Co-simulation models for ship power management systems

Lucas Fransman, 1802351
Master's thesis in Computer Science
Supervisors: Dr. Bogdan Iancu & Dr. Sébastien Lafond
Faculty of Science and Engineering
Åbo Akademi University
2023

Abstract

The maritime sector plays a crucial role in the global economy. With over 80 percent of the world's trade by volume transported by sea, the maritime industry plays a critical role in the all more globalised world. However, the complexity of the maritime environment presents numerous challenges, including safety, energy efficiency, navigation, and operational efficiency. In this context, co-simulation is a promising tool for enabling predictive and proactive decision-making during large construction projects when building ships and during operation.

Co-simulation provides a more modular way of executing simulations than traditional all-in-one simulations. This is crucial for the maritime industry, as the expertise behind a complete ship is so diverse that no single entity has a deep understanding of it all. With co-simulation, we can split the simulation into smaller pieces which are then finally connected together into one complete simulation. Therefore, the construction of the simulation can be easily distributed over several groups with the appropriate expertise. The potential benefits of co-simulation are many. By offering the ability to simulate real-world maritime scenarios with a high degree of fidelity, co-simulation allows for risk mitigation, optimisation of operational procedures, increased safety, improved energy efficiency, and better planning and decision-making.

In this thesis, a co-simulation of a battery sub-system onboard a ferry has been implemented based on actual operational collected data. Further, a module to represent electrical car chargers has been implemented to gain an understanding of how the extra energy requirements such a system would impose on the ferry would affect the battery sub-system. This thesis demonstrates a use case for co-simulation within the maritime industry to gain insight into how an unknown component might integrate with an already existing ferry during the planning phase. The results display the potential co-simulation can play when experimenting with new technologies in the maritime industry, accelerating the positive impact technological innovation will

have on the industry.

Keywords: Co-simulation, Simulation, Open Simulation Platform, FMI, FMU, Hybrid ferry, Electric car charging

Preface

Working on this thesis has been an incredibly insightful and rewarding experience, while at some points quite demanding. I want to express my gratitude to my supervisors, Dr. Bogdan Iancu and Dr. Sébastien Lafond, for helping me and providing valuable advice and direction throughout this project. Further, I want to thank Bogdan for the time he has spent helping me structure this thesis and providing help with both grammatical and styling aspects. I would also like to thank Prof. Johan Lillius for introducing me to this project and involving me in it.

Lucas Fransman

June 1, 2023

Table of Contents

1 Introduction

1.1 Background - operations in the maritime environment

1.1.1 The importance of the maritime industry

The maritime transport industry takes care of most of the international commerce, which has been a major reason for the rise of living standards in the western economic areas by making affordable global commerce possible. Most products used today in our day-to-day life have been transported by sea in container ships, tankers, and bulk carriers, both as raw material and as complete products. Due to international trade and commerce being completely dependent on it, maritime shipping serves as a critical component for the global economy [1, 2].

Based on the United Nations Conference on Trade and Development's (UNCTAD) review of maritime transport, 11 billion tons of cargo was shipped by maritime means in the year 2021. This amount represents more than 80 percent of the total amount of cargo shipped internationally [3].

The only form of transportation for goods which is more affordable than transportation by ships is pipelines. This, naturally, entails numerous restrictions, as it limits the form of goods which can be transported to liquids. Also, changing the route is not feasible and, in many cases, impossible. This makes pipelines financially feasible only in very specific conditions, while ships are much more versatile regarding the above-mentioned factors. Ships can carry any sort of goods which can fit on deck, most commonly inside shipping containers [4].

Other forms of transportation which ships can be compared to are railways, trucks, and aircrafts. The most interesting comparison is with aircrafts, as it is the only form of transportation which can match the unique critical feature of ships regarding international trade, i.e. the capability to transport goods over large bodies of water. Upon further inspection, it becomes rather clear that these forms of transport serve different needs. Aircrafts have very limited capacity, which also makes it a very expensive form of transportation, but as far as speed is concerned it has a clear

advantage. Aircrafts, therefore, attract cargo with higher value and less volume or when fast shipping is required. In other cases, the affordability of ships will be more profitable [4].

With the current trends of globalisation, it is extremely important that we have an agile, safe, and reliable maritime transport industry to support the transportation needs required to keep it up and support further possible economic growth. To do this, the maritime industry should take advantage of new technologies that are constantly being developed, for example, the internet of things (IoT), cloud and edge computing, digital twins, simulation, and machine learning [1].

1.1.2 Digitalisation in the maritime industry

With the amount of networking today and the wide array of pre-made interfaces available, a large number of different applications in the maritime industry have the potential for digitalisation. Traffic, port logistics, and just-in-time shipping will see a considerable change as technology will further be taken into use within the maritime industry. Maritime operations will largely benefit from technologies derived from current fast developing fields like big data, sensors, and networking technologies [2].

Listed here are some technologies which are paving the way for the modern and further digitalised version of the ship and which are discussed throughout this thesis:

- *Internet of Things (IoT).* The term Internet of Things was first used in 1999 at MIT by Kevin Ashton.. The core idea is to take advantage of the already existing internet architecture through connecting everyday objects, or things, not traditionally linked with computers and connect them to the network [5]. In the maritime industry, it is already common to see equipment that has some form of digital measuring device or even control device which can be connected to some network either via cable or wirelessly. Also, in the bridge of a significantly sized ship wide arrays of digital screens are usually visualising data in some form [2].

- *Edge computing.* Edge computing, on a high level, involves bringing compu-

tation closer to the "edge", meaning where data is collected, either through sensors or some form of input [6]. With the amount of possible data to be collected on ships in combination with an environment where internet connection can be limited, a potential solution would be to simply bypass that issue by doing as much computation as possible on the ship itself, as opposed to sending large amounts of data over a network to some cloud server for computation [7].

- *Machine learning.* Machine learning is the use of computational algorithms with the characteristic of being able to learn from data. In the maritime industry with the data being collected machine learning could be used to predict whenever maintenance is needed, voyage optimisation, controlling of freight rates, managing energy efficiency, etc. Also, the concept of autonomous and semi-autonomous vessels relies on machine learning for tasks such as collision avoidance [2, 8, 9].

It is not as if the maritime industry has chosen so far to ignore the technological developments going on. Large amounts of data are already being collected from ships through different arrays of sensors placed around the ship. Also, tools and equipment placed on ships have numerous sensors and use data to some extent to improve design and operation, for instance, cranes and motors which are usually manufactured by third-party companies. The issue with the shipping industry, though, lies in the fact that most of this collected data remains unused. Often the effectiveness of one's data collection cannot be measured by the amount of data collected, but rather how that data is formatted and taken into use to improve design and operation. Raw sensor data does not provide much value. To enable the use of, for example, machine learning algorithms the data must be preprocessed and categorised in some meaningful way to provide any results of use [1].

One field that has sparked particular interest within the maritime industry is autonomous shipping, which promotes the idea that ships could function with as little as possible to no humans on board or make it possible to do human-required tasks from a remote location. The lucrativeness of this field naturally comes from

reducing costs in the form of reducing the number of humans having to be paid salaries [8]. This type of technology has seen ongoing development and for quite some time already, there have been digital systems in place to assist the human crew on board. These systems have been successful in minimising navigational errors [10].

The current cutting-edge research in this field focuses on making ships fully autonomous, that is having no physical human crew on board at all. There are large companies in the maritime industry which receive funding from the European Commission through the Autoship project, Kongsberg and Rolls-Royce being two of these companies. The project aims, within the time frame of two and a half years, at constructing two fully autonomous ships with a possibility for remote control by humans, but with no need for any humans on board. Tests will be conducted on the sea routes from the Baltic corridor to some of the largest ports within the European Union [8].

1.2 Digital twins in the maritime environment

1.2.1 The origin and motivation for digital twins

A digital twin refers to a digital counterpart of some physical entity, modeling that physical entity in some chosen level of detail in a digital environment. In a more academic setting, the concept of a digital twin is usually split into three distinct parts: the physical entity, its digital counterpart, and the data connections in between the two. As previously stated, one of the main issues in the digitalisation of the maritime industry is not necessarily that there is not enough data collected, but instead that there is no way to use all the collected data in a way that provides any form of value. One of the key principles in the concept of the digital twin is to use computation in the digital environment on the digital counterpart with the goal to improve its physical counterpart. Here, we can use the collected data in a meaningful way to take advantage of machine learning algorithms and apply them on the digital twin [11].

The concept of the digital twin has its origin in the US organisation the National Aeronautics and Space Administration (NASA). Michael Greaves and John Vickers,

working at NASA in 2003, saw the potential for this technology through observing the inefficiencies in how much data from physical products were collected manually through pen and paper methodologies. At that time, digital representations of physical products were not well developed and did not see much practical use. Grieves and Vickers held the belief that a digital counterpart of a product would not only be beneficial but could function as the basis for the whole product life-cycle management, which led to them coining the term digital twin [11].

1.2.2 Examples of digital twins in the maritime industry

The term digital twin did not immediately catch on in the maritime industry after being coined in NASA during the first years of the 21st century. It took approximately another decade for the topic of digital twin ships to appear in academia in a significant way. Only in the mid-2010s did the popularity of the topic start to increase. Published academic papers on the concept of digital twins can be grouped into two distinct categories. The first category refers to decision support for ship operations, with the focus on using operational data (collected from the wide array of sensors available to achieve real time condition monitoring) and to use the collected real data and use it to enhance possible simulation models [12].

In 2019, Coraddu et al. used neural networks to estimate speed loss which happens during marine fouling. Marine fouling is the process of accumulation of organisms on submerged surfaces that occurs when organisms attach themselves to submerged objects, in this case parts of the ship that are submerged during operation. A neural network used the collected operational data from the physical ship to generate its estimations for speed loss. This method was further compared to the ISO standardised method to estimate this sort of speed loss and showed better results. However, due to how deep learning algorithms work, there is a considerable need for more data than what the ISO method requires [13].

In 2018, Schirmann et al. used the concept of the digital twin to estimate structural damage caused to the twin's physical counterpart by sea waves during its operation. By using weather forecasts for the route of the ship, they were able to

expose the conditions onto the digital twin before the physical ship was en route and, therefore, able to assess the potential damages before even taking off [14].

In the second category, the digital twin is used for system integration testing and for training personnel. Being able to try out systems in a digital environment can save financial costs instead of discovering potential problems in the later stages of development where a system would already be integrated on the physical ship. Naturally, training personnel in a digital environment rather than on a physical asset also reduces any potential financial, personnel safety, and other risks in case anything would go wrong [13].

1.2.3 Content and usage of digital twins

Ships are very complex systems. Therefore, digital twins in the maritime setting have stayed quite focused on a well-specified problem to avoid the complexity of modeling a whole ship with its large number of systems and connections among them. This makes the models more constrained, but the development also becomes much more manageable. There are some commercial solutions from specialised software vendors that offer detailed complete ship models, but the features for simulation and monitoring during operation are still at a quite primitive stage. Data used in the context of digital twins can be mapped into three different categories: asset representation, behavioural models, and measured data. The category measured data can be divided into two subcategories: data measured from the asset and data measured from its operational context [12].

1.2.4 Asset representation

A digital representation of assets, such as ships, are usually done with some form of computer-aided design (CAD) software. CAD software is a critical tool used at the earliest stages during the ship's lifetime where most work is performed during the design phase before construction has started. With CAD software required blueprints are made and three-dimensional digital models are created, which makes it easy to detect potential construction issues before the construction phase has started, like

for example pipe collisions. These kinds of special purpose CAD solutions usually allow a high degree of customisability to its users through options like scripting or database customisation, which can be used to further automate internal tasks, such as generating product lists from the model. These can be used to make the process of purchasing the required material and equipment more streamlined [12].

In the ship design process, components like the hull, structure, and outfitting are usually modelled during the design phase into CAD files. These CAD files, in combination with models provided by relevant third-party vendors, build the foundation of a ship's digital twin. To complete the digital twin, metadata such as weight distributions and material characteristics might be required depending on the requirements set on the level of accuracy desired [12].

1.2.5 Behaviour models

Behaviour models create the bridges in between the digital representation of an asset and the physical asset which it represents. Behaviour models are implemented based on what the specified purpose of the digital twin at hand is. The behaviour model will execute some form of simulation which is designed for its desired purpose. The simulation can actively utilise live data gathered from the physical asset using potential sensor networks installed on it. Alternatively, it can use static data to study the condition and performance of the physical asset based on operations carried out in the past [12].

Depending on what the use cases and the goals of the simulation are, there are several methodologies to create these behaviour models. Some of the current common methodologies involve methods based on physics and statistics. Also, with the current developments in sensor technology and in the fields of big data, it is also possible to use more data intense models based on machine learning or even deep learning to create more effective predictive models by leveraging the sheer amount of the possible data that can be collected. Methods such as edge computing can be leveraged to avoid the potential restrictions posed by telecommunication when it comes to communicating large amounts of data [12].

1.2.6 Measured data

Measured data means, in this context, the state of an asset at any given time during its operation. This requires tools to quantify the state of the asset during its operation; these can be sensors or reports which describe the asset. With the developments in sensor technologies and remote control technologies, the core value in digital twins will be the ability to extract insight from the vast amount of data collected [12].

To optimise the value of the collected data the way sensor networks are constructed is essential. There are many important things to consider, such as the type of data to be collected and its required accuracy, as well as the frequency of the data collection. These factors will affect what type of sensors are needed. Naturally, the placement of sensors on the physical asset and their integration into different systems that exist on the asset requires planning. As mentioned previously, there is not much value in raw data, so there is still much work concerning post-processing and extracting value from data using, for example, some of the previously listed statistical methods [12].

1.2.7 Standardised approaches toward digital twin vessels

The digital management of a ship's life cycle has traditionally revolved around highly specialised tools, either developed internally or acquired through different third-party developers. One of the core issues standing against the principles of the digital twin and concepts like co-simulation is that interoperability has never been in focus within the development of these tools. Third-party software vendors maintain a view that data formats are a part of their competitive advantage and should, therefore, remain proprietary to protect one's intellectual property. This makes interoperability near impossible in some cases, enticing a need for a translation layer in between tools or everyone committing to a common standard. Standardisation can be difficult to implement, as software vendors prefer their relatively locked-in users to be dependent on their solutions, and not able to transfer easily to a competitor's tool, because it can be perceived as a risk for one's business. This does not even take into account

the investments needed to launch and finish a collaborative and large-scale project [12].

