Multiple Worlds

A multi-actor simulation-based design method for
logistics systems

Multiple Worlds

A multi-actor simulation-based design method for logistics systems

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. ir. K.Ch.A.M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op 29 juni 2011 om 10.00
door

Michele FUMAROLA

Master of Science in de Informatica
geboren te Maasmechelen, België.

Dit proefschrift is goedgekeurd door de promotor:

Prof. dr. ir. A. Verbraeck

Samenstelling promotiecommissie:

Rector Magnificus	voorzitter
Prof. dr. ir. A. Verbraeck	Technische Universiteit Delft, promotor
Prof. dr. H. Vangheluwe	Universiteit Antwerpen
Prof. dr. ir. P.M. Herder	Technische Universiteit Delft
Prof. ir. J.C. Rijsenbrij	Technische Universiteit Delft
Prof. dr. H.G. Sol	Technische Universiteit Delft
Dr. L. Yilmaz	Auburn University
Dr. M.D. Seck	Technische Universiteit Delft
Prof. dr. F.M.T. Brazier	Technische Universiteit Delft, reservelid

ISBN 978-90-8570-773-8

Dedicato a chi mi ha dato la vita e a chi le ha dato un senso aprendone la strada alla conoscenza,
a Comasia, Pasquale e Marino.

Acknowledgements

Writing the acknowledgements of a Ph.D. dissertation provides the perfect opportunity to reflect on the experiences lived during the last four years. A dissertation only documents a fraction of a person's way to becoming a scientist limited to the final part, namely the polished end result of the research. The most important part of a person pursuing his Ph.D. remains mostly undocumented: the interactions with the people that provided inspiration, those who were constantly critical to improve the final result and the people that kept encouraging during the most difficult periods.

I consider myself lucky to have found a promoter, Alexander Verbraeck, that allowed me to make mistakes and was able, each time, to head me in the right direction. Thank you, Alexander; I truly enjoyed working under your supervision. I want to thank Mamadou Seck for every single discussion concerning my research: the challenges he posed each time taught me a lot and helped me improve my work immensely. I would like to thank Cornelis Versteegt for providing me the opportunity for carrying out part of my research at APM Terminals: his openness and availability has helped me tremendously in understanding the "real world" part of my research.

Several things changed in my life throughout the last four years, but everything pales in comparison to the Systems Engineering section. I want to thank former colleagues that were present during my first year at the section: Stijn-Pieter van Houten for motivating me to promptly focus on my research direction, and Gwendolyn Kolfschoten and the late Sam Muniafu for helping me improve my research proposal. Together with Nong Chen, Yan Wang, Roy Chin, Elisangela Kanacilo, and Rafael Gonzalez, they made me feel welcome in a stimulating environment. I would especially like to thank Rafael for the numerous discussions we had throughout the 3 years we spent as colleagues, which helped me better understand several aspects of carrying out scientific research. During my initial year, many Ph.D. students completed their research and left, providing the opportunity for several newcomers to join the section. One of the first was Michiel Renger: I truly enjoyed sharing the office with him for the numerous interesting discussions and for his overall enthusiasm.

He left our section, but luckily the influx of new Ph.D. students did not stop. I would like to thank Jan-Paul van Staalduinen and Kassidy Clark, who joined later, for making the last few years truly pleasant. I would also like to thank Ronald Poelman: I admire his insatiable interest for everything related to virtual and augmented reality (and technology in general) and enjoyed our numerous collaborations on papers and projects. I also enjoyed a lot working with Yilin Huang and Çağri Tekinay on simulation related research. Together with Jordan Janeiro and Evangelos Pournaras, they also made sure I had plenty of opportunities to relax after working hours. I would like to thank Michel Oey, Martijn Warnier, Joseph Barjis, Stephan Lukosch, and other colleagues of the Systems Engineering section for discussing and helping me with aspects of my work on multiple occasions and Sabrina Rodrigues for helping me each time with every administrative matter.

I would also like to thank Akihiro Yonehara of APM Terminals for a pleasant collaboration during the "Virtual Terminal" project and Rich Ceci for hosting me at APMT's Virginia terminal and for his valuable input during several stages of my research.

Undertaking a Ph.D. "far" from home, allowed me to meet many people that are expats as I am. I would like to thank them for making the past four years a memorable experience. I would also like to thank Tim Tutenel for undertaking a Ph.D. with me in Delft which resulted in spending the last 10 years as either a fellow student or a colleague.

I'm indebted to my family for their love and support. I would like to thank my sister, Maria, and her husband, Danny, for always being there for me. I hope this dissertation can serve as an inspiration to her daughter, Nicla.

When I started my Ph.D. in the summer of 2007, I thought that would be the event that, for me, would coin that year. Little did I know this would change quickly... *Manu, grazie di esistere.*

Delft, *Michele Fumarola*
April 2011

Contents

List of Abbreviations

AGV	Automated Guided Vehicle
ASC	Automated Stacking Crane
AStC	Automated Straddle Carrier
CAD	Computer Aided Design
CAPEX	Capital Expenditure
COTS	Commercial Off-the-shelf
DEVS	Discrete Event Systems Specification
DSISR	Design Science in Information Systems Research
GUI	Graphical User Interface
IS	Information Systems
KPI	Key Performance Indicator
M&S	Modeling and Simulation
OPEX	Operational Expenditure
OR	Operations Research
QC	Quay Crane
RMG	Rail Mounted Gantry Crane
SE	Systems Engineering

SES	System Entity Structure
SSM	Soft Systems Methodology
TEU	Twenty-foot equivalent
TOS	Terminal Operating System
XML	Extensible Markup Language

Chapter 1

Towards a multi-actor simulation-based design method for logistics systems

Today's globalized society relies on logistics systems: products are distributed throughout the whole world in just a few days, permitting Europeans to drive Asian cars and Americans to boast European *haute couture*. Nowadays, we take this for granted, yet complex systems are in place to keep things running. Logistics systems come in all shapes and sizes: the air network connects continents, the train infrastructure spans complete countries, assembly lines keep modern factories running, and sushi aficionados get their favorite dish served on conveyer belts.

Engineering methods are applied to design logistics systems of which every single detail is carefully analyzed, leaving little room for errors. Yet, errors do occur. Heathrow's new faulty baggage handling system halted the operation of one of the busiest airports in the world [BBC08]. Inefficient cargo handling cripples the import/export capabilities of Tanzania [Mac10]. Inadequate account of needed capacity will severely affect the United Kingdom's public transportation system during the next 30 years [BBC09]. These are just a few examples of what happens when logistics systems fail to work properly. The failures become painfully clear through their economical and societal impact.

Increasingly better methods, models, and techniques are being developed to avoid potential failures. Mathematical models, design methods, prototypes, and evaluation and test methods are just a few examples of methods, models, and techniques that guide designers and developers during a complete project that leads to a logistics system. The methods, models, or techniques to choose from that suits best to the given project, depend largely on the logistic system at hand. So, what are logistics systems and on which specific systems do we focus in this research?

The Council of Supply Chain Management Professionals defines logistics as the "process of planning, implementing, and controlling the efficient, effective flow and storage of goods, services, and related information from point of origin to point of consumption for the purpose of conforming to customer requirements" [Cou10]. Logistics systems are ubiquitous, therefore a demarcation of the scope of our research is required. In this research, we will focus on logistics systems that are large in scale, highly technical and that involve many actors in the design process. Actors are parties that have their own interests and concerns in the design process [Her05]. The precise implications of this demarcation will become clear in the next chapters.

1.1 Engineering logistics systems

The term 'design' often invokes the image of esthetically pleasing artifacts created by the hand of artists. Yet, engineers design day in, day out. Although the latter are less interested in pleasing the eye, they do want to satisfy some specific design requirements. Alexander [Ale64] poses that form is the final object of design and the problem of design is to fit form to its context. Form is an important part of the design over which designers have control, whereas the context puts demands on the form. In logistics systems, engineers design such things as baggage handling systems, cargo handling terminals, and urban rail networks whereas the large numbers of passengers and goods are the context which shapes the form.

In engineering, designing systems is part of the field of 'systems engineering' (SE). Either implicitly or explicitly, theories and methods from SE are used in large logistics projects such as container [Age04] and airport [Hor10] terminals. During the last couple of decades, SE brought about a change in how technical systems are designed and implemented. The concern of SE is on the whole system instead of concerning on the individual parts. When we take a broad look on complete engineering processes, we can often identify a set of steps such as requirements analysis, design, implementation, and testing. These steps can be taken sequentially with or without feedback possibilities, and one can also iterate through all steps multiple times [Sag00, Hit08]. To illustrate the application of SE methods on the design logistics systems, we will shortly discuss them as applied on warehouses, on airports, and on infrastructures.

1.1.1 First case: warehouse design

Corporations such as Amazon and Dell rely heavily on a distributed network of strategically located warehouses. The use of warehouses depends on supply chain strategies that take into consideration a number of factors such as lead times preferred by customers, and storage costs of products (including devaluation of value). Whatever the reason may be, the individual warehouses are small logistic systems within the supply chain and they need to be designed to efficiently function within the overall supply chain.

In constrast of the importance of warehouses, Baker and Canessa [Bak09] report the lack of a systematic approach for designing and developing warehouses. The authors carried out a literature review and backed it by a survey among practitioners that brought to light a number of different approaches. Hereafter, they identified similarities to be used in a novel, formal approach. This formal approach presents similarities with traditional SE processes with the following steps identified by the authors:

Requirements and Architecture *Define system requirement* such as business strategy requirements and constraints from the environment. This needs to be done by taking into consideration the role of the warehouse within the overall supply chain. *Define and obtain data* by using checklists to collect all necessary data. Such lists include product details, order profiles, goods arrival and dispatch patterns, cost data, and site information. *Analyse data* to be brought together into comprehensive planning bases for a number of planning horizons. Various statistical techniques can be used to carry out this phase.

Design *Establish unit loads to be used* to optimize for the whole supply chain. This decision is mostly taken based on educated guesses by experienced engineers. *Determine operating procedures and methods* for each function of the warehouse by using techniques such as checklists, warehouse zoning, technology assessment charts, concept library, and standard work procedures. *Consider possible equipment types and characteristics* based on costs and performance measures using techniques such as simulation, decision trees, and sensitivity analysis. *Calculate equipment capacities and quantities* based on the results of the previous step. *Define services and ancillary operations* based on checklists of requirements. *Prepare possible layouts* in specialized software environments or general purpose design environments such as AutoCAD. Specific methods can be used to structure the layout generation.

Evaluation *Evaluate and asses* to validate the operational and technical feasibility of the design. Simulation is by large the preferred technique in this step. *Identify the preferred design* from the designs generated in the previous steps.

Although Baker and Canessa [Bak09] describe a large number of steps, they do not describe what would happen from the implementation phase onwards. In traditional SE models such as the waterfall model [Roy70], the steps described above would fit in the conceptual, analysis and design phases. Nevertheless, this case indicates the importance of these phases in SE.

1.1.2 Second case: low-cost airline facilities

The increasing market for low-cost airlines is changing demands in terms of airport facilities. Whereas traditional airlines prefer massive centralized hub airports, low-cost airlines strive for "either secondary airports in a metropolitan multi-airport system, or distributed destinations that by-pass the use of a centralized metropolitan hub" [Neu08]. This shift in the airline business implies a different way of handling new airport projects to face the market demands. The design process of airports changes drastically to take into consideration the uncertainty of future developments (technical, financial, industrial, political, or others). Flexibility in design thus becomes a prominent factor of the entire process. De Neufville [Neu08] describes the design of "traditional" airports as follows:

1. Identify the most likely forecast in terms of market and business demands. Specify the requirements based on these forecasts.

2. Design a master plan that fully satisfies the requirements identified in the first step.

3. Execute the master plan (implementation).

The author continues by discussing the introduction of flexibility to adhere to new market and business demands, which results in a fundamentally different design process that consists of three basic elements

1. Recognizing a range of possible outcomes results in designs that take into consideration the bad outcomes as well as the more favorable outcomes.

2. Designs that allow the possibility of adapting the facilities easily once it has been implemented. This would result in an evolving facility depending on demand.

3. Analysis of the development strategies to consider the various phases of the design and what the initial facility should be in order to easily carry out the subsequent phases.

The three basic elements are introduced to face the uncertainty of different actors including the market, the airline companies, and policy makers. The latter two have to adapt to the first that requires lower prices and less hassle to consume the service.

1.1.3 Third case: infrastructure interactions in expanding regions

Meyers *et al.* [Mey08] describe an infrastructure program consisting of transport, water, energy, telecommunications, and social infrastructure. The premise of their work is that infrastructure interaction needs to be taken into account from the beginning of the program in order to minimize negative impact. Infrastructure interactions can be understood as timing of related projects to co-locating infrastructure, interactions within the physical or stakeholder network and interactions within the institutional framework surrounding the subsystems.

Meyers *et al.* [Mey08] present the typical infrastructure planning process based on Goodman and Hastak [Goo06], that is structured as follows:

- Establishment of goals and objectives

- Problem identification and analysis

- Solution identification and impact assessment
- Formulation of alternatives and analysis
- Recommendations
- Decisions
- Implementation
- Operation and management

According to Meyers *et al.* [Mey08], there are two major problems with this approach: planning theory does not correspond to practice and planning is not a single actor activity. Based on Mintzberg [Min94] and De Bruijn and Ten Heuvelhof [Bru00], they argue that planning should be used as a communication method and as a way to control internal and external actors. Furthermore, a sense of urgency is needed to commit the actors to the decision making process, openness and integrity is needed to create trust, incentives are required, and, finally, process agreements are necessary to keep the substance of the decisions made in order.

Decision making processes in multi-actor environments are different from single actor decision making processes: the process needs to be structured differently in order to achieve 'acceptable' results. The so called network approach is needed to deal with the actors involved in the process.

