA instructors themselves. As stated before, this aspect of the thesis requires further examination by WLSA instructors. However, based on the findings made during the making of this thesis, I would recommend the adoption of digital simulators developed with SimMaker to supplement physical process simulators. All information presented in this thesis has been published with the consent of all participants. Sensitive information obtained throughout the research was kept strictly confidential.

SOURCES

Wärtsilä. n.d. Wärtsilä Official Website. Accessed 1.9.2024. https://www.wartsila.com/

Wärtsilä. n.d. History. Accessed 30.8.2024. https://www.wartsila.com/about/history

Wärtsilä. n.d. Strategy. Accessed 30.8.2024. https://www.wartsila.com/about/strategy

Wärtsilä. 2023. *Methanol: Fuel for Thought in Our Deep Dive Q&A*. Accessed 31.8.2024. https://www.wartsila.com/insights/article/methanol-fuel-for-thought-in-our-deep-dive-q-a

Wärtsilä. n.d. WLSA. Accessed 1.9.2024. https://www.wartsila.com/wlsa

Law, A. 2015. Simulation Modeling and Analysis. Accessed 17.8.2024.

University of Houston. n.d. Historical Simulation. Accessed 17.8.2024. https://uh.edu/~lcr3600/simulation/historical.html

IBM. n.d. What is Monte Carlo Simulation? Accessed 19.8.2024. https://www.ibm.com/topics/monte-carlo-simulation

Kim, T., Sharma, T., Bustgaard, M., Gyldensten, W.C., Nymoen, O.K., Tusher, H.M. & Nazir, S. 2021. The continuum of simulator-based maritime training and education. WMU Journal of Maritime Affairs (20), 135-150. Accessed 28.5.2024. https://doi.org/10.1007/s13437-021-00242-2

Bouhelal, A., Patel, H. & Patel, B. 2012. Value of Virtual Reality in Medical Education. Simulation Training in Laparoscopy and Robotic Surgery 2012. 39-48. Accessed 26.8.2024. https://doi-org.libproxy.tuni.fi/10.1007/978-1-4471-2930-1_5

Krishnan, D.G., Keloth, A.V., & Ubedulla, S. 2017. Pros and cons of simulation in medical education: A review. International Journal of Medical and Health Research, 3(6), 84–87.

Wärtsilä. 2018. Wärtsilä’s acquisition of Transas is completed. Accessed 10.9.2024. https://www.wartsila.com/media/news/04-05-2018-wartsila-s-acquisition-of-transas-is-completed-2174440

Wärtsilä. n.d. MethanolPac. Accessed 11.9.2024. https://www.wartsila.com/marine/products/gas-solutions/methanol-supply-systems/methanolpac

Nikkanen, Lasse. 2024. Engine & Cargo Simulators Product Lead. Wärtsilä. Interview 9.9.2024. Turku. Interviewer Mikko Rajander. Interview in writers possession.

Anish Wankhede. 2019. Bunkering is Dangerous: Procedure for Bunkering Operation on a Ship. Marine Insight 4.2.2019. Accessed 10.9.2024. https://www.marineinsight.com/guidelines/bunkering-is-dangerous-procedure-for-bunkering-operation-on-a-ship/

FMI Standard. 2023. FMI 3.0.1 Overview. Accessed 2.9.2024. https://fmi-standard.org/docs/3.0.1/#_overview