While the maritime industry needs very highly specialised tools that require a high degree of expertise and effort from the software vendor's perspective, the maritime industry represents a relatively small subset of the industries leveraging similar CAD or computer-aided engineering (CAE) tools. With the maritime industry's traditional way of focusing on individual projects, largely due to the sheer size and investment needed for a single project, a large part of the data management solutions is specific to one project and cannot easily be transferred into future endeavours. When it comes to comparing different ship yards and ship design offices to each other, the situation is even worse [12].

1.2.8 SFI Coding and Classification System

With the lack of any well-established ship models within the maritime industry, the alternatives for representing product data are small. There have been initiatives to categorise ship data. One notable example is the SFI group with their SFI Coding and Classification System. It is a standard which is used internationally to provide a functional division of both technical and financial ship information. The motivation behind the development of the system was to solve the frequent difficulties in data exchange both internally inside organisations and externally in between organisations [12].

The primary function of the system is to catalogue and recognise blueprints and specifications, as well as manage and account for components, tasks, and materials. The system operates on a hierarchical numerical label system that organises components based on their functional role in accordance with a functional perspective of the ship. This can be seen in Figure 1 [12].

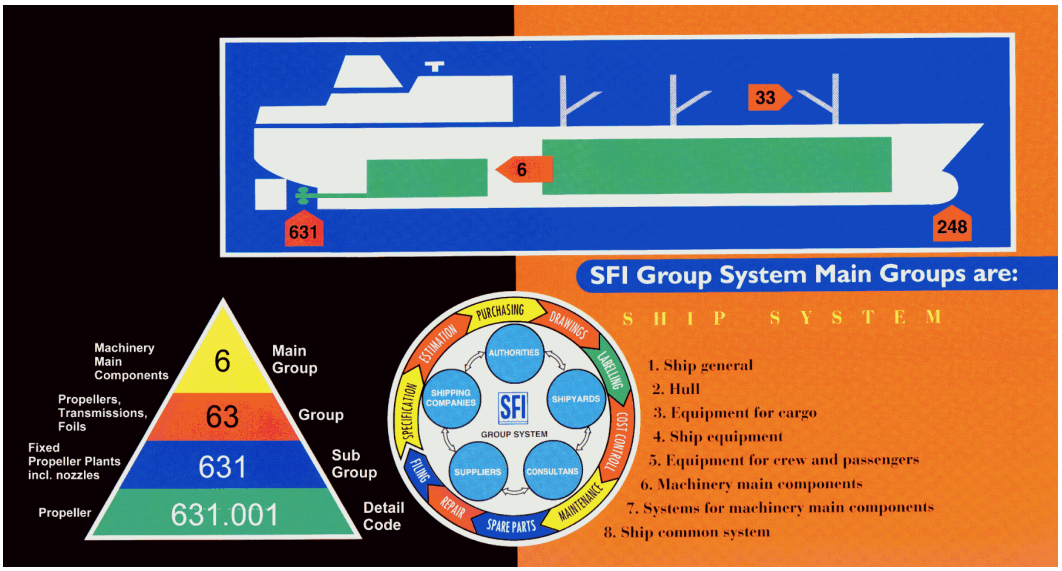

Figure 1: The SFI Coding and Classification System [12].

1.2.9 The Standard for the Exchange of Product Model Data

The Standard for the Exchange of Product Model Data (STEP), or the ISO 10303, provides a standardised way to represent and exchange product data. The goal is to be able to easily exchange data from one CAD or CAE tool into another one, therefore making a number of tools available during a project. With the array of highly specialised tools, naturally some tools might have better functionalities in certain areas while some may be lacking in some areas relative to other tools. With a universal standard for the data format, it would technically be possible to utilise the tools best suited for a particular area without too much difficulty in exporting and importing data between them. Also, it can be argued that the lack of such a standard hinders innovation in the field of CAE, as the financial investment in switching from one framework to another without a common standard is large, as all models would have to be recreated in the new framework, requiring substantial training costs. As a consequence, for a new framework to break into the market, it requires a substantial benefit for an organisation to consider taking on the costs in switching software suites, since small incremental innovations might not suffice [12].

The initiation of STEP took place in 1984, but it was not until ten years later, in 1994, that it was officially launched. Subsequently, several application protocols

for ship data were introduced, encompassing areas like arrangements, piping, structures, and mechanical systems. The STEP standard employs an object-oriented methodology, echoing the object-oriented model well-known in the software industry, incorporating concepts such as properties and inheritance [12].

STEP has not reached its goal of being an industry-wide accepted standard. This was mainly due to the lack of interest from software vendors in this area to take on the work to comply with any such standard. Their reluctance was due to some of the previously mentioned reasons, such as the immense collaborative effort it would require, the large financial investments required without any clear indications of seeing a profitable return on the investment, and with the view that proprietary data formats serve as a competitive edge. This is not the only reason for its lack of widespread use, its lack of technical performance being reported also. Due to individual software solutions promoted by different vendors, all their existing data models could not be completely reflected into one standard, which led to some data being left out in the exchange process between tools [12].

1.3 Introduction to co-simulation

1.3.1 Simulation

Through simulation, we can model the behaviour of a dynamical system, i.e., a real-world physical or computer system. Breaking the system into smaller parts modelling the key functions within a system is achieved by employing a set of different states, which can be reached from one or multiple other states from the set. These transitions between states are defined by a set of functions called evolution rules [15].

A trivial example of a simple physical system to demonstrate these terms is the pedestrian traffic control system one can encounter, depending on geographic location. In areas, mainly urban, where a large part of the motorised traffic coexists with a large pedestrian crowd, locations for crossing the road for pedestrians are placed on defined locations and are commonly bundled with a traffic light system directed at pedestrians. This system might contain states such as green, red, and

no light. The evolution rules might, in their simplest form, be that after reaching the state green light, the state is maintained for 30 seconds before transitioning into the red light state, and from the red light state, it transitions into the green light state after another 30 seconds. To save energy, we might define that a no-light state will be set after 10 PM to 6 AM during times when traffic, both motorised and pedestrian, is a mere fraction of its prime-time state. In reality, these systems are more complicated as they communicate with the traffic light systems directed at the motorised traffic. There might be systems that affect the evolutionary functions in the form of buttons or cameras which allow data such as the number of pedestrians trying to cross at any given moment to be taken into account in real time.

The crucial difference between a simulation or a dynamic system and a static model is the addition of the time dimension and an external environment affecting our simulation. In digital simulations, the dimension of time is defined as an array of discrete time steps which contains a different state of the physical asset [15].

Given a model m, a variable $t \in T$ is commonly referred to as simulated time. This simulated time is not necessarily synchronised with real time, that we live under in the physical world, but is instead multiplied by some factor greater than zero, and commonly less than one. Real time is referred to as wall clock time, $wt \in WcT$. This is a critical component in simulation, as it allows us to fully utilise the computational power of current-day computers allowing us to execute multiple time unit simulations in a comparatively short amount of wall clock time depending on the computational requirements of our model. When running a simulation over the range $[0, t]$ simulated time, a computer will take wt units of wall clock time which, in turn, depends on t. Therefore, wt can be used to measure the run time performance of a simulation [15].

1.3.2 The case for simulation

When considering complex systems, for example, a ship is built by combining smaller systems into one complete system. These subsystems are built by different groups with different expertise unique to the subsystem they are responsible for. There is rarely any such group of people who have a deep understanding of how all of these

systems work, how they communicate with each other, and how they will react when exposed to different environments [16].

The problem is that a deep understanding of a subsystem will not reveal potentially surprising outcomes in the larger global system. This is where simulation emerges and can help, if designed correctly, to gain a new understanding of the system, which might not be intuitive through the original knowledge of the system. Biological systems provide some extreme examples of such cases where forests under pollution loads show how even a tiny perpetual stressor can lead to a sudden collapse in the system, which could not have been predicted through the study of individual components in isolation [16].

Simulation provides a tool to test different hypotheses quickly and to understand and test different parts of a complex system that can contain subsystems with a high degree of expertise. With simulation, we can test how the system reacts in different environments without the limitation of physical time when executing many scenarios. Formulating competing hypotheses and introducing them into a simulation model is, in most cases, relatively trivial [16].

Digitalisation, developments in connected systems, networking, sensors, and new ways of thinking about computing in the form of IoT are shaping the future of many industries, the maritime industry included. As stated many times in this chapter, the amount of data collected in many settings, ships included, has grown exponentially. With the raw data not necessarily providing a direct benefit in its initially recorded state, tools such as simulation provide a lucrative method to leverage all the abundant data collected in a way that potentially can improve design, construction, and operational phases, in turn increasing the output of companies which choose to take advantage of it [17].

The maritime industry is very competitive and dynamic, which ensures that the industry operators must constantly invest in evolving their methodologies further to stay relevant. In many industries, data has been a critical component in their digital transformations, helping organisations to identify new opportunities, improve efficiency, and make more pragmatic decisions. With this recorded success, it makes

sense for the maritime industry to look towards their relatively untapped data sources to optimise different phases in design, production, and operation, as optimisation and innovation are the key to staying relevant in a competitive and dynamic industry [17].

Even though the use of data in the maritime industry has been relatively limited, there are recorded cases of success. Especially regarding fuel efficiency, fuel being one of the significant operational costs within the transportation industry bringing down profits, there have been several examples of enabling incremental improvement towards fuel efficiency through the use of data in a ship's design process [17].

Simulation goes beyond traditional methods of digital prototyping techniques which have been used in ship design. It allows future designs to take advantage and to be optimised through recorded environmental and operational conditions. Design, data, and machine learning techniques, in combination with already existing CAD solutions, offer an effective way to reduce costs and increase output based on real data [17].

1.3.3 What is co-simulation?

With the increase of digitalisation onboard ships, the complexity of the ship infrastructure rises with it to a certain degree. Digitalisation introduces new fields of expertise required to construct the global system necessary to represent a modern ship. Digital technologies such as sensors, software, and networking are starting to play a more significant role, and often the desire is that these digital subsystems communicate with the more traditional subsystems within a ship, further exacerbating complexity. Therefore, the design process must become more and more distributed, as responsibility for different subsystems must be distributed to teams with the right expertise. Also, external providers possessing the required expertise will be increasingly needed. For many companies, especially smaller ones, it might not be feasible to employ certain types of expertise on a full-time basis. While distributing expertise into smaller and more focused entities and combining them to bring on a full-scale global system in the form of a ship certainly has numerous benefits, but it also pro-

vides its set of challenges. Relevant for us is the knowledge of how the full-scale system works, the details within the subsystems, and the communication between them all required to execute a full-scale simulation of the global system [15].

As far as traditional simulation and modelling are concerned this distributed environment poses some challenges in the context of the exchange and integration of all distributed partial solutions. For example, different tools might have been used where there is no standard for common data exchange between them, which is often the case for these types of vertical software tools. In the case of external providers, there might be issues relating to intellectual property that create challenges in the data exchange process or hinder it completely [15].

Co-simulation proposes a solution for these issues. With co-simulation, the simulation of the global system is built by connecting modular sub-simulators into one global simulator. The sub-simulators function as black boxes; the people creating these sub-simulators need no expertise in any other field than their own, as a sub-simulator has no idea of the internals of some other sub-simulator. The only part that is exposed is an interface of input and output variables which can be fed in or forwarded out by the sub-simulator. Also, it solves the issue of intellectual property as the internal logic does not have to be revealed. Instead, the sub-simulator can be delivered as a package of compiled binaries that executes the behaviour of the subsystem in question and ensures the privacy of sensitive data [15].

To have users interact with and operate these physical complex systems, some form of training is often needed. Having a digital environment where this training can be completed can be beneficial and, in some cases, required. The reasons for these benefits reduce into some form of reducing costs or increasing safety. The main challenge is the construction of this environment, as it must reach a certain degree of granularity and accuracy to provide an effective form of training. This is where the inherent modularity of co-simulation can reduce the amount of work, as we can reuse the digital sub-simulators already created during the ship's design phase to create these accurate training environments [15].

These co-simulations can also be exploited during the operation phase of the ship,

for example, to improve the scheduling of maintenance on the ship by using collected data during operation to predict potential flaws or malfunctions before they occur. As systems become more complex, the need for co-simulation scenarios which are larger, hierarchical, heterogeneous, accurate, and IP protected will increase. These requirements are difficult to satisfy using traditional all-in-one simulation but are some of the core issues navigating the development of co-simulation [15].

1.3.4 Co-simulation internals

Co-simulation, though not explicitly implementing distributed computing, lends itself naturally to it. As a co-simulation scenario consists of n completely independent sub-simulators, nothing hinders the co-simulation from running on a network consisting of up to $n + 1$ individual computers, where there could even be a separate computer for the co-simulation master algorithm, if so desired. Simulations of complex systems can be very computationally demanding, and as the underlying systems become more complex, the computational demands rise. Similarly, when more granularity is needed, whether in the digital projection of the system itself or the surrounding digital environment, more computation is needed to express a higher level of detail. This highlights a need for distributed simulations [18].

Co-simulation, like many other forms of digital simulations, is a discrete-step simulation. This means that the state of the simulation is updated only on defined discrete steps, usually through defining a step size in the unit of time. As the step size moves closer to zero, the closer we are to a continuous simulation. Naturally, the smaller the step size, the higher accuracy we can expect, and with it, the need for computation increases. In co-simulation, these discrete steps are called communication points, which establish the communication between sub-simulators through pre-defined connections. Before the communication occurs within the discrete step, the sub-simulator's individual solver strategy, i.e., the function applied to the input variables to produce the outputs, is applied. Each sub-simulator may have its own step size defined, which does not have to correspond to the other sub-simulator step size within the co-simulation, but usually, some restrictions are set, for example, the

need for the step sizes to be within a factor of each other. Furthermore, a global time step is established at the co-simulation master algorithm level. Usually, this global time step is longer than the time steps of the sub-simulators. The co-simulation master algorithm is responsible for managing all data interactions between the sub-simulators and ensuring they operate in sync [18].