1.1.4 Characterizing system design processes

In the three cases, we discussed two important aspects of SE processes: methods with accompanying support tools and a multiple actors involved in the process. In the first case, simulation was used to design the system. Indeed, following Shannon [Sha98] and Sol [Sol82] simulation can be seen as a method of inquiry that provides the possibility of designing systems, existing or not. To study the kind of logistics systems we discussed so far, the dynamic nature of the system needs to be taken into account. Simulation provides the quantitative measures to study the behavior of the system and make informed decisions about its design. Chance *et al.* [Cha96] state:

> Simulation captures both dynamic (time-dependent) and stochastic (random) behavior. Simulation is sometimes used in the early stages of a project to help understand how the system works. It has intuitive appeal for managers, especially when animation is used, because they can 'see' what is going on in the model. However, there are several disadvantages to using simulation models. They typically take much longer to run than analytic models, and the results from a simulation model can be difficult to interpret. Statistical analysis of the output is necessary, because each simulation run represents a single possible sample path. For these reasons, simulation can be an expensive option."

The second and third case highlight the presence of different actors that influence the design. De Bruijn and ten Heuvelhof [Bru08] characterize actors based on standpoint (proponent, opponent, fence sitter) and on power (production, blocking, and diffuse). In both cases, we can identify different actors such as clients, policy makers, and problem owners. In the following two sections, we will study both the actors involved in the design process and how simulation is used as a method of inquiry.

1.2 Designing systems in multi-actor environments

The design process of logistics systems encounters an increase in complexity once multiple actors are involved. Actors are not only present in the system (e.g. company) but are also external parties that do profit directly from the system, for instance public opinion and authorities [Don95]. Xia and Lee [Xia05] attribute the increased complexity to the number of actors involved in the process, the number of stakes and perspectives present, how the different actors relate to each other, the dynamics in the multi-actor environment due to the interaction between actors, and, finally, the uncertainty with respect to actor behavior. During the process, actors interact with each other and with the systems surrounding them [Lei10]. Pruyt [Pru10] pinpoints that the interaction between actors can lead to complex decision-making processes and to unforeseen/unintended effects. Moreover, different forms of organization of actor interactions and different degrees and forms of participation may lead to different outcomes.

According to de Bruijn *et al.* [Bru00, Bru08, Bru09], actors are peculiar due to their reflective nature. This gives them the ability to learn, which has three significant implications:

1. Actors display strategic behavior to serve their own interests and realize their own objectives. This can be achieved by keeping hidden agendas, providing misinformation, etc.

2. Actors learn how to neutralize the interventions of others by creating strategies to sidestep those interventions.

3. Due to the reflective nature of actors, an understanding of the process of interaction is required in order to take a decision. This means that partial understanding about the system is often not sufficient.

In the design process of logistics systems, actors can be found either in the system under investigation (e.g. a supply chain with manufacturers, distributors, and clients; all with own interests and stakes) or outside the system, as designers of the system or stakeholders. The first would be a multi-actor system, the latter a design process in a multi-actor environment (also multi-actor context). A third case, would be a combination of the two, in which a multi-actor system is designed in a multi-actor environment. In this research, we choose to focus on technical systems designed in a multi-actor environment. Such systems could be material handling systems and transportation systems.

Designing in multi-actor environment implies a collaborative design process. Collaborative design is defined as "purposeful joint effort to create a solution" [Pii09]. This means that although typical actor behavior emerges as described above, the main focus of the design effort still remains: jointly solving a problem though a process of finding a satisfying solution. Unfortunately, this does not happen without hurdles. Piirainen *et al.* [Pii09] identified the challenges that present themselves in collaborative design process. From these challenges, we present the ones that we consider to be relevant for this dissertation (quoted from Piirainen *et al.* [Pii09]):

Creating understanding : ensuring shared understanding and mental models of the problem, current state of the system, and envisioned solution.

Satisfying quality : balancing individual requirements and joint quality constraints while making design choices

Organizing interaction : effective organization beyond project management, facilitating interaction between the actors, achieving rationality in the process and finding ways and means to work effectively

The fact that logistics systems are designed in a multi-actor environment has two important implications. Firstly, the fact that actors display strategic behavior to serve their own interests and realize their own objectives means that they might not all agree on what constitutes the optimal design for a given logistics system. Secondly, collaboration processes involving various actors face the challenges presented by Piirainen *et al.* [Pii09]. These implications need to be taken into account in design methods that are used in multi-actor environments.

1.3 Using simulation models in systems engineering

Following [Sag00], a SE process can be viewed as a sequence of system definition, system development, and system deployment phases, each composed of formulation, analysis, and interpretation steps. The traditional role of simulation models has been in the analysis step of system definition, to evaluate design alternatives synthesized in the formulation step. Models of some of these design alternatives are built and experimented with in order to forecast key performance indicators (KPI) in relation with a predefined value system. Typically, the number of alternatives effectively assessed through simulation is small for various reasons including the following: (a) setting up a simulation model for a complex system is costly and time consuming [Car04], (b) when the simulation work is outsourced, the external analysts need some time to reach sufficient proficiency on the problem at hand [Cla95], (c) models are generally built from scratch, given the low reusability levels offered by commercial off-the-shelf (COTS) simulation packages [Rob04]. Simulation models are much less used in the other phases of SE, in particular in the actual synthesis of design alternatives. Despite these limitations, the use of simulation models has greatly contributed to engineering complex systems, as many success stories can demonstrate [The09, Fry95].

Oren *et al.* [Ö06] discuss the synergy between modeling and simulation (M&S) [Zei00] and SE calling for a paradigm shift in the study of large and complex systems based on simulation models. The advantages that would result from the possibility of analyzing more alternatives and a better integration of simulaiton models in the system formulation phase could yield numerous benefits. Modeling helps clarifying the issue and can turn implicit mental models into explicit and tractable information for all actors, thus fostering shared understanding and communication. A simulation model is an instrument for formally and unambiguously expressing knowledge or hypotheses about the system to be designed. Model specification in itself forces the analyst to get rid of most imprecision. As a result, an early use of simulation models could permit to integrate the synthesis and analysis steps into one unified, creative, and explorative activity where alternative systems could be developed with models in-the-making and evaluated on the fly, leading to new vistas and supporting iterations and bifurcations in the design trajectory.

This convergence between M&S and SE is not surprising given the fact that both disciplines have their roots in two threads of the same systems movement [Ö71, Goo57]. Within Zeigler's framework for M&S, design and analysis are described as two fundamental systems problems articulated with Klir's levels of system epistemology [Kli85, Zei00]. System design is about climbing the levels of system specification, i.e. synthesizing the system from components. Analysis, on the other hand, is about going from the structure system level to the components level in order to understand their behavioral characteristics.

These synergistic opportunities have also been identified by a number of authors advocating simulation-based design of complex systems. Paredis and colleagues [Par01] introduced a new simulation method to support a design process based on reconfigurable models where highly abstracted components can be iteratively refined when more detail becomes available in the design process. The paradigm was applied to the design of mechatronic systems. The capability of structure modification was considered by the authors as the key feature for a simulation formalism to be able to support a design process. Another important feature of their paradigm is modularity, facilitated by port-based communication. Once this approach is adopted, reusability of components becomes highly enhanced. Being able to reuse components is critical to the simulation-based design endeavor: building valid components is time consuming, thus a library of valid standard components is required.

Peak and colleagues [Pea07] propose a similar approach based on OMG SysML [OMG10]. In their method, much emphasis is put on building blocks, composability, and reusability of model components. Their contribution focuses on representing system level constrains between components in terms of SysML parametrics, and managing the interoperability with various external tools such as Computer Aided Design (CAD) packages and mathematical solvers. Other notable efforts towards simulation-based design can be found in

[Zei05], [Hu05b] and [Zia07], all based on the Discrete Event Systems Specification (DEVS) formalism and exploiting its hierarchy, modularity, and composability.

Advances on the use of simulation models during design processes discussed so far seem to neglect the presence of multiple actors that need to design a system. As discussed in Section 1.2, the presence of multiple actors that have differing interests and concerns in the design process poses new implications to be taken into account in a design method. In the following section, we will identify the research opportunity that presents itself by including the support for multiple actors in a simulation-based design method.

1.4 Identifying research opportunities

Jacobs [Jac05] distinguishes, based on findings from Hlupic [Hlu93] and Tewoldeberhan *et al.* [Tew02], two types of simulation environments. The first type of simulation environment is built to support one actor to use one simulation model designed by one simulation model designer to conduct one experiment. He calls this the $1-1-1$ paradigm: one stakeholder on one computer system in one location. With the D-SOL simulation environment [Jac05], he introduces the second type of simulation environment, the so called $N_m - N_n - N_o$ environment: N_m stakeholders on N_n computer systems in N_o locations.

During system design, $N_m - N_n - N_o$ environments do not suffice. The use of simulation models is often present during a system design process although it is mainly restricted to a rather limited set of designs that have been declared most promising. The use of simulation models in SE is classically presented to evaluate a system design. The advent of simulation-based design changed the role of the use simulation models towards being a ubiquitous part of the SE process. The use of simulation models is promoted in the early stages of the system engineering process: from requirements specification to implementation. According to Aughenbaugh and Paredis [Aug04], the use of simulation models throughout the design process is essential to cope with the lack of knowledge about the system's requirements, the system's environment, the future design decisions, and emergent attributes.

But what about the actors involved in the process? Multiple actors are involved and all of them have a different opinion on what the best design should be. All of them also need to explore the design space and will display strategic, opportunistic, blocking, and reflective behavior [Bru00]. This means that the probability of having every actor agreeing on the same design is rather low.

The behavior actors display is bound to the fact of them being human beings. Simon [Sim96] states that human reasoning is boundedly rational, which means that humans do not seek optimal solutions but rather find satisfactory solutions, which he calls *satisficing*. Actors display this behavior for a number of reasons such as incomplete information (which is possibly not available), insufficient resources to optimize (some optimums may be even impossible to compute), and time pressure.

From the previous sections, we can conclude that simulation is required to study the behavior of dynamic systems, such as logistics systems. Simulation is needed in the design process of such systems, however, the use of simulation models as understood right now in SE does not fully cover the needs of human actors. This provides the research opportunity that we are looking for in our research, which will be presented as a research objective and research questions in the following section.

1.5 Research objective and questions

The discussion so far brought us to acknowledging the presence of shortcomings in the use of simulation models in the design process of large logistics systems. Current SE methodologies that incorporate the use of simulation models do so by using the simulation models for the analysis of the design that came out of a complete design process. There is thus a clear distinction between the synthesis and analysis phase in a design process. The final design is the product of individual design decisions that have been taken during the process and the analysis encompasses that whole product. To provide insight into the effect of the individual decisions, one has to be able to analyze each individual factor instead of the total product.

The use simulation models to support the design process wherein multiple actors are present, requires an information system. Information systems are generally seen as the set of actors and processes that are supported by various ICT-tools[1] to achieve a certain goal [Dic68]. So far, we have framed our research in terms of methods in the information systems (IS) domain. Following Brinkkemper [Bri96], we define a method as "an approach to perform a systems development project, based on a specific way of thinking, consisting of directions and rules, structured in a systematic way in development activities with corresponding development products".

The contribution we are aiming at in this research, is to provide the means to use simulation models during the design process instead of at the end of the process. Early and quick assessments, which are carried out using simulation models, should provide actors the possibility to explore different possibilities instead of focusing on the design possibility that seems most promising, given the past experiences and strong opinions. Rephrasing this brings us to our research objective:

Research Objective. *To provide a method that uses simulation models to support multi-actor design processes of logistics systems.*

The research objective cannot be achieved at once. A set of research questions are posed that lead us through our research. Firstly, we need to know what the current state-of-the-art is in M&S to support the design process of logistics systems. This will shed light on the current shortcomings that are perceived in using simulation models in design processes.

[1] Information and communication technologies tools

Research Question 1. *What are the current methods used for designing logistics systems in practice?*

Simulation models have been used extensively in SE. Many theories can aid us in achieving our objective. Nonetheless, we need to select a filter to restrict ourselves to theories we can actually use. This leads to the following question:

Research Question 2. *What are the issues in current multi-actor design methods that use simulation models?*

To design the method formulated in our research objective, we have to ask the following research question:

Research Question 3. *How can a multi-actor design method that uses simulation models address the issues identified in RQ2?*

1.6 Research strategy

Research in IS is socio-technical in nature and over the last decades, a methodological pluralism has developed within the IS field. Throughout the past three decades, many research approaches have been developed that are presented to be apt for conducting research in IS. We subdivide our research approach in three main parts:

Research philosophy Before conducting scientific research, we need to have an understanding of what we perceive to be reality, and how we interpret this reality.

Research methodology A research methodology enables rigorous and transparent science to be conducted following a well defined set of steps.

Research instruments We employ specific research instruments that are apt for each specific step of our research methodology.

1.6.1 Research philosophy

The methodological pluralism found in the IS field is characterized by different ontological and epistemological assumptions. Becker and Niehaves [Bec07] state that the methodological pluralism is still an open issue in IS, meaning that there is still not a binding theory of philosophy of science. Table 1.1 documents the most commonly used ontological and epistemological assumptions in IS. The different ontological and epistemological assumptions are taken from IS research projects [Gon10, Hou07, Jac05], surveys on the subject matter [Dub03, Wad04, Wal95b, Che04], and from literature discussing the subject matter [Fal99, Web04, Nan97, Bec07, Hir03, Ric07]. It is interesting to note, given our research subject, that this classification bears resemblance to Flood's onto-epistemological view on systems [Flo90], which is pointed out in the table as well.