Skjong presents the following example in his PhD thesis [18] in the form of two subsystems defined as:

$$\dot{x_i} = f_i(x_i, u_i, \tau_{c,i}) \tag{1}$$

$$y_i = h_i(x_i, u_i, \tau_{c,i}) \tag{2}$$

In this scenario, $x_i \in \mathbb{R}^n$ signifies a vector which illustrates the state of subsystem i, $u_i \in \mathbb{R}^m$ stands for the input vector from all the systems linked to subsystem i, and $\tau_{c,i} \in \mathbb{R}^p$ represents the control vector input for subsystem i. f_i is the vector of differential functions associated with subsystem i, $y_i \in \mathbb{R}^r$ is the corresponding output vector, and h_i is the vector for the output mapping function relevant to subsystem i [18]:

$$\mathbb{R}^n = \{(x_1, ..., x_n) : x_j \in \mathbb{R} \text{ for } j = 1, ..., n\} \tag{3}$$

$$\mathbb{R} = \mathbb{R}^1 = \{(x_1) : x_1 \in \mathbb{R}\} \text{ The set of real numbers.} \tag{4}$$

$$\mathbb{R}^2 = \{(x_1, x_2) : x_j \in \mathbb{R} \text{ for } j = 1, 2\} \text{ The set of all pairs of real numbers.} \tag{5}$$

The two subsystems are linked in a co-simulation such that u_1 is equivalent to y_2 and u_2 matches y_1. The values of u_1 and u_2 remain fixed between global time steps T_d, and are only updated at each distinct time step t_i. As depicted in Figure 2, a co-simulation master algorithm controls the data exchange in the subsequent co-simulation. It also oversees the overall simulation process and the global system time for the co-simulation [18].

In his PhD thesis [18], Skjong also presents a simple mathematical representation of a co-simulation implementation, shown in Algorithm 1. Many parts are left nondeterministic, such as the internal logic of the subsystems and the data exchange procedure in between them, only when these actions are performed relative to everything else is shown. If both subsystems were developed using the same tools, operated under the same operating system, and executed on the same computer and processor, establishing the connection would be relatively straightforward. In such a case, a co-simulation master algorithm could be readily adjusted to manage all individual data connections and execute the entire co-simulation process. Although this is not done in a generic fashion, it arguably sacrifices one of the more lucrative possibilities of co-simulation, reuse, and modularity [18]. To solve this, we need standardisation, which will be covered more in-depth in Section 3.4 in the form of the FMI standard [19].

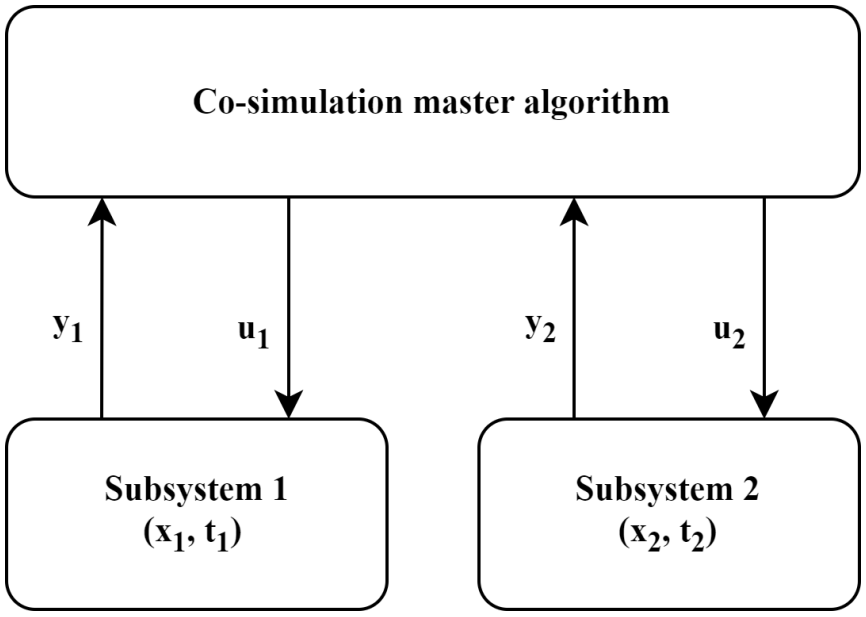

Figure 2: Two subsystems in a co-simulation connected to each other [18].

1.3.5 Recorded use and challenges

In [20], Liberatore and Al-Hammouri propose a system for a smart grid communication network, *PowerNet*. They describe it as a system of interoperable, heterogeneous

Algorithm 1 Solution procedure for a co-simulation containing two general subsystems [18].

procedure CoSimulation
  $initialise()$
  while $t \leq t_{stop}$ **do** ▷ Solver loop
    $t = t + T_d$ ▷ T_d is the time in between communication points
    $u_1 = y_2$ ▷ Updating subsystem inputs
    $u_2 = y_1$
    while $t_1 < t$ **do** ▷ Solving subsystem 1 until next communication point
      $\dot{x_1} = f_1(x_1, u_1, \tau_{c,1}, t_1)$ ▷ Calculate rate
      $[x_1, \Delta t_1] = Solve(x_1, x_1, u_1, \tau_{c,1}, t_1)$ ▷ Solve for the next local time step
      $y_1 = h_1(x_1, u_1, \tau_{c,1})$ ▷ Update subsystem output
      $t_1 = t_1 + \Delta t_1$ ▷ Update local time
    end while
    while $t_2 < t$ **do** ▷ Solving subsystem 2 until next communication point
      $\dot{x_2} = f_2(x_2, u_2, \tau_{c,2}, t_2)$ ▷ Calculate rate
      $[x_2, \Delta t_2] = Solve(x_2, x_2, u_2, \tau_{c,2}, t_2)$ ▷ Solve for the next local time step
      $y_2 = h_2(x_2, u_2, \tau_{c,2})$ ▷ Update subsystem output
      $t_2 = t_2 + \Delta t_2$ ▷ Update local time
    end while
    $Collect(t, x_1, x_2...)$ ▷ Store results
  end while
  $plot(t, x_1, x_2...)$ ▷ Post-processing, e.g. plotting data
end procedure

networks that offer sufficient real time speed, security, and reliability while utilising currently available hardware and software solutions.

They use co-simulation to combine a power grid simulator based on Modelica with the Network Simulator 2 (ns-2) [20]. This allowed them to evaluate the interactions between the communication protocols and the physical properties of the power grid. Co-simulation could be used to test different scenarios and conditions, such as different communication protocols, network failures, and congestion, and to analyse the performance of the smart grid under these conditions, providing the means for further optimisation based on data and observations [20].

Some of the challenges they encountered were linked to the synchronisation between the simulators. With multiple simulators running separate processes and communication between them, it is crucial that the simulated times of each simulator are synchronised to avoid erroneous output [20].

In [21], Trčka et al. discuss the difficulties in addressing the persistent requirement for reduced energy consumption in contemporary buildings. They recognise that heating, ventilation, and air-conditioning (HVAC) systems account for between 10–60% of a building's total energy use, making them an obvious focus for advancements and enhancements. Furthermore, the extended lifespan of modern buildings underscores the necessity of making sound decisions rooted in concrete data and testing. This is because the repercussions of non-optimal choices will surface over a much longer duration.

In [21], different co-simulation strategies were examined and the study was expanded to encompass existing standalone Building Performance Simulation (BPS) tools. A noted advantage of co-simulation was the capability to link pre-existing domain-specific tools for a comprehensive simulation, while maintaining their individual attributes. This, in turn, facilitated the utilisation of distinct tools that cater to specific domains and the creation of control algorithms using well-known tools like MATLAB/Simulink, offering beneficial toolboxes [21]. The co-simulation prototype developed was built on two cutting-edge BPS tools: EnergyPlus and TRNSYS. With these two simulators, varied strategies such as loose and tight coupling were instituted. The co-simulation prototypes were put to the test in a case study, where they could achieve fully integrated design analysis, something that wouldn't have been feasible by any of the mentioned BPS tools individually [21].

2 Co-simulation in the maritime environment

2.1 Overview

The idea of digital twins, precise models that represent their physical equivalents, shows promise for the upcoming generation of modern ships. The implementation of digital twins in the maritime sector allows for data analysis and surveillance of maritime systems. This can potentially avert issues before they even arise and facilitate future planning through simulations. Nonetheless, various obstacles need to be addressed to actualise this concept [22].

One of these challenges is the integration of heterogeneous systems and hardware, making it difficult to create a centralised or monolithic digital twin. Another challenge is the computational requirements to run these simulations. Co-simulation can be utilised to address these challenges. Co-simulation allows for the modeling of different subsystems independently, but the simulation of these subsystems together allows for a more efficient and effective use of resources [22].

The maritime industry is also setting constantly higher and higher requirements when it comes to issues such as lowering fuel consumption and reducing emissions, especially as the climate goals are becoming more ambitious, which will lead to financial incentives. Issues of operational safety and operational weather dependencies are other vital topics [18].

Norway is known to have experts in the field, when it comes to taking advantage of co-simulation in the maritime industry and is known for being among the top experts in the field, especially when it comes to complex maritime operations are facing challenges in staying financially competitive due to relatively high wage costs, and research and development spending. Therefore, the Norwegian maritime sector has had to invest substantial resources to preserve its technological edge and reduce expenses. This has been achieved through an intensified emphasis on developing technologies and methodologies, aimed at optimising work processes and decreasing costs [18].

2.2 Recorded use cases

Co-simulation has been used to some extent in the maritime industry already. Skjong et al. [23] used simulation to conceptually represent the use of a hybrid maritime propulsion system for an international freight vessel which is a rare occurrence as these kinds of systems have mainly seen use on smaller ferries [23].

Their simulation outcomes indicated that incorporating an energy storage device enabled the use of the shaft generator as an active contributor to propulsion without negatively impacting the generator load. Moreover, the efficiency of the system was found to match that of a conventional system, even without necessitating

optimisation of the plant [23].

Both an all-in-one fashion simulation and a co-simulation were used to compare the two different methodologies. Co-simulation exhibited results in agreement with the more traditional all-in-one simulation while also providing a significant improvement in computational speed by approximately halving the required run time in all tested cases [23].

In a different study [24], Skjong and Eilif Pedersen carried out a co-simulation case study on a maritime offshore surface vessel operating in Dynamic Positioning (DP). In this study, the DP controller was on a physical Arduino microcontroller [24].

The vessel model included all the different mechanical and environmental requirements for the simulation, such as propulsors and environmental forces. A communication FMU is used to connect the Arduino microcontroller to the co-simulation. The communication FMU establishes a connection with the microcontroller by channeling signals between the microcontroller and the co-simulation at predetermined moments in time. The communication FMU also has the task to synchronise the microcontroller with the co-simulation, with the microcontroller. The Nonlinear Passive Observer (NLPO), with the task of filtering away all oscillatory vessel motions which did not result in a drift, was also implemented as a separate FMU. In [24] Figure 3 the constructed co-simulation is shown [24].

Using co-simulation makes it possible to distribute the model to multiple cores, making the simulation run more efficiently than just using a single core. It also naturally made the model more modular by having sub-models that can easily be reused in other simulation contexts. Also, in this form of Hardware-In-the-Loop (HIL) type simulation, the hardware components are easily interchangeable as the sub-models do not recognise anything else than their inputs, outputs, and internal logic. The hardware components essentially function as an FMU in the way that they can function as individual components and can easily be switched out by some other hardware component [24].

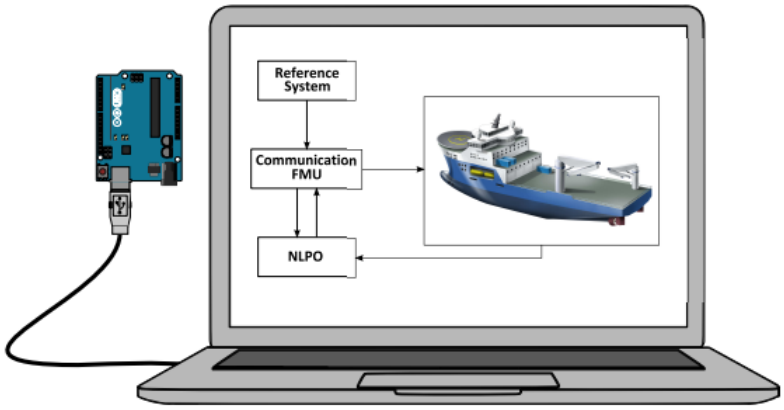

Figure 3: Conducting a co-simulation of an offshore maritime vessel in Dynamic Positioning operation utilising an Arduino micro-controller as the DP controller [24].

2.3 Benefits

Conventionally, the ship design process has been recognised for its distinctiveness, without any extensive industry standards. This process also grapples with severe time and resource limitations, and many variables in the design process are determined early on, relying predominantly on experience instead of any method grounded in science or engineering [22].

With the rise of use in technologies within ships through innovations such as autonomous shipping, the verification needs are steadily increasing. Traditionally models and simulations have been done in an all-in-one fashion, with models and simulations very concentrated on the current project in question and dependencies on the tools being used. This naturally limits re-usability or hinders it completely, following higher investment costs as new solutions must be developed for each project [18].

Co-simulation, through its design, promotes modularity and reuse. Through its black box design principle, a sub-simulator that can be made as atomic and generalised as desired holds no dependencies on any simulation tools or other possible sub-simulators and can, therefore, easily be reused in future projects. The fact that the sub-simulators are built upon well-established standards such as FMI makes it easier to create standardised tools when it comes to executing the co-simulations

through a co-simulation master algorithm and will enable cooperation between different organisations and teams when it comes to ship design process, which due to its size could not be handled by a singular team nor organisation [18].

Simulations in a maritime context can quickly get large and computationally expensive due to the vast number of systems and variables a ship depends on. With all-in-one solutions, we are limited to running our simulation in a more constrained environment. The modular co-simulation design and the fact that sub-simulators are independent of each other makes running computation hardware in parallel much more feasible [22]. This has been recorded in some of the use cases presented in the last section, where the computation time had been significantly decreased and where a comparison was made with an all-in-one solution, the computation time had been approximately halved by using a co-simulation based solution [23].

2.4 Open Simulation Platform – general description

The Open Simulation Platform (OSP) is the co-simulation platform that is used within this thesis, the primary motivation being its sole focus on the maritime industry and its support by major players in the industry. The OSP consists of a set of tools for running simulations, verifying models, and open-source code libraries for running co-simulations. The OSP was founded in 2018 by the organisations: DNV GL, Kongsberg Maritime, NTNU, and SINTEF Ocean and is constantly being joined by new partners [25]. The OSP is covered in depth in its own section; see Section 3. In the following part, other similar platforms will be discussed.

2.5 Other similar platforms

2.5.1 Coral

SINTEF developed coral as a part of the Virtual Prototyping of Maritime Systems and Operations (ViProMa) project. It is a software package for running co-simulations. It is built from scratch with support for the FMI standard, similar to the OSP; see section 3.4. Coral uses a master/worker structure in which the

sub-simulators function as workers, and the Coral simulation API is the master connecting them. The primary responsibilities which Coral takes care of are the communication and synchronisation between the sub-simulators, making sure the data is transferred according to the defined connections and that everything is synchronised through signaling to the sub-simulators when they can execute the following time step in the simulation [18].