Ontology	External realism or Flood's *close ontology*	"There exists a reality totally independent of our representations of it." [Dob01, Sea95]
	Internal realism or Flood's *distant ontologist*	"Reality is an inter-subjective construction of the shared human cognitive apparatus" [Wal95b] Reality is not discourse independent because "a sign that is actually employed in a particular way by a particular community of users can correspond to particular objects within the conceptual scheme of those users. 'Objects' do not exist independently of conceptual schemes. Since the objects and the signs are alike internal to the scheme of description, it is possible to say what matches what." [Put81]
	Critical realism or Flood's *beyond the horizon ontologist*	"Our knowledge of reality is a result of social conditioning and, thus, cannot be understood independently of the social actors involved in the knowledge derivation process. However, it takes issue with the belief that the reality itself is a product of this knowledge derivation process." [Dob02]
	Idealism or Flood's *visionary ontologist*	"Reality is dependent on our perception of it." [Wys04]. It springs from Kant's 'transcendental' idealism [Kan99], which poses the universal conditions for the possibility of knowledge are that our minds structure our experience of reality in terms of space time and causality [Min04a]
	Subjective idealism or Flood's *psychic prison ontologist*	"Reality as a personal construction of each individual." [Dob04]
Epistemology	Positivism	"Facts and values are distinct and scientific knowledge consists only of facts."[Wal95b]
	Interpretivism	"Facts and values are intertwined, both are involved in scientific knowledge."[Wal95b]
	Critical approach	"The critical researcher attempts to critically evaluate and transform the social reality under investigation"[Orl91]

Table 1.1: The ontological and epistemological assumptions commonly used in the IS field.

In this research we aim at providing a design method for technical systems carried out in multi-actor environments. We postulate that these kind of socio-technical systems "cannot be understood independently of the social actors involved in the knowledge derivation process" [Dob02]. Therefore, we choose to take the ontological stance of critical realism wherein "real-world mechanisms are viewed as producing actual events that can be empirically observed, while at the same time the resulting production of knowledge is viewed as a human, socially and historically conditioned activity" [Gon10].

As noted, we believe IS need to be studied in their totality: the supporting information technology and the actors involved in the activity, without diminishing the importance of their personal, social and cultural values. To understand the complete intervention that an information system imposes on an environment, we believe we need to create a rich picture that encompasses the artifact and the actors that are influenced by the artifact. An interpretivist stance in terms of epistemology provides the means to create this rich picture using the qualitative instruments that are commonly associated with interpretivism. In IS research, this epistemological stance has gathered acceptance although, traditionally, the positivistic view still prevails [Wal95a, Che04]. Iivari [Iiv07] and Niehaves [Nie07] argue that taking a clear epistemological stance is required to judge the validity and quality of scientific research.

1.6.2 Research methodology

In the field of IS, advances are made in framing a methodology that differentiates behavioral sciences from design science. Based on Simon's call for 'a science of the artificial', the first one focuses on 'what is' whereas the second one tries to explore 'what ought to be'. According to Gregor [Gre06], five types of theories exist: (i) theory for analysing; (ii) theory for explaining, (iii) theory for predicting; (iv) theory for explaining and predicting; and (v) theory for design and action. Whereas the natural and behavioral scientists tend towards the first four types of theories, theorizing in IS would predominantly be situated in the fifth type. This brought us to a research methodology that has been firstly introduced by March and Smith [Mar95] and later refined by Hevner [Hev04, Hev10]. This has been coined as 'design science in information systems research' (DSISR).

The DSISR methodology proposes three cycles: the relevance, the rigor, and the design cycle [Hev07]. The balance between the relevance and rigor cycle enforces the researcher to focus on real-life problems using grounded scientific theory. However, exceptions may be tolerated: Iivari [Iiv07] states that it would be possible to focus on new opportunities without starting from identified problems. In this case, the relevance cycle would go about proving that the new opportunity is an improvement for the selected (business) environment. A schematic view of the DSISR methodology is provided in Figure 1.1.

Although not strictly so, we initiate a research project from the business environment. It provides the requirements from the real-life (business) environment and the acceptance criteria that are needed to carry out the evaluation once the artifact has been designed. The rigor cycle provides the means in terms of theories and methods to serve as input for the design cycle. Additionally it provides domain specific experiences and expertise and also the existing artifacts that are already being used in the application domain. The rigor cycle must guarantee the the presence of innovative aspects of the research: this is what distinguishes DSISR from the regular practice of building IT artifacts [Iiv07]. Together both cycles should guarantee, if applied correctly, that a DSISR research solves a problem that is relevant to the application domain, one that has not been solved before and using rigorous scientific theories and engineering methods.

The design cycle forms the crux of the methodology: an artifact is designed and evaluated in an iterative fashion taking as an input information and knowledge that has been produced in the rigor and relevance cycle. During the design cycle, generic methods and reference models [Win09] are evaluated through their instantiations [Mar95]. Although a distinction is made between methods and models, it can be argued that they are 'two sides of the same coin' [Win09]. Whereas the generic methods focus on the activity, the reference

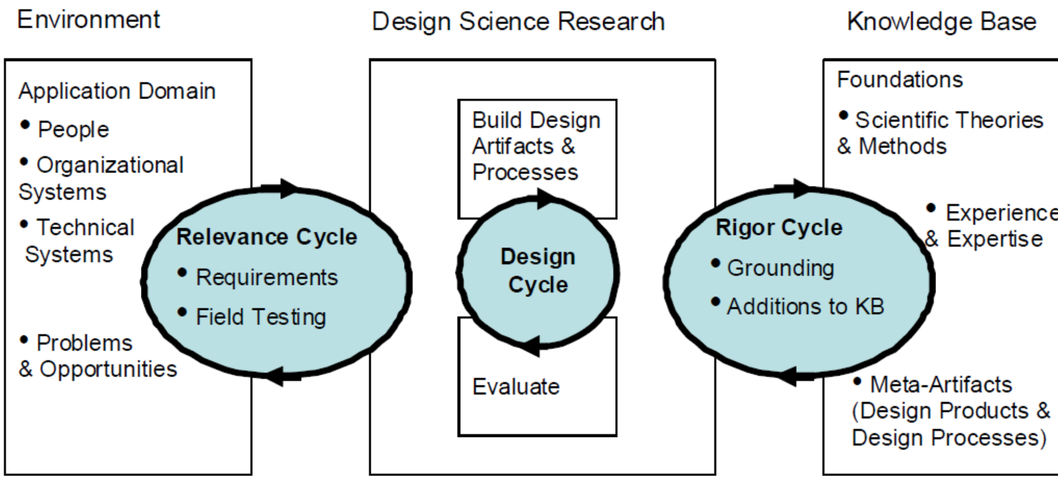


Figure 1.1: Design science research in information systems guides researchers along three cycles to design artifacts that are both scientifically rigorous and relevant to a chosen application domain.

models focus on the result. Nevertheless, they both try to accomplish a complex IS artifact. In DSISR, a peculiar and controversial role is given to theory: should the outcome of DSISR be restricted to artifacts (in terms of construct, methods, models, and instantiations) or should it also result in new theory? According to March and Smith [Mar95], theorizing is part of the natural sciences and should not be sought in design science. However many disagree on this particular issue [Nun91, Wal92, Ven99, Mar02, Ven06]. Although Hevner *et al.* [Hev04] were inconclusive in their seminal paper, more attention has been put in more recent work [Hev10]. In the latter, a definite stance is not taken either. Instead, a clear hint towards Gregor and Jones [Gre07] is given as to what would be the preferred starting point for theorizing in the DSISR framework.

Gregor and Jones [Gre07] propose the anatomy of what constitutes a good IS design theory. The anatomy contains the following core components (Table 2 in [Gre07]):

Purpose and scope "What the system is for", the set of meta-requirements or goals that specifies the type of artifact to which the theory applies and in conjunction also defines the scope, or boundaries, of the theory.

Constructs Representations of the entities of interest in the theory.

Principle of form and function The abstract "blueprint" or architecture that describes an IS artifact, either product or method/intervention.

Artifact mutability The changes in state of the artifact anticipated in the theory, that is, what degree of artifact change is encompassed by the theory.

Testable propositions Truth statements about the design theory.

Justificatory knowledge The underlying knowledge or theory from the natural or social or design sciences that gives a basis and explanation for the design (kernel theories).

Furthermore, the anatomy also contains the following additional components:

Principles of implementation A description of processes for implementing the theory (either product or method) in specific contexts.

Expository instantiation A physical implementation of the artifact that can assist in representing the theory both as an expository device and for purposes of testing.

Although the anatomy discussed above does give the decisive answer on what theory is in DSISR, it helps in the right direction. Its completeness, clarity, and structure sets yet another milestone in the evolution of the DSISR methodology.

Although Hevner [Hev07] proposes that the epistemological underpinning of DSISR should be pragmatism, this view is not shared by everyone. Gonzalez [Gon10] discusses the need for choosing an ontological and epistemological underpinning of DSISR, following Iivari [Iiv07] and Gregor [Gre06]. Dobson rightfully asks 'Why bother with philosophy?' regarding IS research [Dob02]. The answer can be pluriform, including consciousness of one's doing and the ability to defend and argue the chosen research approach. Scientific researchers that are not conscious about their philosophical stance cannot foresee the ramifications of the philosophical stance they implicitly follow, since "even in the slightest manifestation of any intellectual activity whatever, in 'language' there is contained a specific conception of the world" [Gra71].

1.6.3 Research instruments and outline

We structured this dissertation following the DSISR cycles: relevance, rigor, and design. We have discussed current practices in the design of logistics systems and how this is tackled from a SE perspective. This lead to the definition of a research objective and the research questions that need to be answered to meet that objective. The research philosophy defines how we perceive reality and how we believe scientific knowledge can be attained.

In Chapter 2, we will initiate the relevance cycle. We will use a single-embedded case study as discussed by Yin [Yin02] to analyze the design process of a state-of-the-art automated container terminal in the Port of Rotterdam. Our research instruments will include the use of existing documentations, interviews and direct observations [Yin02]. We will study the design process that APM Terminals went through to design a container terminal that will be finished in 2014. This container terminal is expected to be a high performance terminal capable of serving the biggest vessels operational today. We will enrich the findings from this case study by analyzing literature on design studies of logistics systems.

Using the problems identified during our case study, we will study existing literature and seek possible solutions. We will discuss how the current notion of SE, and more specifically the use of simulation models, does not fully satisfy the requirements that are specified to solve the problems found in the case study. This will turn in requirements that we will pass on the design cycle.

In the design cycle, we will present the design method for logistics systems that we deem capable of satisfying the requirements presented in Chapter 4. The instantiation of this design method will be presented in 5. The instantiation is build using the case study presented in Chapter 2 to support the design of automated container terminals. This instantiation is required to evaluate it, as presented in 6. The evaluation is threefold and uses the following instruments: we evaluate the usability using a controlled experiment with graduate students which we interview using a semi-structured interview; we present the results from the controlled experiment to a domain expert to assess the quality of the design product and process; we evaluate the usefulness of the method by submitting it to experts for critical analysis. The latter also satisfies the requirement posed by DSISR to feed artefacts back to the business environment.

We will conclude this dissertation by summarizing our contribution in the epilogue. We will reflect back on our work and propose some new lines of research that flow from it.

Chapter 2

Designing automated container terminals: a case study

In Chapter 1 we presented our research objective concerning the design process of logistics systems. To initiate the relevance cycle, we have to select an instance of a business environment that is concerned with the design of logistics systems. Therefore, we opted for a case study at APM Terminals [APM11a], a subsidiary of A. P. Moller-Maersk Group [A. 11]. APM Terminals' core business is to design and operate container terminals. Recently, APM Terminals has been focusing on designing (semi-) automated container terminals: initially with a semi-automated container terminal in Portsmouth (Virginia, USA) [APM07] and later with a fully automated container terminal on Maasvlakte 2 in Rotterdam (the Netherlands) [Mea09]. Both container terminals are part of our relevance cycle and are studied through a case study.

We will continue this section by introducing container terminals and the need for automation to improve the operational perfomance. Hereafter, we will discuss our case study design following the structure proposed by Yin [Yin02]. In the next section, we will report on our case study and this will be followed by an analysis. In the analysis, we will identify the problems or research opportunities that have arisen during the case study.

2.1 Developments in the container terminal industry

From the sixties onwards [Sch70], container terminals have become a crucial link in modern logistics. Standardizing the size of 'boxes' led to a spur in efficiency in transporting and handling goods. The rapid growth of containerization led to a large number of container terminals around the world that all serve as decoupling points between sea and land transport and as storage facilities.

In spite of the global crisis in 2008 and 2009 that left a fierce wound on the global shipping industry with a decline in volume of 10%, growth has again recovered and shows a new promising increase, expected to be between 5 and 10% for 2010 [APM10, Reu10]. Although 10 to 15% was the figure associated to normal growth prior to 2008, the current growth will still result in increasingly large volumes over the coming years. The 'usual suspects' on the list of busiest ports on the planet are Singapore, Shanghai, and Rotterdam which report a volume of respectively 14.05 million TEU, 13.85 million TEU, and 5.4 million TEU for the first half of 2010 (TEU stands for twenty-foot equivalent, the size of a small container)[Bai10, tra10]. These large amounts of volume call for an appropriate infrastructure.

Current developments in Rotterdam involve some 2000 hectares of reclaimed land right next to the 'old' Maasvlakte area. This project, Maasvlakte 2, promises to dramatically increase the capacity of the port of Rotterdam from the current 10 million TEU per annum to a staggering 27 million TEU as soon as the terminals on the new areas are fully operational. The area will house state-of-the-art container terminals with the capability for handling Ultra-Large Container Ships: vessels that exceed a capacity of 10000 TEU, which is seldom possible in European ports. To achieve these high figures, many years have been put and high investments have been made into research and development for innovative handling equipment on container terminals. Automated equipment has already been employed in terminals such as ECT Delta Terminal and Euromax Terminal in Rotterdam [Eur10], and HHLA-CTA in Hamburg [HHL11]. Automation has come to play a prominent role in modern large container terminals and promises an incomparable increase in efficiency that can cope with future volumes.

Besides being more efficient, automated equipment presents other advantages. Operational advantages include the reduction of damage to the containers (automated equipment operates in more precise manners) and the increase of service level up to 24 hours a day, each day of the week. Economically, automated equipment is more attractive thanks to the reduced life cycle costs: notwithstanding the higher initial investment, the lower operational costs will guarantee savings in the long run. Automation promises also more environmentally friendly solutions: mainly electrically powered equipment is used which results in less local sound and air pollution [Pie05, Ver04b, Saa04].

Although automation sounds like the Holy Grail in container terminal operations, it can turn out to be Pandora's Box as soon as the design process of a new terminal commences. The use of advanced equipment poses real challenges to decision makers involved in the design process of container terminals. Design decisions on advanced equipment with complex control logic are challenging to make and difficult to defend in front of stakeholders. Although the operational costs are lower in comparison with manual terminals, the initial investment is higher due to the costs of research and development of automated equipment. The initial financial risk that needs to be taken is therefore bigger, which is something that container terminal operating companies are not eagerly waiting for, especially not in an economically uncertain environment.

2.2 Case study

To initiate the relevance cycle, we have decided to use case studies as our main research instrument as described by Yin [Yin02]. We have opted for a single-case embedded case study. The rationale for a single case study is twofold: the selected case study is *unique* and it is *representative*. It is unique due to the usage of innovative equipment that has been specially developed for these terminals and for the size of the terminal itself, especially the terminal on Maasvlakte 2. It is representative because APM Terminals is one of the leading companies in the world that develops and manages container terminals around the world. We have opted for an embedded case study because we have multiple units of analysis. As discussed in Chapter 1, we want to study the process as well as the system of a design study of logistics systems, thus opting for an exploratory case study. The process perspective focuses on the actors involved in the design study and the steps they undertake to complete the process. The system perspective focuses on the logistics system itself from an ontological perspective and on the techniques used to analyze the system.

To collect data, we have selected the following *sources of evidence* [Yin02]:

Documentation can be each form of written documentation that is deemed relevant to the case study and can be used as a source of evidence. We have put careful consideration in selecting appropriate documentation that does not contain unwanted biases by the authors. We have used multiple source of the documentation without restricting ourselves to APM Terminals [Der09, Ste09, Waa08, Hu08, Ver10, Ver09, Far09, Oya09].

Interviews can be carried out with members of the company to enforce data that has been acquired elsewhere or to collect new data. We have carried out several interviews with members of APM Terminals that are involved into the design process of automated terminals. These interviews are described in Appendix A

Direct observations can lead to data that cannot be captured through documentation and interviews. Especially in our case, we tried to capture research opportunities to improve design studies. These opportunities may remain unidentified by members of the company, which causes that they will not appear in interviews and existing documentation. These observations are part of the documentations mentioned above.

The case study is designed to be a descriptive one: we do not aim for generalizations or quantitative measures to study a specific phenomenon. Rather, we want to describe the design process that leads to logistics systems, more specifically an automated container terminal. We need a descriptive case study to identify the research opportunities that occur in design studies to be able to seek for appropriate solutions.

2.3 Automated equipment: increasing the handling capacity of container terminals

Container terminals play an important role in modern logistics by providing a cost and time efficient solution for connecting maritime and land transportation modes. From the mid fifties onwards, container terminals have become the main hub for transshipment (from one ship to another) and transloading (from a ship to other forms of transportation): containers arrive and depart on vessels, trucks, and trains. Due to the growing demand during the last couple of decades, container terminal operators have been steadily increasing the capacity of container terminals by employing larger vessels and transforming small terminals into more efficient larger ones [Vis03]. Advancement in modern technology has decreased the time and costs required to establish the connection between maritime and land transportation modes e.g. standardization of equipment and containers, employing more efficient equipment and control logic optimization.

A major shift has taken place in recent years substituting manual labor (e.g. crane operators) by automated controls. Modern Terminal Operating Systems (TOS) are able to control a large amount of equipment to unload containers, transport them to storage stacks and retrieve them when needed. The types of equipment that are currently best suited for automated controls are the one for horizontal transportation and the container stacks. Equipment used in horizontal transportation transports containers between the yard and the quay. Equipment in charge for the stacks, pile containers in specified areas for storage and retrieve the containers when needed.

According to Versteegt [Ver04b] and Saanen [Saa04], this shift from manual to automated labor brings numerous advantages such as lower life cycle costs, improved safety (humans are less likely to be involved in accidents), reduced damage to containers and equipment (automated equipment has a higher precision during operations), and increasing service levels (automated equipment can run almost without any breaks). Thanks to these advantages, an increasing number of newly developed container terminals are automated. Table 2.1 contains the current status of automated container terminals around the world.

2.3.1 Designing terminals within APM Terminals

APM Terminals has grown to be one of the leading companies in designing and operating container terminals throughout the world. In 2007, it has officially opened its first semi-automated terminal located in Portsmouth (Virginia, USA). The RMGs are automated on the seaside, which means that human intervention is only needed on the landside to serve trucks [APM11b]. On the new Maasvlakte 2 site in Rotterdam, APM Terminals has opted for a full-blown automated terminal [Mea09]: computer software is in charge of controlling all horizontal transport and stacking equipment requiring human operation solely on the quay cranes (QC) to load and unload the vessels.

Container Terminal	Year	Automated equipment
ECT Delta Terminal in Rotterdam (the Netherlands) [Eur11a]	1989 - 1993	RMGs and AGVs
HPH London Thamesport in the United Kingdom [Hut11]	1982 - 1990	RMGs
HHLA Container Terminal Altenwerder in Hamburg (Germany) [HHL11]	2003	RMGs and AGVs
ECT Euromax Terminal (Rotterdam) [Eur11b]	2007	RMGs and AGVs
Patrick Brisbane Terminal (Australia) [Pat11]	2007	AStCs

Table 2.1: A quasi-extensive list of automated container terminals worldwide using RMGs (Rail Mounted Gantry Cranes), AStCs (Automated Straddle Carriers), or AGVs (Automated Guided Vehicles)

As discussed earlier in this chapter, there is still a low number of automated container terminals. The two terminals by APM Terminals are recently developed and use modern technology. Our case study will focus on both the technical system and on the process of the design study of these two terminals. This case study is unique because there are few automated terminals in the world. This case study is also representative because APM Terminals is one of the leading companies in this business and uses advanced methods and technology to solve their business needs.

In Chapter 1 we defined the boundaries of our research using the system's engineering lifecycle. Within this lifecycle, we chose to focus ourselves on the system design phases. During these phases, the requirements serve as an input and a detailed design results as an output. After these phases, the (physical) implementation takes place. The design phases are usually subdivided in conceptual and detailed design. During the case study, we limited ourselves to these boundaries. This will also be reflected in the following sections.

2.3.2 The technical system

At the end of the SE lifecycle, a full fledged container terminal is rolled out ready to serve vessels, trucks, and trains. However, in our case study, we restrict ourselves to the design of the system: a blueprint of a container terminal that can be handed over for implementation, given initial requirements. The design defines which components (e.g. equipment and control software) will be used during the operation of the container terminal to fulfill specific tasks using well-defined processes.

In this section, we will provide a comprehensive overview of the different components present on automated container terminals that have been identified during our case study. We will also present the physical processes that these components have to go through in order to operate in the terminal. As we focus on automated container terminals, we will also present the system that controls the equipment on the terminal.

Layout and equipment

The most prominent part of a container terminal is the physical infrastructure: large cranes serve immense vessels while containers are being transported from and to the stacking area. During the design process of a container terminal, selecting the right equipment is fundamental for the performance of the overall terminal. Most hardware present at a container terminal is material handling equipment needed to transship containers to and from ships, barges, trains and trucks.

To identify the various types of equipment, the analysis of the different processes present at a typical container terminal, can help. A container terminal is an open system with two external interfaces [Ste05]. At the sea side we can find the quay where vessels are loaded and unloaded, whereas at the land side we can find equipment to serve trucks and trains. In the middle, we can find the stacks located in the yard, which work as an interface between the different modes of transportations. The yard provides the storage functionality that is essential for a container terminal: it works as a buffer for the transshipment operations to reduce the complexity of the overall system [Zij95]. The major processes are shown in Figure 2.1. Vis *et al.* [Vis03] schematized the different types of equipment active in the different areas of a terminal, which can be seen in Figure 2.2. QCs (also known as ship-to-shore cranes) are responsible of loading and unloading containers to and from the vessels. Smaller ships, such as barges, are served by so called barge cranes, which are smaller than normal QCs. Between the QCs and the stacks, dedicated vehicles are active to get the containers from the lanes near the QCs to the stacks. In the yard, cranes, such as RMGs, stack the containers. At the land-side of the stacking area, dedicated vehicles are again responsible to bring the containers from the yard to the rails or trucks.

During our case studies, we have identified a limited set of equipment that is used for the different areas in automated container terminals. Three tables present the equipment identified during the case study: Table 2.2 discusses the equipment on the quay, Table 2.3 discusses the equipment in the yard, and Table 2.4 discuss the horizontal transport system. Choosing between different equipment for a given task goes further than comparing specifications such as speed and costs. If we analyze the various choices that we identified for the AGVs, we see that a decision has an impact on the design of the terminal. This is due to the different nature of the subsystem and its interface with the other subsystems. Traditional AGVs (such as the ones on ECT Delta Terminal) have to wait at the stacks and on the quay until a container is unloaded from and loaded on them. On the other hand, lift-AGVs do not have to wait thanks to the AGV racks. This forms a decoupling point between the subsystems. Other horizontal transportation solutions, such as the TTS

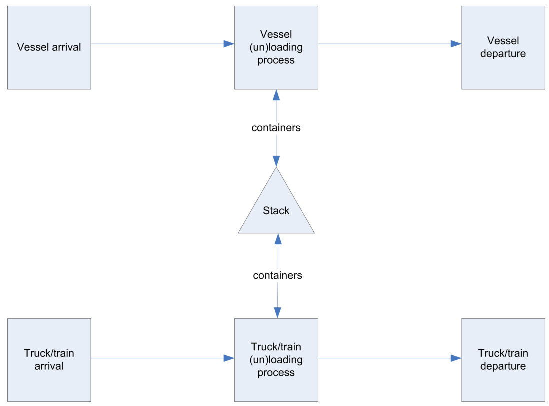


Figure 2.1: Processes at container terminals: a terminal loads and unloads vessels, trucks and trains and stores containers in the stack, which functions as a buffer or decoupling point for the complete system. Diagram based on Saanen [Saa04].

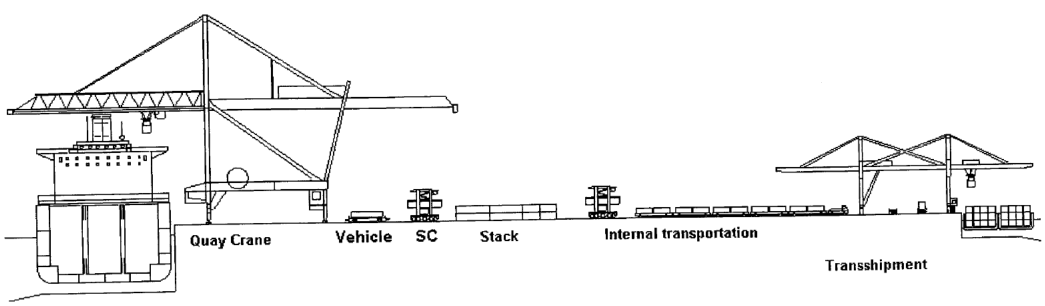


Figure 2.2: The high level processes taking place at a container terminal as schematized by Vis *et al.* [Vis03].

Automated Guided Cassette, need less space due to smaller driving curves. This influences the distances between the yard and the quay, thus potentially reducing the time needed to transport containers back and forth. These types of considerations need to be given to make an informed decision on the choice of equipment.

Type	**Name, Specification, and Description**	**Picture**
Quay crane/Ship-to-shore crane	ZPMC or Cargotec Ship-To-Shore Super Post-Panamax (capable of serving vessels that are too wide to pass through the Panama Canal, normally about 18 container rows wide). The crane load and unloads containers using a trolley that lowers and raises a spreader to lift a container. Two important decisions are assessed: 1. Single or dual hoist: single hoist containers have one spreader lifting one 40 foot container whereas the dual hoist alternative has two spreaders that can lift two 40 foot containers. 2. Lashing platform: an intermediate platform on the crane decouples the operation of the main trolley that loads and unloads the vessels and the operation of loading and unloading horizontal transport vehicles on the terminal. The second operation is carried out by a secondary trolley that is, usually, automated.	

Table 2.2: Equipment on the quay identified during this case study. (Source pictures: APM Virginia Terminal (USA)

An important part of the physical infrastructure is its layout. The design of a terminal should take into consideration driving distances and congestion to achieve a high throughput. One of the major decisions during the design of a terminal is the orientation of the stacks towards the quay: it can be perpendicular, parallel or a combination thereof (the latter could be in case the complete terminal is not a rectangle but a more complex shape like an L-shape). The two layout options are schematized in Figure 2.3. As mentioned before, the layout and the choice of equipment are closely related: depending on the choice

Type	Name, Specification, and Description	Picture
Rail mounted gantry crane	KCI Konecranes RMG (lifting capacity of 40 tons, a lifting height of 1 over 5 containers, and a span of 25.4 m)	
	Gottwald Automated Stacking Crane (ASC), which uses a rigid guiding beam to increase the precision of the operations, thus speeding up operations.	
	Cargotec Automated Stacking Crane	

Table 2.3: The stacking system identified during this case study. (Source pictures: 1) KCI Konecrane RMG at APM Virginia Terminal (USA), 2) Gottwald ASC from http://www.-gottwald.com, 3) Cargotec ASC from http://www.cargotec.com)

on horizontal transportation, a different layout may be required.