Coral supports the distribution of the simulation using a network of computers. This is enabled through a server program which makes sure all the sub-simulators are properly started and initialised on their computers before starting the distributed simulation. Coral is implemented in the form of a C++ library which can be used to execute co-simulations in any application as desired through the library interface. The library is entirely open-source and released under a permissive open-source license. A set of command line tools to run co-simulations using configuration files is also provided [18].

2.5.2 SystemC

SystemC is a system-level modeling language used to design hardware and software systems. It is based on the C++ programming language and includes a set of libraries, data types, and constructs that enable the modeling of complex systems. SystemC is often used in the design and verification of electronic systems, such as digital circuits, processors, and embedded systems. It allows designers to create models of these systems at various levels of abstraction, from high-level behavioral models to detailed hardware descriptions. One of the key features of SystemC is its ability to model concurrent and parallel behavior using processes and threads. It also provides support for simulation and verification, including the ability to model timing and synchronisation between components [26].

A SystemC model consists of a set of modules that represent a system's hardware and software components and the connections between them. A module is implemented as a C++ class containing data members and methods that define the module's behavior. The module, as any C++ class, can be instantiated multiple

times to represent multiple instances of the same component, such as multiple processors in a system. Modules can communicate with each other using channels which are also implemented as C++ classes. Channels provide a way for modules to send and receive messages to and from each other and can be used to model both synchronous and asynchronous communication between components [26]. A SystemC model is essentially a co-simulation with modules representing sub-simulators and channels representing the connections in between them.

2.5.3 Simulink

Simulink, a product of MathWorks, is a visual programming platform used for creating models, executing simulations, and analysing dynamic systems. It finds application in areas like engineering, control systems, and signal processing. Simulink facilitates the construction of intricate systems through block diagrams, symbolising diverse components and their interconnections [27].

Simulink can perform multi-domain simulations, where users can model and simulate systems involving different physical domains, such as mechanical, electrical, and hydraulic systems. This flexibility is achieved by integrating Simulink with a wide array of specialised toolboxes, such as the Control System Toolbox, Signal Processing Toolbox, and Simscape, which offer a range of pre-defined blocks and functions for specific domain requirements [27].

Simulink provides a robust framework for implementing co-simulation through its S-Function blocks, which allow users to integrate external code, custom algorithms, or third-party software into the Simulink model. Additionally, Simulink supports the FMI standard, which enables the integration and co-simulation of models from various simulation tools using a standardised format. Users can leverage Simulink's capabilities to perform seamless co-simulation with other software and tools, performing comprehensive analysis and validation of complex, multi-disciplinary systems in a streamlined way [27].

3 Open Simulation Platform

The Open Simulation Platform (OSP) provides a tool for executing co-simulations focusing on the maritime industry. Also, tools for data visualisation that can be used for generating plots are built into the platform. It also provides tools for manually overriding variable values mid-simulation and a system to define when a variable's value should be overridden through a scenario file using the JSON file format. The co-simulation master algorithm is completely implemented within the tool. It provides an interface for the user to specify the relation between the simulation time and wall-clock time. It is helpful if our goal is to get insights from observing the simulation instead of executing them as fast as our computing hardware allows [25].

The increasing complexity of modern ships, primarily due to the increased number of software systems required, increases the diversity of expertise required to construct such a ship. Therefore, ship projects, known throughout history as large projects, now require a wider array of providers with different specialisations to create the required equipment and systems. This further exacerbates the complexity of a modern ship's design, building, and operating phases. It also makes it more challenging to understand the ship as a complete system. The Open Simulation Platform aims to provide the crucial tools and work processes to address these challenges [25].

In 2018, the Open Simulation Platform Join Industry Project (JIP) was established by organisations including DNV GL, Kongsberg Maritime, NTNU, and SINTEF Ocean, and has since been joined by 20 additional partners from various sectors within the maritime industry. In their paper titled, "Open Simulation Platform – An Open-Source Project for Maritime System Co-Simulation" [25], Smogeli et al. outlined five principal complexity challenges confronting the maritime industry. These challenges provided the rationale behind the project [25]:

- Designing and optimising integrated ship systems has become increasingly challenging due to the complex interactions between systems and software from various sources. Predicting the final system behaviour is difficult, so simulations and operational observations are necessary. Testing and optimising each

system within the integrated ship system is crucial to ensure proper functioning [25].

- Assessing whether systems are safe or doing what is required is complex. Even after being constructed and put into operation, only a small subset of the possible scenarios that the system should be able to handle can feasibly be tested [25].

- The wide array of providers for the global system makes commissioning and integrating them into one system challenging, particularly since no system integrator has a complete overview and insight into all the different subsystems and their connections [25].

- The system is complex, and the required human interactions with the machines are also further exacerbating the issue [25].

- Managing changes becomes more challenging, both during the construction and operation phases. It is impossible to predict the consequences of changes in a software system or other connected systems, as well as the ship system as a whole [25].

The authors theorised that this situation arises because, unlike mechanical systems that are inherently bound by physical limitations and natural laws, the complexity in integrated software systems can rapidly exceed human understanding. Their crucial observation was that these types of system properties cannot be deduced but instead must be observed. Either through the operation of the system or by means of simulation [25].

3.1 Key principles

Smogeli et al. further lists out six fundamental properties and principles which were crucial and steered the development of the Open Simulation Platform. These can be seen in Figure 4 [25]:

- *Model reuse.* Models for simulation and digital twins for smaller parts or subsystems of a ship have seen increasing use by manufacturers, but they are typically not made available for other parties that also play a part in the construction of the global system. For the sake of efficiency and scalability, it is crucial that these digital models, which have required an abundance of resources, be able to be further reused [25].

- *Protection of IPR.* To enable the reuse of digital twin components and simulation models, it is a requirement that no Intellectual Property Rights (IPR) contained in it are visible to other parties without the IPR holder's permission. This requires that the components are delivered as a so-called black box that its user cannot easily reverse-engineer. See more from the section on FMUs, Section 3.4.3 [25].

- *Co-simulation.* One core problem is making these system subcomponents work together as they would in a complete system. This is where co-simulation comes in, by splitting the simulation into a bundle of smaller individual subsimulations connected into one whole simulation, which is covered in detail in Section 1.3 [25].

- *Common standards.* To efficiently construct large co-simulations consisting of individual sub-components, possible interoperability and common standards are required [25].

- *Open source.* Open source is an important tool to push standardisation and interoperability. The idea is that any party can use the open source software to improve their working processes and tools, reducing the amount of friction in cooperation and promoting common ownership and for organisations to invest into further developing the open source software [25].

- *Collaboration.* The challenge of complex systems is too great for any one organisation. As a result, the focus is on collaboration and interoperability between stakeholders and tools [25].

The Open Simulation Platform Join Industry Project has developed the OSP Interface Specification (OSP-IS) by adhering to these principles. To build the OSP Interface Specification, they have utilised the Functional Mockup Interface (FMI) standard, which is already widely used in the automotive industry and is under constant development, see Section 3.4.3. Moreover, they have also created an open-source C++ library, named *libcosim*, to support co-simulation, which adheres to the OSP-IS standard. The libcosim library is based on a range of technical solutions, with FMI at its core. A set of reference models has been created for some of the more common maritime systems and ship dynamics. These models are constructed using the OSP guidelines and interface standards and are intended to assist with the simulation of ship systems [25].

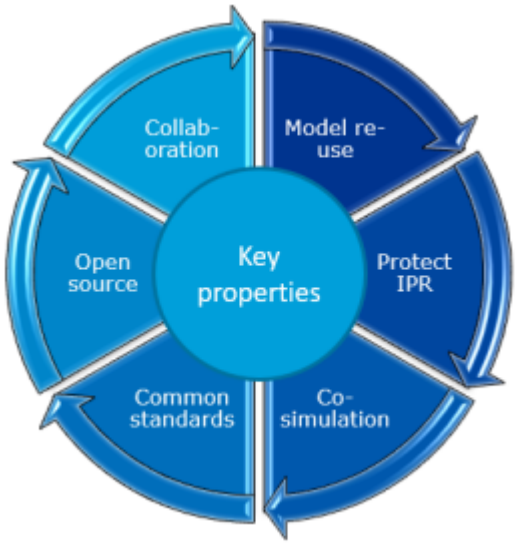

Figure 4: The main attributes and principles of the Open Simulation Platform initiative [25].

3.2 OSP co-simulation architecture

The Open Simulation Platform Joint Initiative has developed a set of tools called the Core Simulation Environment (CSE) that enables effective co-simulation. The CSE consists of five parts [28]:

- *The C++ co-simulation library*, an open-source library offering support for co-simulation in alignment with the OSP-IS. This library constitutes the centerpiece of the CSE. It facilitates the development of models adhering to the OSP standard and is capable of conducting simulations of FMU models set up as system structures. It includes a fixed-step co-simulation master algorithm, a scenario runner, and features for monitoring and manipulating simulation variables [28].

- *The Demo application*, a graphical user interface (GUI) application that demonstrates how to use the C++ co-simulation library. The Demo application is intended to provide an overview of how to use the library and the various features it offers while providing a simple, convenient user interface to display real-time plots and adjust simulation parameters through a GUI application [28].

- *The Command-line interface*, a tool that allows users to interact with the C++ co-simulation library through a command-line interface. It is a more streamlined approach to execute several simulations using scripts or other tools [28].

- *The Model interface validator*, a tool that checks if a given FMU follows the OSP Interface Specification. It is designed to ensure that all FMUs are created following the OSP guidelines and standards, is interoperable, and can be used in OSP co-simulations without any external issues [28].

- *The CSE Java wrapper*, a Java-based tool that provides a wrapper around the C++ co-simulation library. It allows Java-based applications to use the C++ co-simulation library through a Java interface [28].

In figure 5 [25], the architecture of the OSP is visualised. The FMUs depicted in the figure, provided by Original Equipment Manufacturers (OEMs) and denoted by the blue rectangles, are linked to the co-simulation interface in the CSE, represented by the dark gray rectangle. The OSP co-simulation master algorithm facilitates data

transfer between the FMUs based on the input and output variable mappings defined in a distinct interface specification file. A scenario management tool allows for the external manipulation of simulation variables, using a scenario file. This file specifies the values for a variable within the simulation by defining the simulation time point at which the variable should take the given value [28].

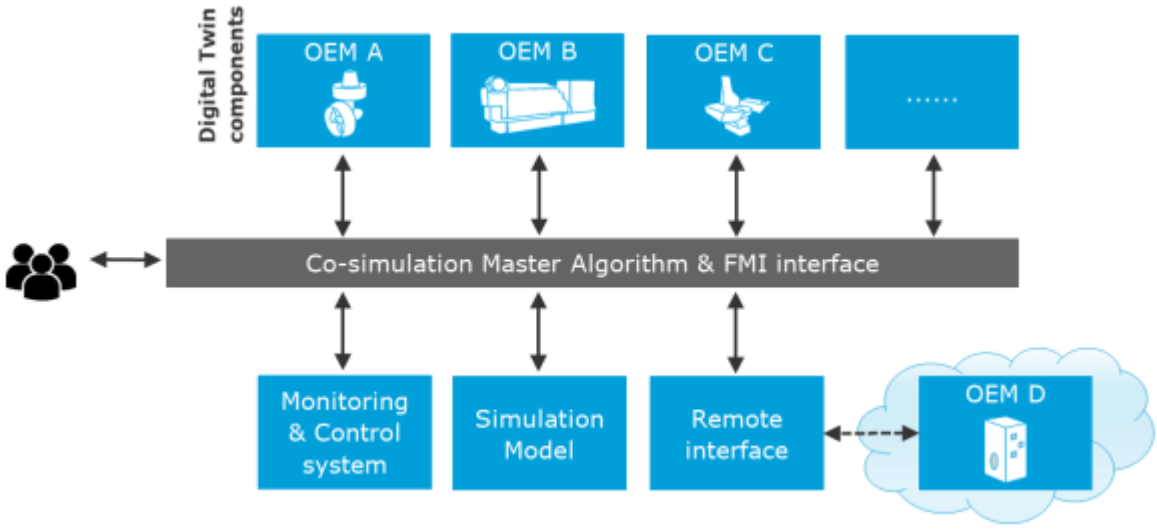

Figure 5: The conceptual architecture of the OSP architecture, with components from several OEMs connected into one co-simulation [25].

3.3 OSP Interface Specification

With the lack of any well-established interface standard, integration of subsystems into one complete co-simulation is problematic. This is what the OSP Interface Specification (OSP-IS) strives to solve. The core of it relies on the already existing and well-established standard, especially in the automotive industry, FMI. FMI does not provide a straight out-of-the-box solution but a partial one. It ensures binary compatibility for model connections. However, it does not address semantic correctness, meaning it does not guarantee that the models are semantically correct or that they will produce accurate results when used together [25].

The OSP-IS is an extension of the FMI standard that simplifies model connections and ensures that they are semantically correct. At the core, OSP-IS is a set

of metadata specifications for models and model interfaces. This metadata is represented in the form of an ontology, which is a formal representation of the knowledge required for the model connections [25].

The connections between the metadata and the model interface are specified in a separate XML file named OSPModelDescription.xml. This file links signals in the different Functional Mock-up Units (FMUs) to concepts in the ontology [25].

The ontology provides definitions that can be used to describe the model interface signals, as well as rules for connections that can be used to figure out how models can be connected. The ontology consists of concepts such as variables, each with its datatype, causality, and unit. Variable groups are also supported, enabling the grouping of tightly coupled variables. This is especially useful when dealing with multidimensional vectors where we can group all of its elements into one variable group [25].

3.4 FMI (Functional Mock-up Interface)

The Functional Mock-up Interface (FMI) is a tool-independent standard designed to facilitate the uncomplicated exchange of models and their use as sub-simulators in co-simulations. The initial version of this standard, FMI 1.0, was introduced in 2010, and quickly gained acceptance with over 30 tools supporting the FMI 1.0 standard by 2012 [29].

The FMI standard is primarily employed in industrial and scientific projects, with the automotive sector being a key user. The driving force behind the FMI standard was to enhance the transfer of simulation models for original equipment manufacturers. This would enable manufacturers to test and contrast various subcomponents in a digital setting before making a physical commitment [29].

Mercedes-Benz employs FMI for software-in-the-loop simulations in all their new gearbox projects. Prior to the implementation of the FMI standard, vehicle models from multiple vendors had to be imported through vendor-specific and version-specific procedures into the software utilised for modelling. This process was not only costly but also prone to errors. With FMI, these have now been supplanted by

uniform import interfaces, which has significantly reduced the long-term expenses associated with simulation [29].

Previously known as Daimler AG and now operating as The Mercedes-Benz Group AG, the company utilised FMI for mechatronic gearshift simulations. They integrated controller software with a one-dimensional power train model in the SimulationX tool. The model was then exported as an FMU and imported into a multi-body simulation tool, where it was connected to a detailed truck model. This facilitated a comprehensive simulation and optimisation of shifting comfort [29].

FMI has since 2010 been updated with two new major versions: FMI 2.0 and FMI 3.0.

3.4.1 FMI 2.0

The FMI 2.0 standard comprises two key components: FMI for model exchange and FMI for co-simulation. FMI for model exchange enables a modeling environment to generate code for a dynamic system model as an input/output block. It can be seen as a function that takes inputs and generates outputs based on some logic. Other modeling and simulation environments can use those blocks [29].

FMI for co-simulation is geared towards the coupling of two or more models with solvers within a co-simulation environment. Data exchange between subsystems is confined to discrete communication points. Between these points, the subsystems are independently resolved by their respective solvers. A co-simulation master algorithm manages the data exchange among them [29]. Section 1.3.4 goes more in-depth about the internals of co-simulation.

In the FMI 2.0 standard, the interfaces for model exchange and co-simulation were integrated. This contrasts with the FMI 1.0 standard where they were defined in separate XML documents. Furthermore, a Functional Mock-up Unit (FMU) can implement both interfaces simultaneously, see Section 3.4.3. In addition, interface variables received an expanded range of classification options [29].

Variables have a *causality* attribute, which is an enumeration that dictates the causality of the variable. The FMI 2.0 standard allows for values such as parame-

ter, input, output, and local. Parameter represents an independent variable that is required to stay constant during the simulation. Input indicates that the value will be supplied by another model, while output means that the value can be utilised by another model. Local refers to a variable that is calculated based on other variables, but it's not permitted to be used in other models [29].

Variables have another attribute called *variability*, which is an enumeration defining the temporal dependency of the variable. In other words, it delineates the instances when the variable can alter its value. The permitted values include constant, fixed, tunable, discrete, and continuous. Constant implies the value never fluctuates. Fixed indicates that the value of a variable remains unchanged post-initialisation. Tunable variables have values that remain steady between externally prompted events due to the alteration of variables with causality. This allows a modelling environment to reveal independent parameters that can be manually adjusted during the simulation. Discrete implies that the variable's value stays constant between internal events, while continuous places no restrictions on the variability of a variable's value [29].

The FMI 2.0 standard presents a method for preserving and reinstating an FMU to an earlier state. This saved state encompasses values of continuous and discrete states, iteration variables, parameter values, input values, file identifiers, and internal status details pertaining to the FMU. Moreover, the state of the FMU can be duplicated, allowing the utilisation of that pointer to revert back to that state within a simulation [29].

There were also several more features and improvements, such as more options for dependency information, precise time event handling, explicit alias/anti-alias variable definitions, and improved unit definitions [29]. The FMI 2.0 standard has gained widespread recognition as the industry standard for model exchange and utilisation in modeling and simulation tools. Presently, over 160 simulation tools provide support for this standard [30].

3.4.2 FMI 3.0

With the FMI 3.0 standard, enhanced co-simulation algorithms and new applications, including the packaging and simulation of virtual electronic control units (vECUs). A third interface type, scheduled execution, is added to the existing model exchange and co-simulation interfaces. Clocks allow for synchronisation of the FMUs, and support for data types has been improved. Signal handling has been simplified by terminals which allows bundling signals together into groups [30].

The previous FMI standards were taken into use by the industry and by numerous different simulation tools. For an extended period, stability was a critical for the success of the standard, leading to several releases for the FMI 2.0 standard. Through new use cases, the need for improved capabilities was clear; these are targeted by the FMI 3.0 standard [30].

In the FMI 2.0 standard, incorporating control code into FMUs necessitates some workarounds. However, the FMI 3.0 standard simplifies the process of exporting virtual electronic control units as FMUs by introducing new features such as terminals, clocks, enhanced integer and binary data types, array variables, and the new interface known as scheduled execution [30].

Event handling in FMUs is improved through a more flexible API for event handling and communication, such as the synchronous clocks API. Scenarios driven by events enable FMUs to communicate better information about both timing and the cause behind events [30].

Source code build configurations have been added to give the user of the FMU the option to choose the desired build configuration for their platform. This is done with an XML document which defines build information and specifies source files, such as processor definitions, include paths and dependencies [30].

3.4.3 FMU (Functional Mock-up Unit)

The FMI standard outlines that a model should be encapsulated within a Functional Mock-up Unit (FMU), which is a folder containing model code files designed for one or multiple platforms. This folder can also contain metadata and documentation

for the model. In the case of the Windows operating system, an FMU typically consists of shared libraries (dll files) and an xml file to store metadata and other supplementary files. The standard also defines the structure and format of files and directories within the FMU, see Figure 6 [18]. Additionally, the FMI standard specifies the APIs that must be defined for the model code, predominantly using the C programming language. Wrappers are available to support other programming languages such as C++, Java, and Python [18].

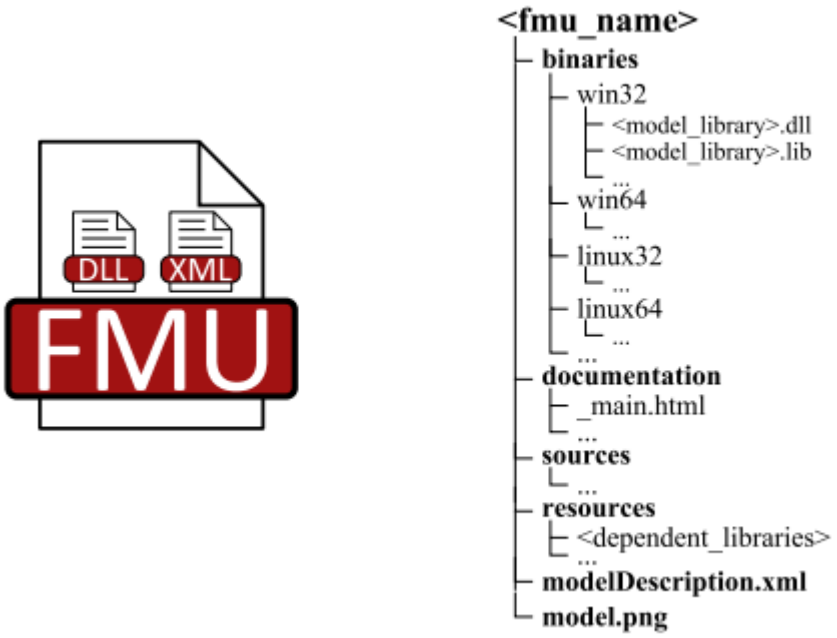

Figure 6: The folder structure of an FMU [18].

FMUs based on FMI for co-simulation define an interface for models which come packaged with numerical solvers for each model. The resulting FMU implements a sub-simulator as compiled code and metadata. A co-simulation is a set of these FMUs connected, with the co-simulation master algorithm handling the data transfers between the FMUs and the synchronisation of time steps. In co-simulation, we have discrete time steps where the exchange of data happens at the discrete communication points. The models are solved independently between the communication points by their own solver implementation [18].

The API does not, however, restrict whether a solver is implemented. It only

specifies that the FMU performs a time step of defined length. If not necessary, an FMU does not need to implement a solver. This is common when an FMUs purpose is to function as a communication device between some hardware or user interface for some form of human interaction in the co-simulation [18].

The communication in the FMI standard for co-simulation is based on a master/-worker model where the FMUs are workers, which are controlled by the co-simulation master algorithm. Therefore, an FMU has no concept of the other FMUs participating in the simulation. It is a strictly individual entity that produces output based on its inputs and the implementation of its solver. FMUs are immutable; that is, after they have been created, their behaviour or interface cannot be changed in a trivial way. This is because the code in the FMU is in binary code format, so numbers, types, and names of variables are static. This sacrifices some flexibility in modifying models continuously, it is one of the key features of the FMI standard since it allows manufacturers to share their models without revealing any business secrets. Hence FMUs are often called black boxes [18].

The FMI standard for co-simulation does not specify any implementation details for the co-simulation for how data should be exchanged between FMUs as well as how the FMUs are time synchronised. This makes it easy for a wide array of simulation tools to implement support for the FMI standard, and in fact, the number of tools supporting FMI today is large and is seeing continuous growth [19].

3.4.4 fmiCpp

The fmiCpp library [24], developed by Stian Skjong, is used to create FMUs that can be used for running co-simulations. The fmiCpp library is the primary tool used within this thesis's scope to construct the FMUs for conducting the experiments in Section 4.

The fmiCpp library is under the Big Time Public License and is free to use in a non-commercial setting. As of the time of writing, there is no full public release available of the library. A pre-release version can be requested from Stian Skjong [1].

[1] stian.skjong(at)sintef.no

The library uses CMake, version 3.12 or higher, for building and uses the Conan package manager, version 1.4 or higher, for package handling. The C++ version should be C++14 or above. The CMake build process requires that the source code for an FMU is placed in a folder in a specified way for the building process to be successful, see Figure 7.

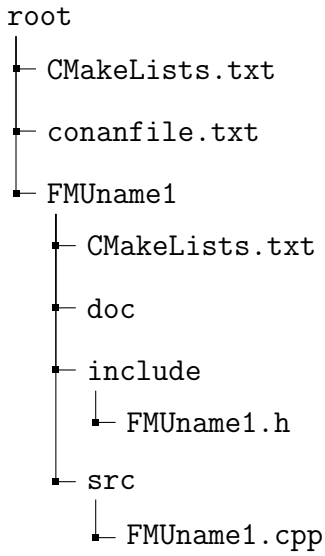

Figure 7: The folder structure in fmiCpp [24].

The library provides a set of abstract classes which can be inherited for the concrete class which represents the FMU. The classes have virtual methods for defining behaviour for the FMU at different stages of the discrete co-simulations step. Also, they come bundled with a set of data members containing data about the FMU, for example, its time step size. The library also enables the creation of standalone executables for the FMUs, which makes debugging convenient. An interface to the Python library Matplotlib is included, which makes it possible to generate plots for data visualisation of the variables inside of the FMU during co-simulation [24].

4 Battery sub-system simulation of a Ropax ship with electrical car charging

4.1 The battery sub-system

In my simulation, the battery sub-system comprises three primary components: the power management system, the battery, and the generator. The battery and the generator keep track of the state of the generator and the battery, i.e. the amount of power being generated by the engines, the number of engines being used, and the state-of-charge (SoC) of the battery. The power management system is the most complex component of the simulation. It functions as the main component of the battery sub-system simulation, controlling the generator and battery based on inputs such as battery SoC and required demand for the system. The battery provides an accessible reservoir of power which can be used when demand exceeds the current output from the generator, during periods of high-energy usage.

In the automation and energy systems of maritime vessels, particularly those with electric propulsion and station-keeping thrusters, the power management system is crucial. Its primary objective is to optimise blackout prevention and reduce fuel consumption by effectively regulating the power system. Additionally, it helps reduce maintenance expenses by safeguarding equipment against faults and malfunctions. Enhanced vessel performance is achieved through the power management system's collaboration with other control systems on board. The primary purpose of the power management system is to ensure a reliable power supply to the power-consuming components within the vessel [31]. It achieves this through [31]:

- *Generator allocation control.* The power management system controls the number of active generators based on the network's load and operational conditions [31].
- *Propulsion load limiting control.* The power management system prevents excessive load increases by controlling the maximum individual consumption of components, such as the thrusters [31].

- *Fast load reduction.* To prevent overloading the generators, the power consumption of variable frequency drives is regulated. In the event of an overload, such as a generator shutdown, the power management system will initiate load reduction on one or more variable frequency drives until the situation stabilises [31].

- *Blackout restart.* In the event of a partial or total blackout, the power management system will execute a restart of the power system [31].

The battery is usually monitored and controlled by a separate component, the battery management system. Due to the high level of abstraction in the simulation, the battery management system's functionality is bundled into the power management system.

The battery management system ensures efficient and safe utilisation of onboard battery energy storage systems. In maritime vessels, the battery management system is responsible for monitoring and controlling the charging and discharging of batteries, optimising their performance, and extending their lifespan. The battery management system aims to reduce fuel consumption and emissions by supporting hybrid propulsion systems, enabling peak shaving, and facilitating load leveling. Also, it enhances the ship's overall energy efficiency by integrating with the power management system to maintain a stable power supply and optimise power distribution. The battery management system also enhances the ship's safety and reliability by protecting the batteries from overcharging, over-discharging, short circuits, and thermal issues [32].

4.2 The OSP model architecture of a Ropax ship with hybrid charging capabilities

4.2.1 The PowerManagementSystem FMU

The *PowerManagementSystem* serves as the central control unit for the battery sub-system, facilitating communication between the generator and the battery. It

monitors the demand and the status of the generator and battery. It decides the number of active engines, engine power, and whether to use or charge the battery. The decision-making logic is based on an ABB study's controller strategy flow chart, with some modifications to accommodate the data collected from an operational ferry [33], see Figure 8.

By continuously analysing the power demand, generator output, and battery SoC, the *PowerManagementSystem* can optimise the battery sub-system's operation to meet the required power demand while ensuring efficient use of energy resources. Its decision-making process should help maintain a balance between power generation, energy storage, and consumption, contributing to the performance and sustainability of the ship's power management.

The *PowerManagementSystem*'s ability to dynamically adjust the number of active engines and engine power should allow for optimisation of the load management and reduced fuel consumption. At the same time, decisions regarding battery usage help prolong the battery's lifespan by maintaining an optimal SoC within a given threshold as consistently as possible.