Managing automated equipment

Although the physical equipment is the most prominent part of a container terminal, the functioning of the system depends on software that manages that equipment. A TOS is extremely important in automated container terminals: it manages the orders (containers that have to be transported) and interfaces with the equipment subsystems for rail, yard, and quay handling. The complexity of a TOS is increased by the need of optimizing operations: it has to take into account the arrival of large vessels by preparing the yard to load the vessel and reserve enough capacity to carry the operation out as quickly as possible.

APM used Navis SPARCS TOS for its semi-automated terminal in Portsmouth, Virginia. An important component of the TOS for this terminal is its ASC Manager module. This module provides the automated stacking crane functionality in the yard. As this ter-

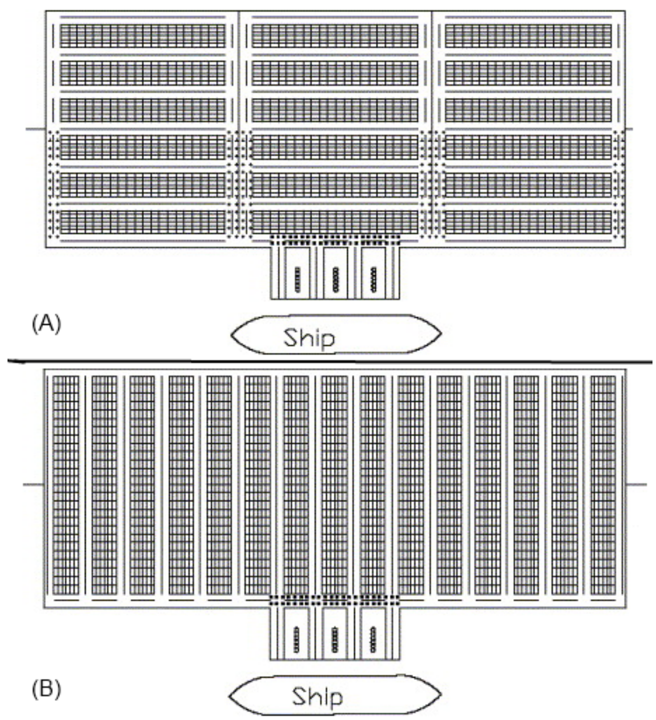


Figure 2.3: The yard of automated container terminals are positioned either parallel (A) or perpendicular (B) to the berth as schematized by Liu *et al.* [Liu04]

Type	Name, Specification, and Description	Picture
Automated guided vehicles	Gottwald Lift-AGV: The name reflects the characteristic of the AGV to lift a container and put it on a rack near the stacks in the yard. At the sea side, the QCs will put the container on the AGV that needs to wait on appropriate spots.	
	TTS Cassette AGV: cassettes are used as a decoupling point between the horizontal transport system and the systems it interacts with. Cassettes are positioned beneath the QCs that load or unload the containers in the cassettes. The loaded or empty cassettes are transported to the required places by the AGVs. Cassettes are placed near the stacks for RMGs to load or unload the containers in the cassettes.	
	Cargotec Kalmar Shuttle Carrier is able to pick up containers that are placed on the ground. The QCs or RMGs put the containers on appropriate spots and lift containers from these spots. The shuttle carrier picks the containers to transport them from the quay to the yard and back.	

Table 2.4: The horizontal transport system identified during this case study. (Source pictures: 1) http://www.gottwald.com, 2) TTS Cassette AGV from http://www.ttsgroup.-com, 3) Cargotec Kalmar SC from http://www.cargotec.com)

minal is highly automated, the TOS has to interface with many software subsystems such as

- gate software to manage the gates from which trucks enter and leave the terminal using OCR technology to carry out security checks and to register the order;
- GPS software to keep track of the equipment on the terminal;
- RFID software to register the containers on the terminal;

- Reefer Monitoring Software to keep track of reefer (refrigerated) containers;
- and security software to check for hazardous and illegal materials.

The complexity of the TOS increases on terminals such as the fully automated Maasvlakte 2 terminal. The TOS that will be implemented on this terminal needs to interface with the subsystem of the automated vehicles. Intuitively this would mean that the TOS sends orders to the vehicle subsystem on a need-to-know basis. However, this would reduce the performance of the overall terminal. To optimize the terminal, a more complex interface is needed that allows the TOS and the vehicle subsystem to prepare for upcoming orders and position the equipment as efficiently as possible. The management strategy differs depending on the vehicle types: automated straddle carriers can pick up containers that have been put down earlier by the QCs, whereas a cassette needs to be put in place if the cassette AGVs operate on the terminal. For this reason, a TOS needs to be highly customized to each terminal and requires extensive fine-tuning and testing before a terminal can go operational.

2.3.3 The design process

The goal of this chapter is to dissect the design process for automated container terminals. In the previous sections, we have situated container terminals within global supply chains, highlighted the performance gain that needs to be made in order to stay competitive and presented the equipment on automated container terminals that could potentially improve the efficiency on those terminals. In this part of the chapter, we will elaborate on the design process itself, which we consider to hold the research opportunities we are looking for. Before we discuss the process itself, we need to pay attention to the actors involved in the process. Once we have done this, we can discuss the process itself.

Actor analysis

We pinpointed the fact that the design process of automated container terminals is carried out by multiple actors. However, we did not yet go into detail on the specific actors. An actor analysis [Lei09] can help us in identifying the different actors and defining their role in the process. This will help us understand better the interactions that take place during the design process. Hu [Hu08], Steenstra [Ste09], and Derksen [Der09] presented an actor analysis of APM Terminals. The actor analysis is carried out following Enserink *et al.* [Ens10] method and using Hermans's guidelines [Her05]. To focus this analysis on the design process of automated container terminals, we have interviewed 5 employees within APM Terminals that have a leading position in designing automated container terminals within the company (see Appendix A for the interview protocols and the list of interviewees). Based on the actor analyses and the interviews, we have identified the key-actors involved in design processes of automated container terminals. We define the 'key-actors' as the actors that have a high commitment to the design process and are the direct design decision makers. The actors can be found in the following business departments:

Design & Innovation is mainly involved into the operational aspects of new container terminals. Main tasks are the provision of terminal layouts and the development and analysis of new technology.

Business development and implementation is in charge of developing the business case, costs analysis, and project management.

Health, Safety, Security and Environment ensures the minimization of hazards and pollution of new container terminals.

Legal & Tax is in charge of the legal aspects of a new business, more specifically the contracts with suppliers and customers.

However, a number of external actors also greatly influence design decisions:

Consultants are part of the design process to provide specialized knowledge in areas such as project management, operations, and technical infrastructure.

Suppliers work closely during the design process to ensure that their products can be implemented correctly.

Customers provide the estimate on required throughput of the new terminal.

During the design process, only a limited set of actors influence the design of the system directly. The decisions certain actors make, has a direct impact on the design:

Business developers define the financial constraints and business opportunities that are applicable to the new design. Their perception on the system is twofold: their decisions influence the investments that can be made and they also provide the clients that will use the terminal in the future. On the one hand, this means that the terminal needs to remain within a certain budget, restricting investments on equipment and infrastructure. On the other hand, they base specific design decisions on the possible clients: for instance choosing to be able to serve large vessels if the appropriate client is present. The KPIs they use are in terms of capital expenditure (CAPEX), operational expenditure (OPEX), revenue, and cash flow [Hu08].

Operational developers (within Design & Innovation in the previous list) are responsible of optimizing performance given the financial restraints by the business developers. The KPIs used by operational developers are extensively discussed by Saanen [Saa04]: e.g. container terminal throughput, annual container handling capacity, and transshipment factor.

Environmental analysts provide a secondary restraint, besides budget, on the operational developers in order to reduce environmental impact such as the carbon footprint. This actor interacts with above as the environmental impact depends on operations (more operational hours mean high emissions) and budget (environmentally friendly equipment tends on being more expensive) and expressed in CO_2 emissions [Gee11].

Different design phases

Container terminals are logistics systems that are designed, implemented, and operated using the SE lifecycle. Rijsenbrij [Rij99] identified the different steps that are taken up to the start of the technical design, thus mainly focusing on the conceptual design. The process is shown in Figure 2.4. Comparing this process to typical SE process such as the waterfall, spiral, and V model, shows us that Rijsenbrij's process focuses on the requirement analysis and design phases.

From the interview described in Appendix A, we have gathered the phases employed at APM Terminals to develop new container terminals. During each phase, multiple actors are involved that we described in the previous section. We also want to know which support tools are used for each phase. We will mention the support tools and go into more detail about these tools in the following section.

The phases of a design process at APM Terminals are as follows:

The project acquisition phase is the phase in which different organizations present competing conceptual designs to win a bid on the organization that is going to develop a new container terminal. This phase is mainly in hands of the business development actor. During this project acquisition, visualization software is used to present a preliminary design of terminal with the main goal of winning the bid. This is also in accordance to Budde *et al.* [Bud92] notion of presentation prototypes.

The global engineering phase is the phase in which static calculations (e.g. rules of thumb on the number of cranes necessary to get certain productivity rates) are used to generate a draft design of a container terminal. The static calculations are mainly carried out using spreadsheet models. This phase involves practically involves all actors described in the previous section.

The detailed engineering phase entails refining the draft design through dynamic models using computer simulation. The scope of this phase is to have a detailed understanding of the system from an operational standpoint. As with the previous phase, most actors are also involved during this phase.

The implementation phase transforms the detailed design into a physical reality: the container terminal is actually being developed. Before the actual construction takes place, various plans need to be developed by architects and various types of engineers

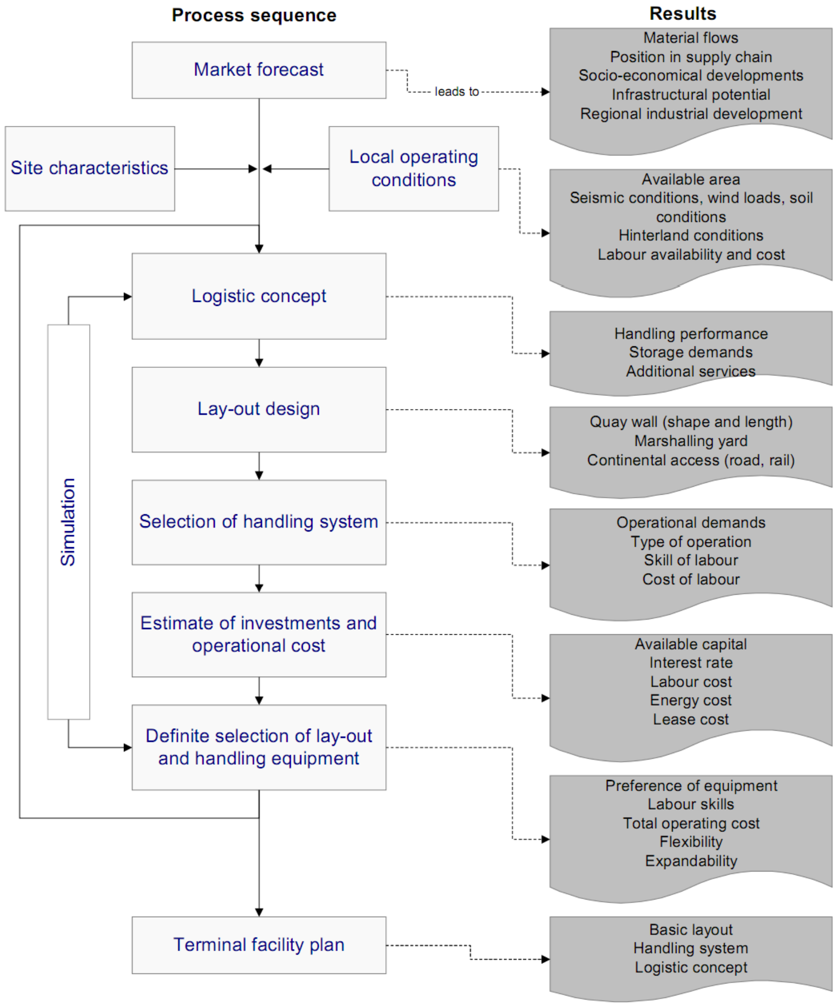


Figure 2.4: Container terminals are typically designed following specific steps up to the technical design [Saa04, Rij99].

(e.g. civil, mechanic, electrical). During this phase, close cooperation is required with external actors such as suppliers and consultants (e.g. IT consultants for the control infrastructure).

The operational phase is strictly speaking not part of the design process, but can still contain changes to the design of the terminal. This phase contains the lifetime of the container terminal, after it has been developed and is in full operation. Simulation models can interact with real-time data feeds to improve the operations of the actual terminal. This option has also been discussed by Shu *et al.* [Shu07].

Using support tools

In the previous section we named two important support tools that are used during the design process of automated container terminals. Static spreadsheet models are used to make the preliminary calculations whereas simulation models are used to have a detailed insight into the operations of the terminal.

Hu [Hu08] and Derksen [Der09] describe an analytical model they have developed to support decision making at APM Terminals. This model, developed in Microsoft Excel, requires input parameters such as volume, modal split, dwell time, purchase prices, equipment specifications, and fuel costs and outputs parameters such as capital and operational expenditures, number of equipment needed, operational hours of the equipment, and energy usage. The structure of the model is shown in Figure 2.5.