Inputs:

- *Demand (kW)*
- *Car charger demand (kW)*
- *Engine power (kW)*
- *Battery SoC (%)*

Outputs:

- *Number of engines*
- *Requested engine power (kW)*
- *Requested battery power (kW)*

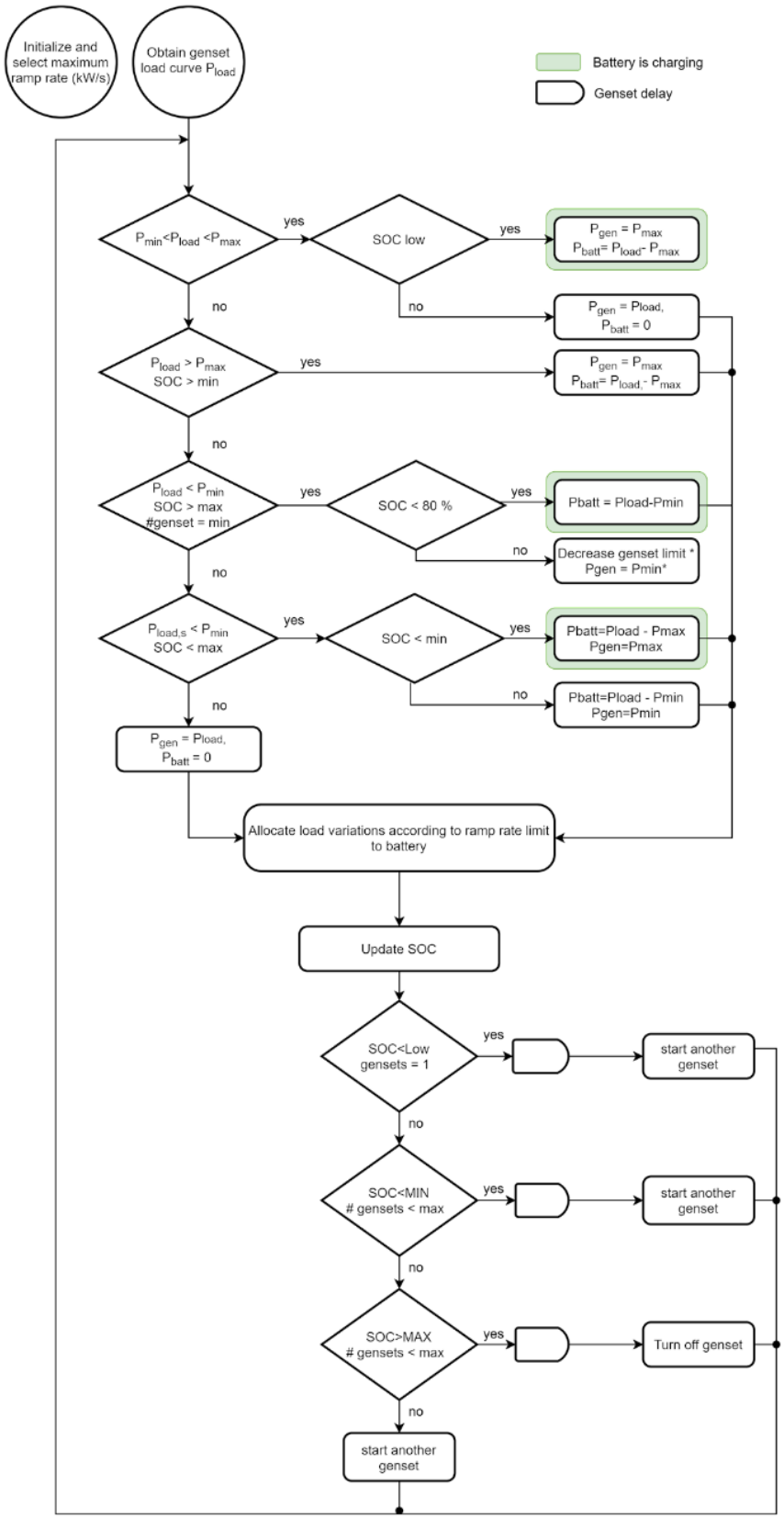

Figure 8: Flow chart of the control strategy used in the ABB study [33].

4.2.2 The Battery FMU

The *Battery* is designed to model the onboard battery of the ship. It receives input from the *PowerManagementSystem*, which dictates the desired power level for the battery based on the system's overall demand. By adjusting its operation according to these inputs, the *Battery* can optimise energy storage and distribution to satisfy the power requirements of the ship.

The *Battery* calculates power peaks in kilowatts, representing the maximum power output at any given moment. This information is essential for understanding the battery's performance and ability to handle sudden changes in demand or generation.

Additionally, the *Battery* calculates the SoC in percentages, reflecting the current energy level of the battery in relation to its total capacity. Monitoring the SoC is vital for effective power management. It allows the *PowerManagementSystem* to make informed decisions about when to charge or discharge the battery, with the goal of extending its lifespan and ensuring efficient utilisation of energy resources.

Inputs:

- *Requested power (kW)*

Outputs:

- *Peak (kW)*
- *SoC (%)*

4.2.3 The Generator FMU

The *Generator* FMU manages the engines within the battery sub-system. It receives input from the *PowerManagementSystem*, which includes the desired power level and the number of engines running at any given time. By adjusting the engine's operation based on these inputs, the *Generator* can optimise power generation to meet the system's requirements.

Additionally, the *Generator* calculates methane slip, which refers to unburnt fuel that has not been completely combusted in the engines. Methane slip is an important parameter to monitor, as it can have negative environmental consequences and reduce the overall efficiency of the engines. By accounting for methane slip, the *Generator* FMU can help identify potential areas for improvement in engine performance, leading to enhanced energy efficiency and reduced environmental impact.

Inputs:

- *Engine power (kW)*
- *Number of engines running*

Outputs:

- *Generated engine power (kW)*
- *Methane slip (kg)*

4.2.4 The DemandFunction FMU

The *DemandFunction* is a virtual FMU designed to simulate power demand to the battery sub-system. It uses operational data of a ferry to account for the real-world power consumption as accurately as possible. The dataset consists of samples collected over 24 hours, with samples collected every second. During the simulation, the *DemandFunction* will update the power demand at every second according to the collected data, providing a dynamic and realistic representation of energy consumption. The demand is measured in kilowatts, allowing for precise quantification of the power required by the battery sub-system to meet the needs of the ferry at any given time in the simulation.

Outputs:

- *Demand (kW)*

4.2.5 The CarCharger FMU

The *CarCharger* adds demand to the system in addition to the demand originating from the *DemandFunction*. Its purpose is to approximate the power demand that an onboard car charger would require, considering the SoC of the cars and the number of cars charging simultaneously. The demand functions are based on a 2021 Tesla Model 3 LR AWD with an 82 kWh battery charging analysis, see Figure 9. Some simplifications have been made within the FMU: the charging rate is constant from zero percent to 100 percent, and the SoC to charging power relationship has been converted into a discrete function, with the charging power defined at every 10 percent interval.

During the initialisation phase of the simulation, the number of cars on deck can be set, allowing for customisation of the charging demand. Additionally, the number of cars being charged at any given moment during the simulation can be adjusted, offering further flexibility in evaluating different charging scenarios.

Outputs:

- *Demand (kW)*

4.2.6 Logic

The battery sub-system is defined by the *PowerManagementSystem*, *Battery*, and *Generator* FMUs. The *DemandFunction* and *CarCharger* FMUs simulate components that consume power generated by the battery sub-system. The demand from the *DemandFunction* and *CarCharger* demand is simply summed together to create a total demand required for the battery sub-system, see Figure 10. Based on the total demand, the *PowerManagementSystem* contains the logic that decides how many engines are required to be active, how much power the engines should produce, and whether the battery should be charged using surplus power generated from the engines or if the battery is needed for the extra power to satisfy the total demand.

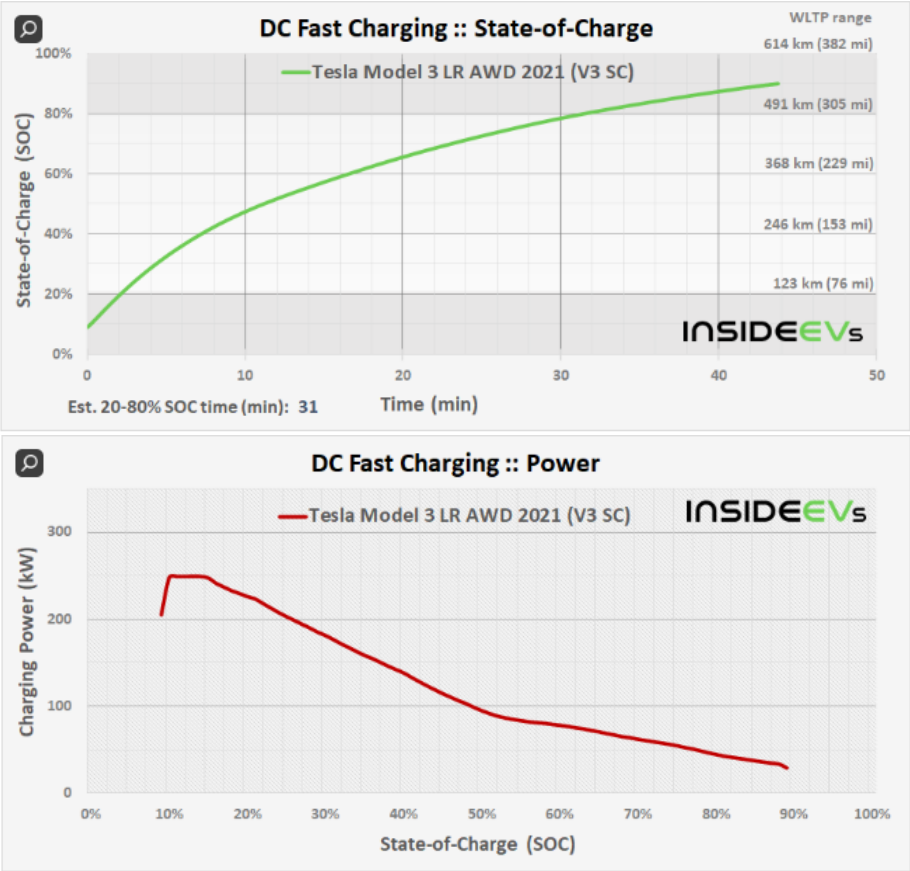

Figure 9: Time under charge to SoC and SoC to power demand [34].

The *PowerManagementSystem* constantly monitors the SoC of the battery, taking it into account when making decisions about charging or utilising it for power. As shown in the ABB flowchart, see Figure 8, the battery has lower and higher thresholds defined for its SoC. The goal is to keep the battery's SoC within these thresholds as consistently as possible, which in turn helps increase the battery's lifespan. In the given simulation, the lower threshold is 60 percent, whereas the upper threshold is 80 percent. By adhering to these limits, the *PowerManagementSystem* can optimise the battery usage, ensuring efficient power management and reduced long-term wear on the battery.

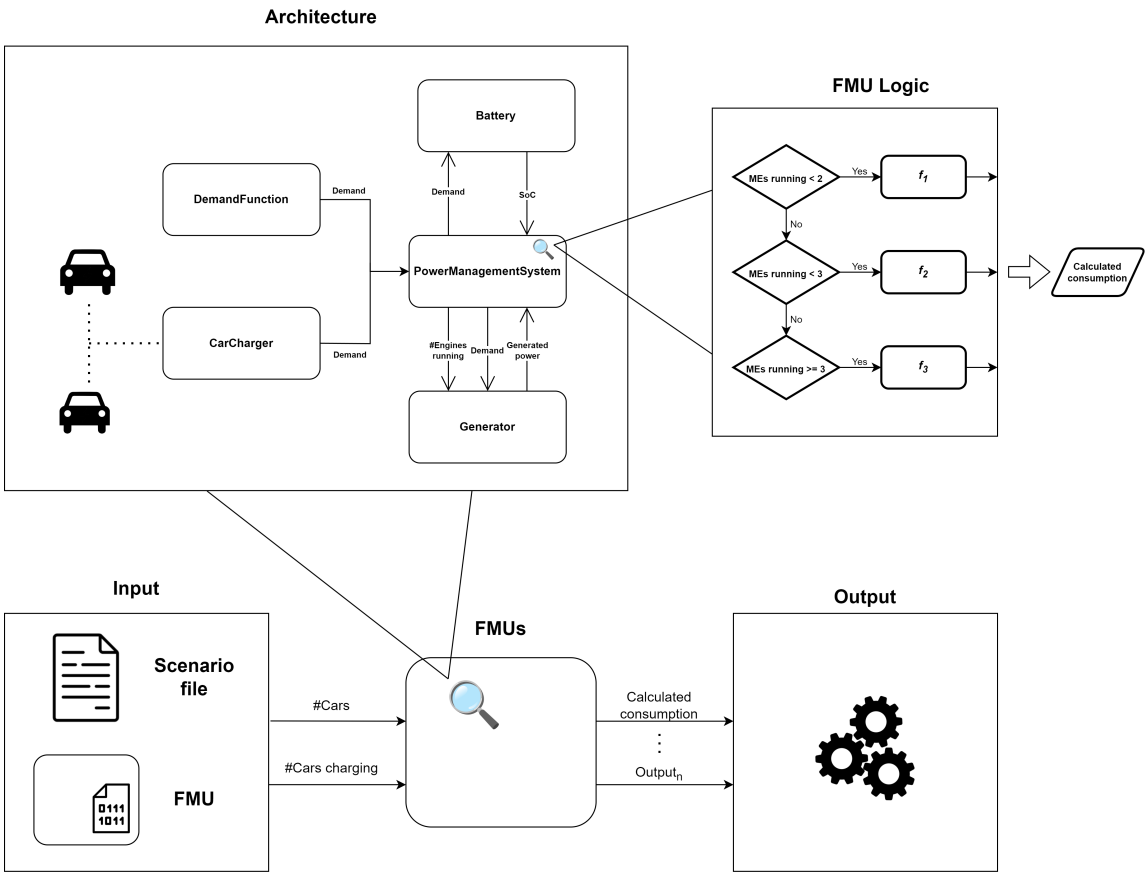

Figure 10: An overview of the simulation, the magnifying glasses indicate a drop in the abstraction level. In the upper left the connections between the FMUs are shown.

4.3 Experiments

4.3.1 Simulation setup

For the simulation, the Open Simulation Platform (OSP) command-line tool is utilised to execute the simulations. Each FMU participating in the simulation generates a CSV file containing the values for each variable at every discrete time step within the simulation. A one-second time step is used for all FMUs, which aligns with the recorded demand values from the ferry which were collected in one-second intervals.

The FMUs participating in the simulation must all be placed into a single folder to execute the simulation. For each FMU, an OSP model description file must be created, which defines the variables from the FMU to be used in the simulation, see

Figure 11. The OSP system structure file needs to be added to the folder in order to establish the connections between the FMUs and set the step sizes, see Figure 12.

Once the folder is set up with the required files, the OSP command-line tool can be run to initiate the simulation. During the simulation, the FMUs will interact according to the connections defined in the system structure file, exchanging data and performing their respective tasks. The generated CSV files provide a detailed record of each FMU's variables throughout the simulation, allowing for system performance analysis and evaluation.

```
<?xml version="1.0" encoding="utf-8" ?>
<OspModelDescription xmlns="https://open-simulation-platform.com/
    OspModelDescription/1.0.0" version="1.0">

    <VariableGroups>
        <Generic name="Peak (kW)">
            <Variable ref="Peak (kW)"/>
        </Generic>

        <Generic name="SoC (%)">
            <Variable ref="SoC (%)"/>
        </Generic>

        <Generic name="Requested power (kW)">
            <Variable ref="Requested power (kW)"/>
        </Generic>
    </VariableGroups>

</OspModelDescription>
```

Figure 11: Example of the OSP model description file for the battery FMU.