The analytical model is mainly used during the global engineering phase that results in a draft design. The model is used by actors that are involved into the design, costing, and innovation of the new terminals. The model is used to develop the business case that is presented to the board of directors of APM Terminals to get the approval to continue with the development of the container terminal. Once this is achieved, the output of the analytical model can be used as input for the following phase in the design process.

During the detailed engineering phase, simulation models are used to support the design process. Besides the internal actors, external actors such as simulation consultants and suppliers are heavily involved during this phase to supply simulation models of the overall terminal and specific models of the equipment of various suppliers. During the design process of APMT's Maasvlakte 2 terminal, a project was commissioned to an external consultant to develop the overall simulation model of the container terminal. The simulation model consists of two sub-models: one to model the distribution pattern of vessel and land (trucks and trains) arrivals, the other to model the actual container handling using QCs, horizontal transport and stacks. The output of the first model is used as an experimental frame for the second model [Waa08]

This model was used to assess different design decisions, such as:

- Layout: the orientation of the stacks towards the quay can be either parallel or perpendicular. For each layouts, a model was developed to analyze the results.

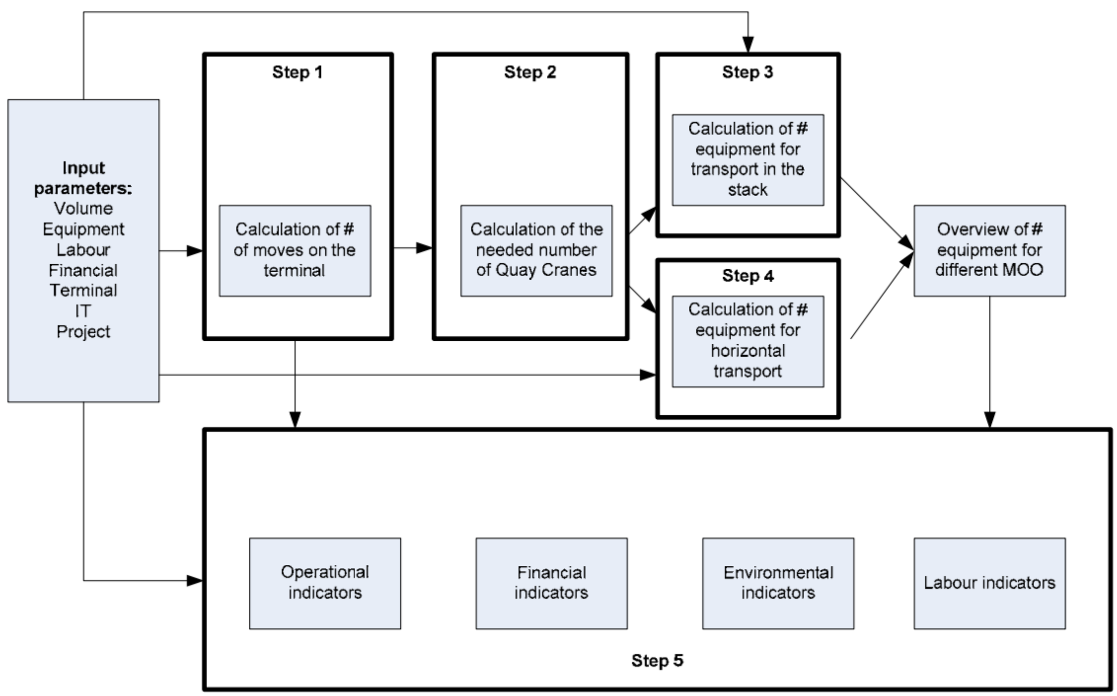


Figure 2.5: The structure of the analytical model developed by Hu [Hu08] and Derksen [Der09] (the latter is the author of this figure). MOO stands for Mode of Operation.

- Quay: alternative solutions for the quay were assessed using various types of QCs, piers, jetties, etc.
- Horizontal transport: the suppliers of AGVs developed simulation models of their equipment to be run in a distributed setting with the terminal model.

Simulation models are used to study the dynamic aspect of the terminal such as QC cycle times, congestion, and stacking algorithms. The output of a simulation model is presented in terms of KPIs for utilization time of equipment, terminal throughput, and waiting times. Using the simulation model, final decisions can be made on equipment, control algorithms and physical properties of the terminal. These decisions are then used during the implementation phase of the terminal.

The complexity of the design process of automated container terminals is defined by the number of design decisions that needs to be taken by several actors. Furthermore, the throughput time of such a project is long: from a few months to several years. This accentuates the need of having a documenting system to capture which decisions have been taken, why these decisions have been taken and by whom. Failing to adopt such a system, results in a process that is hard to follow and reconstruct in a later stage. The reconstruction is fundamental for auditing the process, but also for following the progress of the process and to infer knowledge for future reference for similar projects [Ver09].

2.4 The role of simulation models: identifying the research opportunity

In the previous section, we have discussed the presence of various disjunct models used throughout the whole process. The design process of a logistics system such as container terminals is characterized by the use of models, either static or dynamic, to analyze the future design and hopefully reduce risks.

2.4.1 Designing automated container terminals using simulation models

The use of models, either static or dynamic, to support the design process of container terminals is a common approach. It is not new: the first published results date back to 1972 [Dun72]. The rationale for using models has not changed much throughout the years. Designers want to be able to estimate design specifications such as the optimal storage capacity, equipment number, and the size of the container terminal [Bor75]. In the last four decades, simulation models have been used to provide more detailed analyses and new methods have been developed to construct the simulation models. Steenken *et al.* [Ste05] report on a number of advances such as:

- Simulation models to evaluate dispatching algorithms for AGVs (a) using an agent based AGV controller that provides an effective flow and no deadlocks, even in complex designs [Wal01]. (b) by minimizing (un)loading times for vessels and avoid deadlocks to improve the throughput of a terminal in Singapore [Leo01], and (c) by using an auction algorithm using a distributed decision process with communication among related vehicles [Lim03].

- Kim *et al.* [Kim02] discuss a simulation study on operational rules for Double-RMGs: crane dispatching rules are tested with and without different roles for the different cranes.
- The analysis of the optimal number of berths and QCs for a terminal in Busan (Korea) using different operational patterns in four scenarios for performance evaluation [Nam01].
- An object-oriented simulation model for a container terminal analysis including gates, container yard, berth and equipment [Yun99].

The simulation methodology itself has not changed much as well: the rigid simulation project life-cycle, as described by Nance [Nan83], is still being applied today in similar ways [Sar06]. In our case study, we can clearly identify the simulation project life-cycle proposed by Nance [Nan83] with the extra step of starting the simulation study with the results of the static model run a priori. Using the static, spreadsheet based model allows the designers to experiment with different parameters before commissioning the simulation study to external consultants. This is not done with the dynamic simulation model, because it is less quick (experiments can take various days), less flexible (each change needs to be implemented in code), and more costly.

The inability to use simulation models instead of the static models has a number of disadvantages as it cannot capture the dynamicity of the system. As we have discussed before, different layouts, equipment types and numbers need to be assessed during the design process. However the dynamic nature of the system has a high impact on the final design. Design decisions such as layout and number of equipment can cause less or more congestion, different algorithms to run automated vehicle can change the throughput of the terminal, and the size of the yard influences the time needed to retrieve a certain container. Considering the fact that static models are employed to have a possibility to assess a design early in the design process, the question poses itself whether the use of simulation models can be repositioned earlier in the design process to take the dynamic nature of the system into consideration.

2.4.2 Repositioning the use of simulation models in the design process

Saanen [Saa04] called for using simulation throughout the whole design process, yet current practice suggests little has changed. According to Saanen, the extended use of simulation is a new paradigm that should be “favored over other approaches, because the interaction between the components of a container terminal cannot be captured by analytical models [Pie01]”. Saanen continues by citing Hilkins [Hil02] to argue that mathematical models that use queuing theory require major simplifications of the system that makes it impossible to analyze certain interactions, visualize the operational process, and validate the results.

Current practices in container terminal design still use simulation during the evaluation phase of the SE lifecycle. During the early phases of a design project, designers prefer to adopt assessment tools that are easy to use and give fast results. Simulation models, on the other hand, need specialized knowledge and require a longer time to run in order to produce results. Simulation is treated as a scarce resource that has to be tapped when absolutely needed. To reposition the use simulation models in the design process, a design method is needed that takes into consideration the actors involved in the process and the technology that supports the process.

This case study showed the complexity present in the design process of automated container terminals. The many decisions that need to be made incur multiple interactions in the technical system and the actors involved in the process. Simulation models can capture the dynamicity of the technical system and by doing so, support the decision making process of the designers. By repositioning the use of simulation, by having it during the early stages of the design process, it would be possible to support the actors in rigorously assess multiple design alternatives during the design process.

2.5 Conclusion of the case study

In this chapter, we have discussed the case study we have carried out at APM Terminals. This case study was part of the relevance cycle of the design science research presented in this dissertation. As part of the relevance cycle, we aimed at identifying problems or research opportunities that can be fed into the design cycle.

The research opportunity we identified during this case study was focused on the position of simulation models in the design process of automated container terminals. We have seen that the complexity of these design processes is present both from a system as from an actor perspective. In the following chapter, we will present the rigor cycle in which we are going to analyze theories, methods, and models that relate closely to our research opportunity.

Chapter 3

Designing systems using simulation models

In the previous chapter, we have discussed the current practice of designing automated container terminals. Various methods and models were discussed that proved to be fundamental in carrying out successful design processes. But where do these methods and models come from? Automated container terminals are logistics systems whose operational aspects are studied from fields such as OR and SE. These fields established their own distinctive identify throughout the second half of the 20th century. In this chapter, we will explore both fields as part of our rigor cycle. We will discuss the fields in light of the research opportunity that we have identified in the previous chapter. Therefore we will focus on the role of simulation models during the design process of logistics systems. Based on this, we will work towards the identification of issues: limitations that we perceive in current methods or models that support the design process of (logistics) systems. Throughout the chapter, we will highlight these issues and these will be fed into the design cycle discussed in Chapter 4.

3.1 Designing systems

In the early sixties of the previous century, Alexander [Ale64] presented his seminal work that introduced the notion of methods in design studies. Up to that point, a design effort was considered a craft that relied solely on intuition and skills. Alexander postulated that design methods could aid the designer in fitting the form (the design artifact) to its environment: "every design problem begins with an effort to achieve fitness between two entities: the form in question and its context". To fit the form to its environment, formal methods and techniques should be required for both design products and design processes. Although the method he presented turned out to be unsuccessful for numerous reasons, the prior misconception of rejecting formal procedures in design was abandoned. This led to an abundance of novel design methods from 1962 onwards [Cro93].

During the same period as the design method movement, a modern understanding of systems thinking gained traction in science and engineering. Holistic thinking was put next to reductionism, which dominated science for a long period of time. Von Bertalanffy's general systems theory [Ber50] was readily translated to new fields like OR and SE [Ack73]. Hall [Hal62] clarified the distinction between the two by pointing out that OR is concerned with optimizing existing systems, and SE's concern is focused on the design of new systems. Jenkins [Jen69] sharpened the difference by suggesting that system engineering looks at the total system whereas OR tinkers at the level of the more mechanical sub-systems. It is important to note that the recognition of SE as focused on design binds system's thinking to Alexander's idea on formal procedures for design studies. But what exactly is a system? Based on Ackoff [Ack73], Klir [Kli69] and Churchmann [Chu71], we can define a system as follows

Definition. *A system is a set of components and relationships to serve a certain purpose.*

To design a system we have to go through a system design process, which is defined as follows:

Definition. *A system design process is a set of activities to produce a system design.*

At the crossroad of system's thinking and design lies Simon's [Sim96] contribution. He observed that a design process must follow a specific path of first structuring the problem, followed by a formulation of alternative solutions based upon selected criteria and finally by a selection of the best alternative. Human's bounded rationality limits the designer's capabilities in exploring the solution space, and thus precludes from finding an optimal solution. This limit is due to various factors such as incomplete information, cognitive limitation of the designer and time pressure. The designer cannot and should not aim for optimizing while designing, but should compromise between satisfying and optimizing, which Simon denotes as *satisficing*. Based on these premises, future system engineers would go about accepting the paradigm as follows: : "There is a desired state, S(1), and a present state S(0), and alternative ways of getting from S(0) to S(1). 'Problem solving,' according to this view, consists of defining S(1) and S(0) and selecting the best means of reducing the difference between them" [Che78]. More specifically, the idea ruled supreme that the problem task to tackle would be about selecting the efficient means.

Design studies are carried out in different fields, most notably architecture and engineering. Design is considered the essence of engineering [Whi99]. Hubka [Hub88, Pah95] suggested to design technical artifacts as systems that are connected to their environment by means of input and outputs. The system can be divided into subsystems taking into consideration their boundaries. Hubka argued this would be fundamental to define appropriate systems at any stage of abstraction, analysis or classification. Pahl and Beitz [Pah95] pinpoint that from this notion onwards, it was a short steps towards using system's theory in design processes and specify that "systems approach reflects the general appreciation that complex problems are best tackled in fixed steps, each involving analysis and synthesis". Similarly, van Gigch [Gig91] discusses the systems approach as a methodology of design. In his work, he presents modeling as the fundamental aspect of the system design process. Modeling implies that "the modeler abstracts properties from things in order to obtain a representation of the physical world". The abstraction process plays an important role in design as designers go through it to refine images of reality through different levels of conceptualization. The importance of models and emphasis on abstraction is further given by Hoover and Renderle [Hoo91]. They discuss that abstract models can be defined at different stages in the design process to test design decisions and to provide a framework for making design refinements.

In engineering design, the aforementioned notions on design space exploration, abstraction and modeling have found considerable impact in methods and support systems. These methods and support systems sprang from the following issue:

Issue 1. *Without a well defined structure, the system design process may not be able to support the abstract, divergent and convergent thinking of system designers.*

Engineering design denotes a "systematic, intelligent process in which designers generate, evaluate, and specify concepts for devices, systems, or processes whose form and function achieve clients' objectives or users' needs while satisfying a specified set of constraints" [Dym05]. Choosing among design alternatives is a common underlying concept for many methods. Hazelrigg [Haz98, Haz99] goes as far as framing a truly rational process to produce the best possible result using 'a mathematics for design'. Many others have proposed constructs to structure and choose among alternatives: such constructs include trees [BA91], matrices [Die83], rankings, and charts [Ott95]. Methods focusing on modeling on different level of abstraction also exist, most prominently in the work presented by Paredis *et al.*. [Par01]. This variety of methods highlight the challenge designer incur when multiple alternatives need to be managed. In light of this, we can formulate the following issue:

Issue 2. *Without the ability to manage different system designs throughout the system design process, designers may not be able to choose among alternatives.*

System's thinking as originally understood in OR and SE did not fulfill its promises. This induced Ackoff [Ack79] to call for a new paradigm that would break away from the ever-increasing "mathematization" of the field. OR and SE did not pay attention on the actors involved in the decision making process. Particularly with regards to management problems, this was recognized as an important shortcoming. Mingers [Min10] discusses that traditional SE designs systems starting from the purpose of the envisioned systems and working backwards with mathematical techniques to find ways to achieve their objectives. This is based on the flawed assumption that the objectives are clearly stated at the beginning of the design process. The mere fact of including the notion of actors induced a paradigm shift. This brings us to the introduction of the next definition:

Definition. *An actor in a design process is a party who has its own interests and concerns in the design process, and who controls a part of the resources needed for successful implementation [Her05].*

It is important to note that the system designer is also an actor.

An important step forwards was taken with the introduction of the soft systems methodology (SSM) by Checkland [Che81b]. Acknowledging changing requirements, different actors with different stakes and opinions, and the importance of a learning process, proved to be fundamental to develop the SSM. The methodology stood the test of time and, after some revisions, is still successfully used today [Che06, Che10]. The so called hard systems methods in OR and SE had to face competition with a number of new soft systems methods. A taxonomy for system's approaches, called the system of systems methodology, presented by Jackson [Jac84], was developed to help select a suitable system's approach for a given problem task. Mingers [Min03] argued that the whole set of methods could be classified according to ontology, epistemology and axiology, that is what they model, how they model and why the model. According to this classification, the hard systems method lays the importance on the artifact. Conversely, soft systems methods tend to focus on the users, whether they can understand and discuss the problem situation. Because of the different foci of both types of methods, the question arises whether multimethodology can exist: why choose one or the other when we can exploit both to best tackle a problem (situation)?

3.2 Spanning the complete system's life cycle: from inception to disposal

SE has evolved from a rather mathematical theoretical underpinning of system design towards an approach of transforming user's needs into an operational system via an interdisciplinary process. Thus, the current focus of SE is on the design and management of complete systems from their conception to their full disposal. After its inception at Bell Telephone Laboratories [Hal62], SE methods and techniques have been developed and practiced by NASA [Nat07], the Department of Defense (DoD) [Dep10, Pie05], and IEEE, each producing their own best practices and standards. Nowadays, the institution in charge of advancing and disseminating SE theory and practice is the International Council on Systems Engineering (INCOSE) [INC10].

In SE, a system is designed and managed using specific models that take into consideration the various life cycles of the system. Most SE models are generic and can be used within various projects, although specific tailored made models exist to fit special requirements, such as the NASA SE Engine [Nat07]. The new breed of SE models aimed at coping with large-scale systems that are complex and require multidisciplinary teams to cooperate. Estefan [Est08] argues that there are three seminal lifecycle development models from which each other model is derived: Royce's Waterfall Model [Roy70], Boehm's Spiral Model [Boe88], and Forsberg and Moog's "Vee" Model [For92]. Although the Waterfall

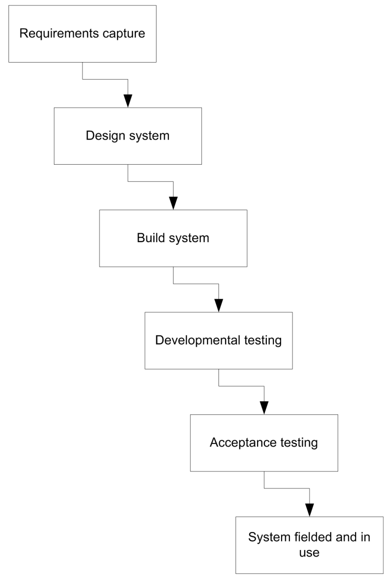


Figure 3.1: The waterfall model (figure based on [Roy70]

Model, shown in Figure 3.1, and the Spiral Model, shown in Figure 3.2 have found plenty of applications in software engineering, the "Vee" Model, shown in Figure 3.3 remains predominant in systems engineering and systems development [Est08].

Modern SE tends to go from a document-driven to a model-driven approach. Whereas the waterfall with its system definition, system design, and design qualification follows a strict process of passing documents from one stage to another, modern model-centric approaches concentrate on activities that support an engineering process that is to be accomplished through development of increasing detailed models [Est08]. Although this allows for a more iterative and explorative design process, documenting the process becomes more challenging. As discussed in Section 2.3.3, this results in a process that is hard to follow and audit. This leads us to the following issue:

Issue 3. *Without the ability to document assumptions, decisions and argumentations, actors may not be able to follow and audit the system design process.*

Looking back at the lifecycle model observed during our case study, we can recognize the "Vee" model in APM Terminal's design projects. After subdividing the design phases, the integration phase takes place with suppliers and external consultants. During our case study, we focused on the "systems design" phase, whereas we neglected the "systems integration" phase. This is due to our research objective that we defined in Chapter 1. Following this objective, we are interested in positioning the use of simulation models within the SE lifecycle.

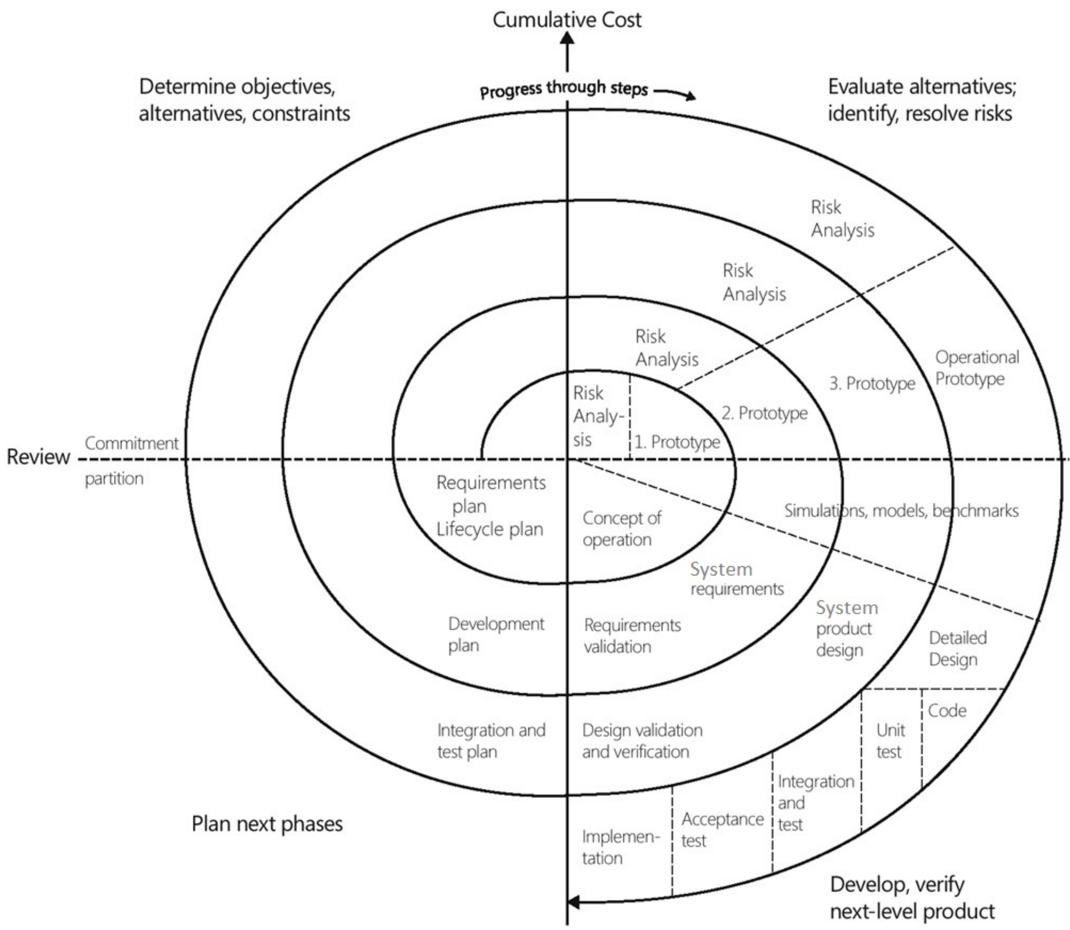


Figure 3.2: The spiral model (figure based on [Boe88]

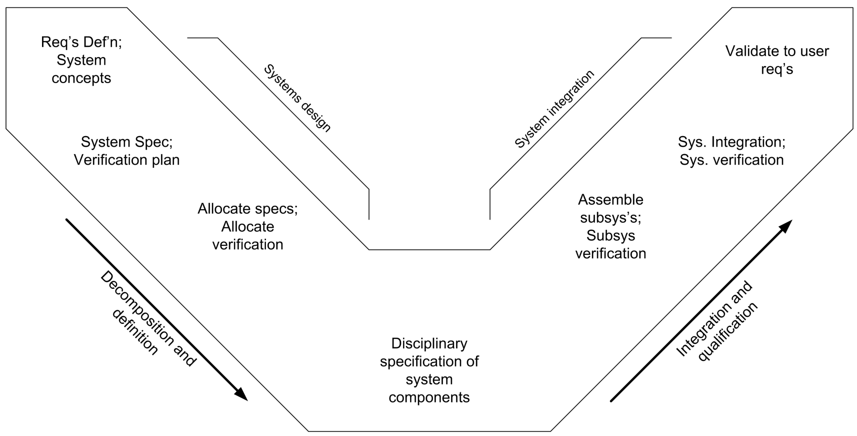


Figure 3.3: The Vee model (figure based on [Est08]

3.3 Systems Simulation in Design

As we saw in the previous sections, design is a goal seeking activity which tries to reduce a gap between a current state and a desired state with the help of an engineered artifact. In this activity, models have been used to evaluate whether an alternative solution takes us close enough to the desired state. The complexity of the design problem generally forbids the use of optimization techniques alone. Thus, the goal seeking behavior is assumed by the system designer, and not by the model itself, which is neutral and structurally static. The computational model, used as a replacement of reality, is altered until its output variables give satisfactory results.

In SE lifecycles, the simulation models are used for analysis. As noted by Banks, "simulation is used to describe and analyze the behavior of a system, ask 'what if' questions about the real system, and aid in the design of real system" [Ban00]. This supposes the ability to compare systems that vary in a limited set of parameters to be able to identify a system that is preferred by the system designers. This leads us to the following issue:

Issue 4. *Without the ability to compare different system designs throughout the system design process, actors may not be able to identify their preferred system design.*

Simulation models are only used to evaluate the design choices that have been reached using other tools, such as CAD. This is because designers are not generally comfortable in modeling using simulation languages which still require specialized knowledge. Furthermore, model building is generally a costly and time consuming activity which is externalized to analysts or consultants. Integrating design and evaluation activities could improve the quality of the designed artifacts and make the design process smoother.

To achieve this goal of facilitating modeling for a better integration in the design cycle, component based approaches look promising. Modular system frameworks are key enablers. DEVS [Zei00] and similar formalisms are particularly well adapted to help achieving that goal. Well tested simulation model components with well defined interfaces can be made available to the designer. The design activity then means to couple the different components in order to achieve a certain function. With the predefined components, the user does not have to worry about the underlying complexities. The focus can be completely put on the design itself.

Computational models are mathematical objects, made of variables and operators. They are obtained after a process of abstraction which gets rid of most of the complexity in the real system. The modeler - to make the task feasible - chooses a certain theoretical or pragmatic perspective, draws a boundary as tight as possible around the system of interest, selects a set of relevant variables, and chooses to exclude other variables. After this process of abstraction, the model reflects the system of interest, but it also strongly reflects the intentions and preconceived ideas of the modeler. If this model has to be used in a multi-actor design activity, all stakeholders are implicitly asked to adhere to the paradigm of the designer, otherwise, all questions and ideas they may have based on the model can be out of scope or meaningless.

Using simulation in design also raises the question of model validity. The system being designed does not exist in real life, or at least not in its desired state. Replicative validation techniques, which are based on a statistical comparison between the model and the real system, are not applicable. To be of any use, the model used in a design activity should at least have predictive validity. To support fruitful discussion on the real envisioned system, the model should be structurally valid as well. Of course, the fact that the components are valid does not guarantee that the aggregated model is equally valid.

A final note on simulation based design concerns the static structure of most simulation languages. A framework for simulation based designed should allow models to see their structure altered dynamically during runtime. Formalisms with such capabilities have been introduced recently [Bar03, Hu05b].