4.3.2 Simulation scenarios

In the simulation scenarios, the *CarCharger* variables are the ones that will be modified. Specifically, these include the number of electric cars on deck requiring charging during the itinerary and the number of cars being charged simultaneously. Both variables will be defined at the start of the simulation and will remain static throughout the simulation process.

For example, in a scenario with twenty total electric cars and two being charged at a time, the simulation will add extra demand to the system according to the two cars being charged, based on the logic defined in the *CarCharger* FMU. Once the

```
<?xml version="1.0" encoding="utf-8" ?>
<OspSystemStructure xmlns="http://opensimulationplatform.com/MSMI/OSPSystemStructure
    " version="0.1">
    <StartTime>0.0</StartTime>
    <BaseStepSize>1.0</BaseStepSize>
    <Algorithm>fixedStep</Algorithm>
    <Simulators>
        <Simulator name="Battery" source="Battery.fmu" stepSize="1.0"/>
        <Simulator name="PowerManagementSystem" source="PowerManagementSystem.fmu"
            stepSize="1.0"/>
    </Simulators>

    <Connections>
        <VariableConnection>
            <Variable simulator="Battery" name="SoC (%)"/>
            <Variable simulator="PowerManagementSystem" name="SoC (%)"/>
        </VariableConnection>
    </Connections>

</OspSystemStructure>
```

Figure 12: Example of the OSP system structure file with two FMUs and one connection in between them.

two cars is fully charged, two other cars will begin charging, and this process will continue until all twenty cars have been charged. At that point, the *CarCharger* FMU will no longer add any extra demand to the system.

By altering the *CarCharger* variables, various scenarios can be simulated to study the impact of electric car charging on the battery sub-system and the overall energy management of the maritime vessel. This enables assessment of the capacity and robustness of the system under different levels of electric car charging demand to provide insights for optimising power management strategies and infrastructure planning.

Through these simulation scenarios, potential challenges and bottlenecks in the system could be identified and addressed, ensuring that the battery sub-system and power management system can effectively meet the demand for electric car charging.

Three simulation scenarios will be conducted to assess the impact of electric car charging on the battery sub-system and overall energy management of the maritime vessel. These scenarios will help us understand how the system performs under electric car charging demands.

Case 1: No electric cars on deck

In the first scenario, a case with no electric cars on deck will be simulated, meaning that there is no additional demand for charging. This baseline scenario shows the performance of the battery sub-system and the power management system without electric car charging requirements. By analysing the results from this scenario, a reference point can be established for evaluating the system's performance when introducing electric car charging demand.

Case 2: Twenty electric cars on deck with two cars charging at a time

In the second scenario, a case with twenty total electric cars on deck and two cars being charged simultaneously during the itinerary will be simulated. This scenario will show the impact of electric car charging on the battery sub-system and the power management system when charging a relatively low number of cars at a time. The *CarCharger* FMU will add extra demand to the system based on the charging requirements of the two cars, following the logic defined in the FMU. Once the two cars are fully charged, two other cars will begin charging until all twenty cars have been charged.

Case 3: Eighty electric cars on deck with eight cars charging at a time

In the third scenario, we will simulate a case with eighty total electric cars on deck and eight cars being charged concurrently during the itinerary. This scenario will enable us to evaluate the strain and resilience of the battery sub-system and the power management system under a high electric car charging demand.

In this case, the *CarCharger* FMU will add extra demand to the system based on the charging needs of the eight cars by the logic defined in the FMU. After the first set of eight cars are fully charged, another set of eight cars will begin charging. This process continues until all eighty cars have been charged. By comparing this scenario with the two previous ones, we can better understand how increasing electric car charging demand impacts the system's performance.

4.3.3 Results

Case 1: No electric cars on deck

In the first scenario, see Figure 13, where no electric cars were on deck and therefore, no additional charging demand was placed on the battery sub-system, the SoC consistently stayed within the predefined thresholds of 60% and 80%.

At the start of the journey, and for the first 20% of the trip, there was a drop in SoC to nearly 70%, indicating an increase in the system's energy demand. Despite this increased demand, the battery sub-system maintained the SoC above the lower threshold. After this initial demand period, the SoC was able to recover back to 80%.

There were two instances where the SoC dropped to the lower threshold of 60%, see Figure 13. These instances suggest periods of peak demand. The similarities in the SoC decrease and the time they occurred at indicate that this could be during the layover at the halfway point in the itinerary in both directions. However, the battery SoC did not fall below the lower threshold in both instances. Following these drops, the SoC was able to recharge back up to 80%.

The CH4 emissions, measured in kilograms, fluctuated between 200 and 400, with higher values at the first and last thirds of the trip. This could be due to increased power demand during these periods, possibly from increased propulsion needs during the departure and arrival phases of the journey.

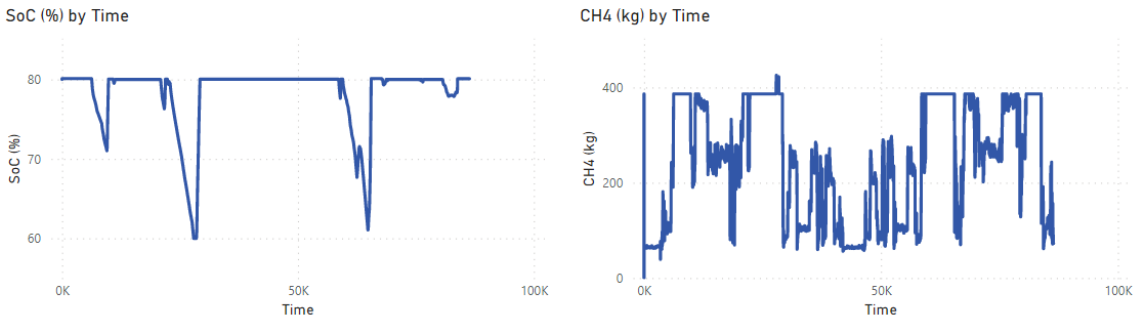

Figure 13: Results from the first scenario, the left graph shows SoC over time in seconds, and the right graph similarly for methane slip.

Case 2: Twenty electric cars on deck with two cars charging at a time

In the second scenario, see Figure 14, where twenty electric cars were on deck with two cars charging at a time, the battery sub-system performed similarly to the first case, with some differences.

The SoC showed a similar pattern as in the first case, staying within the given thresholds of 60% and 80%. However, at the start of the trip, the SoC dropped more significantly than in the first case, reaching almost 65%. This decrease can be attributed to the additional demand for charging electric cars. Despite this increased demand, the power management system managed to maintain the SoC above the lower threshold.

Much like in the first case, the SoC dropped to the lower threshold of 60% twice but did not go below it. These drops indicate periods of increased power demand, but the power management system managed to handle these periods, recharging the battery back to 80% in both cases, see Figure 14.

All electric cars were fully charged early in the trip, indicating that the system was able to handle this additional demand without significant disruptions or deviations from its performance.

The CH4 emissions showed a nearly identical pattern as in the first scenario, fluctuating between 200 and 400 kg, with higher values during the first and last thirds of the trip.

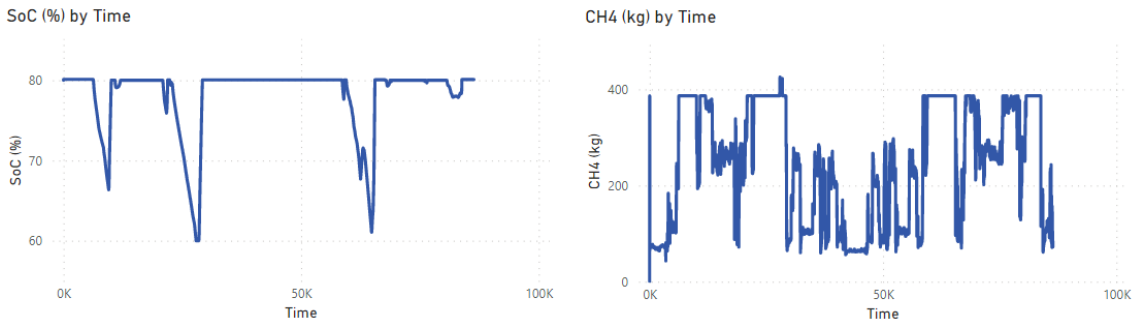

Figure 14: Results from the second scenario, the left graph shows SoC over time in seconds, and the right graph similarly for methane slip.

Case 3: Eighty electric cars on deck with eight cars charging at a time

In the third scenario, see Figure 15, with eighty electric cars on deck and eight charging at a time, there were more differences than in the first two cases, mainly at the start of the trip, when most cars were being charged.

The initial drop in the SoC was the most significant difference. In this case, the SoC dropped to the lower threshold of 60% at the start of the trip due to the higher demand for charging eight electric cars. The power management system could stay within the given threshold despite this increased load, see Figure 15.

Following the first drop, the SoC recharged back to 80%. However, unlike the previous cases, there was another smaller drop in SoC to almost 75% after the first recharge. After these initial fluctuations, the SoC behavior for the rest of the trip was similar to the other cases, staying within the thresholds of 60% and 80%, see Figure 15.

The CH4 emissions were slightly higher at some points during the start of the trip compared to the previous cases, still fluctuating between 200 and 400 kg aside from one peak going slightly above 400 at the start. This increase is due to the higher power output to satisfy the increased demand for electric car charging. However, despite this increase, the CH4 emissions remained within a similar range as the previous scenarios for the remainder of the trip.

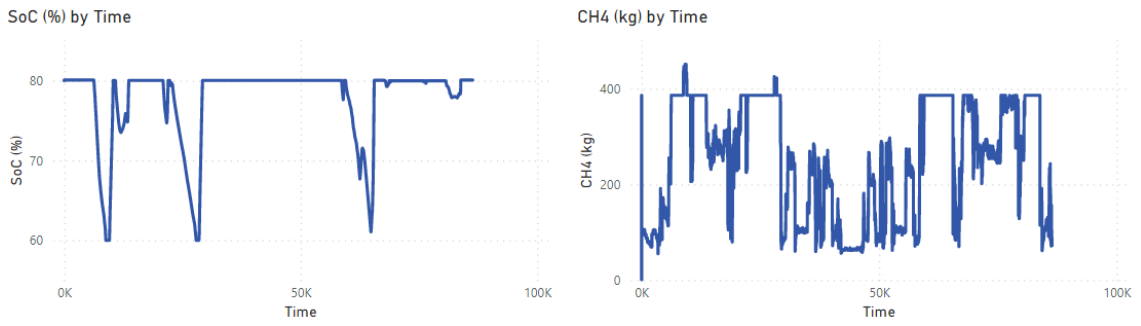

Figure 15: Results from the third scenario, the left graph shows SoC over time in seconds, and the right graph similarly for methane slip.

5 Discussion

This thesis investigates the potential of co-simulation for the maritime industry when it comes to making both better design and operational decisions. The thesis draws from review of literature and case studies on co-simulation both directly from the maritime industry and other industries such as construction and the automobile industry. The maritime industry especially draws benefits from the modular design of co-simulation. Ships are very large and complex systems, being able to easily split the system into smaller sub-systems makes the construction of such an extensive simulation more feasible than in the case of traditional simulation methods.

The data used in this thesis is based on real operational data collected from a ferry and the sub-systems created in the form of FMUs are based on a model done for the same ferry. The goal was to be able to transform this model into a co-simulation, making it more modular and easier to work with. Also, we wanted to evaluate how a potential electric car charging system placed on the car deck of the ferry would affect the battery sub-system. The FMU modelling the electric car charging functionality was based on charging data collected from a Tesla Model 3.

We effectively created a high-level co-simulation that modeled a hybrid ferry, allowing us to experiment with the potential impact of introducing a non-existing component to the ferry during operation. The electric car charging component was connected to our co-simulation and the system's response was observed and analysed.

The results of this simulation, as detailed in figures 13, 14, and 15, revealed insightful findings about the performance and efficiency of the ferry under the varying operational conditions provided by the different test scenarios. These figures provide a visualisation of how the inclusion of the car charging system influences the ferry. We were able to effectively test and evaluate the electric car charging system before having to physically implement it. Such an approach reduces the risks associated with introducing new technology, and significantly accelerates the process of design and optimisation. These findings underscore the potential of using co-simulation and OSP for enhancing the reliability and sustainability of maritime systems, and repre-

sent a crucial step forward in the pursuit of more innovative and efficient maritime vessels.

6 Future work

Our co-simulation featured a high-level of abstraction, for more accurate results a higher level of fidelity would be required for the FMUs. Achieving this fidelity would require a greater depth of expertise in the associated components. The simulation was ran with a single data set. However, to thoroughly evaluate the robustness and versatility it would be necessary to test it with a wider array of data sets. It would be especially important to include data reflecting varying weather conditions as this could have a large impact on energy requirements. Within the scope of this thesis only one tool for co-simulation was used in practice. To establish a better understanding of available technologies and methodologies, future studies should also evaluate and compare other similar tools. Further work should be put into scenario generation. The scenarios used in this thesis were manually constructed. An automated scenario generation tool would significantly increase the value of the co-simulation. Such a tool could produce a diverse set of scenarios, ideally with an emphasis on isolating and identifying faulty behaviours.

7 Summary in Swedish - Svensk sammanfattning

Samsimuleringsmodeller för skepps energihanteringssystem

Den maritima transportsektorn ansvarar för en stor del av den globala handeln, vilket har spelat en stor roll i höjningen av levnadsstandarden inom de västerländska ekonomiska områdena. Detta har lyckats genom att möjliggöra global handel till billigare priser. Största delen av de produkter som konsumeras i vår vardag har transporteras över hav i fartyg, såsom containerfartyg, tankfartyg och bulkfartyg [1, 2].

Sjöfartens roll i den internationella handeln och ekonomin är kritisk. Handel och

ekonomiskt utbyte på en global skala är beroende av den maritima transportsektorn. Sjöfarten fungerar som en avgörande komponent i den globala ekonomin. Ett effektivt och hållbart maritimt transportsystem spelar en stor roll för framtida utveckling och ekonomisk stabilitet [1, 2]. Enligt en undersökning av Förenta nationernas konferens om handel och utveckling var den totala volymen gods som transporterades via vatten 11 miljarder ton under året 2021. Detta motsvarar över 80 procent av den totala mängden gods som fraktades internationellt under samma år [3].

Med dagens globaliseringstrender är det viktigt att en både smidig och säker sjötransportindustri prioriteras för att uppfylla de transportkrav som krävs för att upprätthålla möjlig ytterligare ekonomisk tillväxt. För att möta dessa utmaningar bör sjöfartssektorn ta i användning nya växande teknologier såsom sakernas internet, molnbaserade datorsystem, kantdatorsystem, digitala tvillingar, simuleringstekniker och maskininlärning [1].

Sjöfartsindustrin kan öka sin effektivitet, säkerhet och hållbarhet genom att ta i användning dessa teknologier. Dessutom kan transparens inom globala leveranskedjor förbättras, negativa miljöpåverkningar kan sänkas och mera anpassade transportlösningar möjliggöras [1].

Stora mängder data genereras och samlas in från fartyg genom sensorer som är placerade runtom fartyget. Utrustning på fartyg, såsom kranar och motorer vilka ofta produceras av externa tillverkare, innehåller också ett stort antal sensorer [1].

Trots den stora datamängden som samlas, förblir största delen av denna data oanvänd inom sjöfartsindustrin. Kvantiteten av insamlade data indikerar inte att datasamlingsprocessen är effektiv. Istället är det hur den insamlade datan struktureras och utnyttjas för att förbättra design och operation som bestämmer datans värde. Rå sensordata i sig själv ger inte mycket värde. För att göra det möjligt att använda metoder såsom maskininlärningsalgoritmer krävs det att datan först förbehandlas och kategoriseras. Det här innebär att datan struktureras på ett sätt som gör det möjligt att extrahera relevanta insikter och information [1].

I denna avhandling ligger fokusen på användandet av simulering, specifikt samsimulering, för att maximera utnyttjandet av data inom sjöfartsindustrin och förenkla

processen att konstruera digitala modeller av massiva system som marina fartyg. Dessa omfattande digitala modeller kräver en stor mängd expertkunskaper från flera olika områden vilket sällan kan hittas i en enda organisation utan arbetet måste delas upp mellan ett flertal organisationer. Konstruktionen av digitala modeller för marina fartyg innebär inte bara teknisk design och konstruktion, utan innefattar också aspekter som systemintegration, prestandaoptimering och säkerhet. Dessa kräver expertis inom flera olika vetenskapliga och tekniska områden.

Simulering tillåter oss att skapa en abstrakt representation eller modell av ett dynamiskt system. Detta uppnås genom en process av nedbrytning där det globala systemet delas upp i mindre mer hanterbara komponenter. Nyckelfunktioner inom systemet modelleras genom att identifiera en mängd distinkta tillstånd som systemet kan befinna sig i. Dessa tillstånd kan nås från ett eller flera andra tillstånd i mängden, vilket skapar en nätverksstruktur av potentiella systemtillstånd och övergångar mellan dem [15].

Genom att använda denna modell kan vi förstå och förutsäga systemets beteende under en mängd olika förhållanden och scenarier. Detta ger oss insikter som kan användas för att förbättra systemets design, effektivitet och robusthet, samt för att identifiera och lösa potentiella problem eller utmaningar.

Med framstegen inom digitalisering ökar komplexiteten i skeppsinfrastrukturen, vilket kräver en mer distribuerad kompetens över olika delsystem. Med integrationen av digitala teknologier, såsom sensorer och programvara med konventionella delsystem, höjs systemets komplexitet. Denna nödvändiga expertis utkontrakteras ofta till externa leverantörer eller mindre specialiserade enheter vilket ger upphov till utmaningar i att förstå systemets funktioner och att genomföra en fullständig simulering av hela systemet [15].

Inom traditionell simulering och modellering presenterar dessa fördelade miljöer svårigheter, särskilt gällande bristen på standardisering i datautbyte mellan olika verktyg och potentiella rättighetsfrågor. Samsimulering erbjuder en lösning på dessa problem genom en mera anpassningsbar och flexibel metod för simulering av komplexa system. Denna teknik innebär att simuleringen av det övergripande systemet

skapas genom att koppla samman flera modulära delsimulatorer. Varje delsimulator fungerar som en svart låda, vilket innebär att den verkar oberoende av andra komponenter. Denna strategi säkerställer att utvecklarna av dessa delsimulatorer endast behöver specialkunskaper inom sitt specifika område [15].

I den experimentella delen av denna avhandling används verktyg från Open Simulation Platform (OSP). OSP representerar en mängd verktyg för samsimulering med ett särskilt fokus på sjöfartsindustrin; se kapitel 3. För att skapa modeller av de delsystem vi vill simulera tillämpas Functional Mock-up Interface-standarden (FMI). Denna standard används för att skapa en Functional Mock-up Unit (FMU), se kapitel 3.4. En FMU fungerar som en komponent i vår samsimulering och definierar ett flertal inmatnings- och utmatningsvariabler. Dessa variabler är kopplade till de övriga komponenterna i simuleringen. Denna sammankoppling av komponenter bildar en sammansatt simulering, vilket skapar en heltäckande simulation.

Vår modell modellerar ett energisystem ombord på en färja som innehåller ett batterisystem. Det här har splittras till tre separata delsystem, nämligen generator, batteri och energihanteringssystem. Generatorn som styrs av energihanteringssystemet genererar ström genom att aktivera och reglera det nödvändiga antalet motorer baserat på efterfrågan. Batteriet som också hanteras av energihanteringssystemet lagrar och tillhandahåller ström vid behov, samtidigt som dess laddningstillstånd (SoC) strävar till att hålla sig inom definierade gränser för att förlänga dess livslängd. Energihanteringssystemet fungerar som den centrala styrenheten och fattar beslut baserat på den totala efterfrågan samt generatorns och batteriets status. Det bestämmer antalet aktiva motorer, energin de ska producera och om batteriet ska laddas eller användas för ytterligare energi för att möta efterfrågan.

Efterfrågan på systemet är baserad på riktiga data samlad från en resa mellan Åbo och Stockholm med en färja. En av nyckelfrågorna i vårt experiment är att hur placering av elbilsladdare på bildäcket skulle påverka energisystemet. För det skapades en till FMU som modellerar efterfrågan en elbilsladdare behöver då den laddar en bil baserat på graferna i figur 9. FMUn har variabler som kan justera antalet bilar på däck samt antalet bilar som laddas samtidigt.

I den genomförda simuleringen utvärderades tre scenarier för att bedöma energisystemets prestanda under varierande elbilsladdningsförhållanden. Det första scenariot involverade inga elbilar ombord, vilket fungerade som ett grundfall. Resultaten visade en effektiv drift av energihanteringssystemet. Batteriets SoC sjönk till nästan 70 % vid resans början innan det återvände till det övre tröskelvärdet på 80 %. Metanläckage varierade mellan 200 och 400 kg, med något högre värden under den initiala och slutliga tredjedelen av resan; se figur 13.

Det andra scenariot inkluderade 20 elbilar på däck med två bilar som laddades åt gången. Det initiala fallet i batteriets SoC var något större än i det första scenariot då batteriets SoC sjönk till nästan 65 %, troligen på grund av den extra energibehovet från billaddningen. När alla bilar var fulladdade speglade SoC och metanläckagebeteendet i det första scenariot, se figur 14.

I det tredje scenariot med 80 elbilar och åtta som laddas samtidigt, nådde det initiala SoC fallet den nedre tröskeln på 60 %, troligen på grund av det betydligt högre samtidiga laddningsbehovet. Efter det initiala fallet inträffade ett mindre fall till nästan 75 % innan SoC-mönstret liknade de tidigare fallen. Metanläckaget var något högre i början av resan jämfört med andra fall, se figur 15.

References

[1] P.-L. Sanchez-Gonzalez, D. Díaz-Gutiérrez, T. J. Leo, and L. R. Núñez-Rivas, "Toward digitalization of maritime transport?" *Sensors*, vol. 19, no. 4, p. 926, 2019.

[2] M. Fruth and F. Teuteberg, "Digitization in maritime logistics—what is there and what is missing?" *Cogent Business & Management*, vol. 4, no. 1, p. 1411066, 2017.

[3] "Review of maritime transport," United Nations Conference on Trade and Development (UNCTAD), Tech. Rep., 2021.

[4] M. Christiansen, K. Fagerholt, B. Nygreen, and D. Ronen, "Chapter 4 maritime transportation," in *Transportation*, ser. Handbooks in Operations Research and Management Science, C. Barnhart and G. Laporte, Eds. Elsevier, 2007, vol. 14, pp. 189–284. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0927050706140049

[5] S. Madakam, V. Lake, V. Lake, V. Lake *et al.*, "Internet of things (iot): A literature review," *Journal of Computer and Communications*, vol. 3, no. 05, p. 164, 2015.

[6] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[7] M. Plaza-Hernández, I. Sittón-Candanedo, R. S. Alonso, L. C. Martínez-de Iturrate, J. Prieto, K. Kravari, T. Kosmanis, G. Katranas, M. P. Silva, and J. M. Corchado, "Edge computing and internet of things based platform to improve the quality of life of the silver economy on leisure cruise ships," in *2021 International Symposium on Computer Science and Intelligent Controls (ISCSIC)*. IEEE, 2021, pp. 159–163.

[8] A. Felski and K. Zwolak, "The ocean-going autonomous ship—challenges and threats," *Journal of Marine Science and Engineering*, vol. 8, no. 1, p. 41, 2020.

[9] E. Akyuz, K. Cicek, and M. Celik, "A comparative research of machine learning impact to future of maritime transportation," *Procedia Computer Science*, vol. 158, pp. 275–280, 2019.

[10] T. Statheros, G. Howells, and K. M. Maier, "Autonomous ship collision avoidance navigation concepts, technologies and techniques," *The journal of Navigation*, vol. 61, no. 1, pp. 129–142, 2008.

[11] D. Jones, C. Snider, A. Nassehi, J. Yon, and B. Hicks, "Characterising the digital twin: A systematic literature review," *CIRP Journal of Manufacturing Science and Technology*, vol. 29, pp. 36–52, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1755581720300110

[12] Í. A. Fonseca and H. M. Gaspar, "Challenges when creating a cohesive digital twin ship: a data modelling perspective," *Ship Technology Research*, vol. 68, no. 2, pp. 70–83, 2021.

[13] A. Coraddu, L. Oneto, F. Baldi, F. Cipollini, M. Atlar, and S. Savio, "Data-driven ship digital twin for estimating the speed loss caused by the marine fouling," *Ocean Engineering*, vol. 186, p. 106063, 2019.

[14] M. Schirmann, M. Collette, and J. Gose, "Ship motion and fatigue damage estimation via a digital twin," in *Life Cycle Analysis and Assessment in Civil Engineering: Towards an Integrated Vision*. CRC Press, 2018, pp. 2075–2082.

[15] C. Gomes, C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe, "Co-simulation: a survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 3, pp. 1–33, 2018.

[16] H. Bossel, *Modeling and simulation*. AK Peters/CRC Press, 2018.

[17] B. P. Sullivan, S. Desai, J. Sole, M. Rossi, L. Ramundo, and S. Terzi, "Maritime 4.0–opportunities in digitalization and advanced manufacturing for vessel development," *Procedia manufacturing*, vol. 42, pp. 246–253, 2020.

[18] S. Skjong, "Modeling and simulation of maritime systems and operations for virtual prototyping using co-simulations," 2017.

[19] "Official webpage for the fmi standard." https://fmi-standard.org/, accessed: 2023-01-23.

[20] V. Liberatore and A. Al-Hammouri, "Smart grid communication and co-simulation," in *IEEE 2011 EnergyTech*, 2011, pp. 1–5.

[21] M. Trčka, J. L. Hensen, and M. Wetter, "Co-simulation of innovative integrated hvac systems in buildings," *Journal of Building Performance Simulation*, vol. 2, no. 3, pp. 209–230, 2009.

[22] L. I. Hatledal, R. Skulstad, G. Li, A. Styve, and H. Zhang, "Co-simulation as a fundamental technology for twin ships," 2020.

[23] S. Skjong, B. Taskar, E. Pedersen, and S. Steen, "Simulation of a hybrid marine propulsion system in waves," in *Proc. 28th CIMAC World Congr.*, 2016, pp. 1–16.

[24] S. Skjong and E. Pedersen, "Co-simulation of a marine offshore vessel in dp-operations including hardware-in-the-loop (hil)," in *International Conference on Offshore Mechanics and Arctic Engineering*, vol. 57731. American Society of Mechanical Engineers, 2017, p. V07AT06A038.

[25] Ø. R. Smogeli, K. B. Ludvigsen, L. Jamt, B. Vik, H. Nordahl, L. T. Kyllingstad, K. K. Yum, and H. Zhang, "Open simulation platform–an open-source project for maritime system co-simulation," in *19th International Conference on Computer and IT Applications in the Maritime Industries*. Technische Universität Hamburg-Harburg, 2020.

[26] D. C. Black and J. Donovan, *SystemC: From the ground up*. Springer, 2004.

[27] S. Documentation, "Simulation and model-based design," 2020. [Online]. Available: https://www.mathworks.com/products/simulink.html

[28] F. Perabo, D. Park, M. K. Zadeh, Ø. Smogeli, and L. Jamt, "Digital twin modelling of ship power and propulsion systems: Application of the open simulation platform (osp)," in *2020 IEEE 29th International Symposium on Industrial Electronics (ISIE)*. IEEE, 2020, pp. 1265–1270.

[29] T. Blockwitz, M. Otter, J. Akesson, M. Arnold, C. Clauss, H. Elmqvist, M. Friedrich, A. Junghanns, J. Mauss, D. Neumerkel *et al.*, "Functional mockup interface 2.0: The standard for tool independent exchange of simulation models," in *Proceedings*, 2012.

[30] A. Junghanns, C. Gomes, C. Schulze, K. Schuch, R. Pierre, M. Blaesken, I. Zacharias, A. Pillekeit, K. Wernersson, T. Sommer *et al.*, "The functional mock-up interface 3.0-new features enabling new applications," in *Modelica Conferences*, 2021, pp. 17–26.

[31] D. Radan, "Integrated control of marine electrical power systems," 2008.

[32] O. Alnes, S. Eriksen, and B.-J. Vartdal, "Battery-powered ships: A class society perspective," *IEEE Electrification Magazine*, vol. 5, no. 3, pp. 10–21, 2017.

[33] F. W. Maren Hansen, Tomas Tengner, "Polaris icebreaker study: improving performance and cutting emissions through energy storage," 2020.

[34] M. Kane, "Insideevs blog post." https://insideevs.com/news/519382/tesla-model3-82kwh-charging-analysis/, accessed: 2023-03-29.