3.4 From hard to soft: multimethodological approaches

In systems literature, a distinction is made between hard and soft systems approaches. Up until this point, we mainly discussed hard system approaches, whereas we only briefly mentioned the soft systems approaches. Robinson [Rob07] talks about a continuum between soft and hard instead of bipolar extremes. This is motivated by novel interactive simulation environments developed in the nineties that can deal with uncertainty (changing requirement and lack of knowledge about the system) by allowing incomplete or estimated data. These environments can lead to discussions, which can be framed as a soft systems approach. Soft system methods emerged due to unsatisfactory applications of hard system methods to wicked or ill defined problems. This led to an abandonment of formal methods developed with the hard systems mentality and favored the loose frameworks known as soft systems methods. However, this choice is unfortunate, as both types of methods have their disadvantages that can be overcome by seeking an appropriate combination.

A tendency towards combining hard and soft systems methods is reflected in the results of surveys conducted among practitioners and in the amount of papers reported using a multimethodological approach that are published in prominent journals of the field. A major survey reported by Munro and Mingers [Mun02], show that in situations where a combination of hard and soft systems is considered useful there is a slight preference towards discrete event simulation. Unfortunately, no clear reasons are given for this preference. To understand what justifies a successful implementation of discrete event simulation with a soft systems method, we have to rely on other case studies.

Mingers and Rosenhead [Min04b] report case studies where a multimethodology was used in problematic situations. Among these case studies, several instances of simulation studies were conducted using a soft systems approach, more specifically studies by Lehaney and Paul [Leh96], Lehaney and Hlupic [Leh95], and Bennett and Worthington [Ben98]. In these cases, SSM is used to support the model building phase of the simulation studies. This participative action brings the model closer to the problem owner and allows discussion on the model to explore the problem. A brief list of recent contributions that explore this possibility further show that this is a viable approach:

- Kotiadis [Kot07] discusses how SSM can be used to determine the conceptual model's most important component, the simulation study objectives. The case study was conducted to improve a health care system.

- Lehaney *et al.* [Leh08] present a case study that was initially meant as a conventional resource allocation simulation study, but that ended up as a simulation based discussion to reveal issues of misunderstanding within the studied organization and poor communication that led to misallocation of resources.

- den Hengst *et al.* [Hen07] Explore how soft OR principles can be used for collaborative simulation. They conducted a case study in the Dutch airline industry and concluded that the combination is promising, but more research is needed to tackle a number of issues encountered in their research.

- Pidd [Pid07] advocates the use of 'soft' approaches to make sure analysts tackle the right problem in a situation study. He provides a general guideline on how such a study can be carried out.

- Baldwin [Bal04] states that simulation studies cannot be conducted in health care or other fast-changing businesses. According to the authors, this is due to the vague problem formulation phase present in simulation studies. They argue this phase is crucial and present their approach that puts more attention on defining the problem.

- Robinson [Rob01] discusses how discrete event simulation can facilitate a discussion among stakeholders to identify problems in a user support helpline.

Although this is most probably not a complete list of studies showing this type of approach, we can treat it as a list of prominent examples from which we can draw conclusions. These studies show that in designing socio-technical systems, multimethodology is used for in framing the problem, learning about the problem and finding viable solutions. In each study, strong points are identified in both types of approaches, and exploited in the overarching approach. In the next sections, we will analyze both the hard side and the soft side of these studies, to identify their weak and strong points. But first, we will take a closer look at the most prominent soft system approach: Checkland's SSM.

3.4.1 Soft systems methodology

Checkland's SSM plays a prominent role in multimethodological approaches. The approaches presented in each study at the beginning of this section differ in how much they adhere to the methodology: some prefer to call their approach 'softer' whereas others explicitly state using SSM. We will briefly present SSM.

SSM differs from hard systems approaches in the way it perceives systems. Whereas hard systems approaches assumes a system that is perceived equally by all actors, soft systems approaches discuss systems as a human's view on reality, hence a human construct used for understanding. In contrast with hard systems thinking, SSM does not focus on the solution, rather on a learning process actors go through while dealing with the problem situation. Hirschheim *et al.* [Hir95] describe SSM as "a framework which does not force or lead the systems analysts to a particular 'solution', rather to an understanding". Throughout the many years of development, SSM has matured and has gone through many versions: the most widely used version is known for its seven stages.

The seven stage soft systems methodology

The seven stages of classic SSM are about (1) defining the problem, (2) expressing the problem, (3) finding the root definitions of relevant systems, (4) developing conceptual models, (5) compare the conceptual model with reality, (6) define interventions, and (7) undertake action to improve the situation.

During the first stage, the problem situation is assessed. By collecting all sorts of data, one tries to gather a broad set of information available on the problem situation. No restrictions are given as to how much information is needed: this phase's sole purpose is to explore. In the next phase, expressing the problem, the information is structured to achieve a coherent expressive picture of the situation. The result of the second phase is composed of rich pictures, named to pinpoint the need of expressing the problem situation in all its richness. The rich pictures, which are preferably literally pictures (i.e. drawings or diagrams), could take into consideration structures, processes, climate, people issues expressed by people, and conflicts.

The third stage is about finding the so called root definitions: the perspectives or motivations of each actor in the rich picture. This phase commences by exploring the different perspectives of actors, in a rather unstructured form. After all perspectives have been identified, a structured analysis is carried out on the key perspective using the CATWOE model development process. CATWOE stands for customers, actors, transformation, welthanschauung (worldview), owner, and environment.

In the fourth stage, a conceptual model is constructed using the root definitions. The conceptual models would ideally be diagrams that demonstrate how each actor envisions a system that can fulfill the root definitions. The fifth stage is to compare the conceptual models with reality, to find incongruence but also to understand how the real world can be improved to meet the conceptual models. How these improvements can be achieved, is explored in the sixth phase by developing specific ideas. The last stage, contains the action to actually carry out these improvements in the real world.

It is important to note that all these stages do not serve as a strict framework to follow but are open to interpretations, variations, and iterations. The process serves as base to achieve understanding among actors and identify objectives. The flexibility offered by this process, results in different adaptations that fit the best to the given problem, something we will discuss in the following section.

3.4.2 Simulation studies from a soft systems perspective

Pidd [Pid07] explains why and how SSM can be useful in a simulation study. Simulation revolves often around implementing code, whether it is in general purpose languages such as Java or in simulation environments such as Arena (where the logic is specified using building blocks). It is however important that a conceptual model is build first, to avoid implementing a simulation model that does not fully address the problem or that shows discrepancies with reality. The conceptual model that is developed in the fourth stage of SSM is ideal for developing a simulation model: it covers the different perspectives of the actors and, ideally, it went through much iteration. However, it is imperative that the conceptual model and the simulation model are consistent. This formulated in the following issue:

Issue 5. *Actors may have difficulties in keeping the conceptual system designs and detailed system designs mutually consistent.*

Kotiadis [Kot07] follows a similar approach as Pidd and uses the conceptual model gathered from SSM sessions. The focus lies mainly on the simulation study objectives that can be determined based on the definitions of efficacy, efficiency, and effectiveness. This extension of SSM [Che81a] is used to specify the Performance Measurement Model (PMM): this model is constructed similarly to the conceptual model of the classic SSM. Using the PMM, one can go about finding out performance criteria, breaking the performance criteria into specific monitoring activities, decide what action might be taken based on these activities, and decide which activities might be evaluated in a simulation model.

Baldwin *et al.* [Bal04] argue that Lehaney and Paul's [Leh96] approach, and later also used by Lehaney *et al.* [Leh08], is merely a first step towards combining SSM with simulation. Lehaney and Paul use simulation in the fourth phase of SSM to speed up the process by using quick-and-dirty simulation modeling. Simulation, in their view, can be used to enhance understanding and interpersonal communication of the stakeholders. However, contrarily to what is stated in other studies, the modeling effort should be present to understand the problem and should therefore not happen after the problem has been identified. Modeling becomes a way of communicating between stakeholders and the stakeholders should be involved into the modeling effort from the very beginning. They underline the need for an iterative process where stakeholders provide requirements for the modeling effort and the model produces new information for the stakeholder. This helps us formulating the next issue:

Issue 6. *Not every actor is involved in both the conceptual system design as well as the detailed system design process.*

A looser approach to soft methods is taken by Robinson [Rob01] who states that his "study involved a facilitated discussion around a simulation model about possible improvements to a problem situation". Instead of following a strict SSM process, he prefers to adapt the classic simulation process such as the one presented by Law and Kelton [Law00a]. The methodology has different steps: (1) defining the objectives, (2) conceptual modeling, (3) model development, (4) verification, (5) validation, and (6) facilitation. In contrast to classic simulation studies, the data used for this study was neither complete nor reliable. Because of this, a facilitated discussion was organized instead of performing experiments with the simulation model. The simulation model was not used "as a tool that could accurately evaluate alternative options, but rather as a focus of debate, a means for learning about the problem situation and for reaching an agreement to act". This multimethodological approach is presented in a diagram, shown in Figure 3.4.2, that highlights the different aspects that can be considered "hard" and the one that can be considered "soft".

Den Hengst *et al.* [Hen07] follow a similar approach as Robinson in using simulation models to facilitate a discussion. Their approach follows five steps: (1) conceptualize problem, (2) create and validate empirical model, (3) construct alternative models and conduct experiments, (4) choose most preferred solution, and (5) implement solution. They discuss a case study that provides some valuable insights in the use of simulation models for this kind of interactive purposes. They present the following problems or difficulties:

- Building the simulation models required a lot of time and expertise: 'building the simulation model both conceptually and empirically took an average of two working days per week over a period of five months for two modellers who had significant simulation modelling experience' [Hen07].

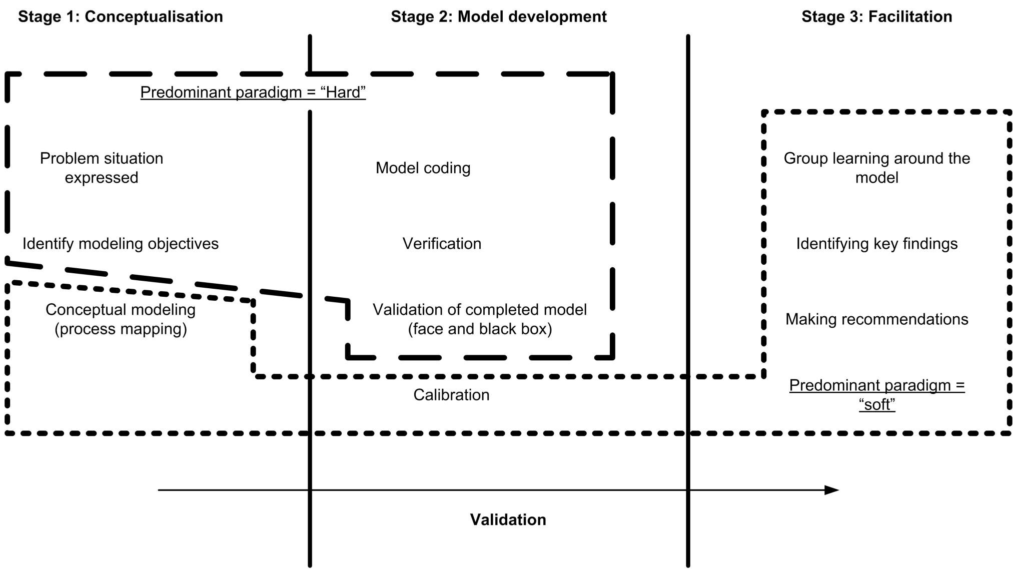


Figure 3.4: The multimethodological approach presented by Robinson [Rob01]

- Due to the complex code to build the simulation model, verification and validation was extremely difficult. In retrospect, they argue this could have been avoided because the validation did not add anything useful to the process.
- Lack of knowledge in simulation modeling resulted in a hard to accept model. The chosen simplifications and animation did not provide enough trust in the simulation model.
- Running the experiments took a long period of time. The chosen simulation environment required several hours to run full simulation experiments, which was unacceptable in an interactive session.

3.5 Discussion

It is clear that using simulation models as a base for discussion is useful and provides fruitful results. Aughenbaugh and Paredis [Aug04] provide a very thorough and to-the-point explanation as to what simulation can bring to design and decision making:

> Without simulation models, design relies on implicit knowledge. Implicit knowledge is unreliable in that designers do not know the assumptions and uncertainty in the knowledge explicitly. When decisions are coupled and require

input from several experts, there is no way to make tradeoffs using only implicit knowledge about uncertainties.

This leads us to the following issue:

Issue 7. *Without simulation models, actors may not able to predict the influence of their design decisions on the system design.*

Whereas traditional simulation studies focus on finding the optimal solution, soft systems approaches aim at achieving shared understanding in a group to support decision making. This is achieved by applying an iterative process that continues until all actors are satisfied with the model. However, as the studies have shown, using the simulation models does not go without any hurdles. The presence of various actors with diverging worldviews provides challenges to the simulation environment. This can be rephrased into the following issue:

Issue 8. *The system design is mostly not presented in a way that all actors can understand.*

Robinson [Rob02] identifies three modes of simulation practice: (1) simulation as software engineering, (2) simulation as a process of organizational change, and (3) simulation as facilitation. The predominant part of simulation practice takes place in the first mode (the construction of large and complex simulation models) whereas there is little to be found in the third mode. This implies that most simulation environments are constructed following a hard systems paradigm. According to Rosenhead [Ros01] this means that

1. problem formulation is in terms of a single objective;
2. there are overwhelming data demands;
3. people are treated as passive objects;
4. there is a single decision-maker with abstract objectives from which concrete actions can be deduced;
5. and there is an attempt to abolish future uncertainty, pre-taking future decisions.

This paradigm is implemented in modern simulation environments where users create one single detailed model per project (instead of various models for diverging opinions), users need to input a large amount of data, and there is a single optimal solution. If we were to use these simulation environments in a soft systems approach, a couple of issues would arise. A major issue is the construction the simulation model: simulation models take considerable time and expertise to develop. Although modern simulation packages relief developers from a lot of work, they are still not appropriate for interactive modeling sessions. Once build, the simulation environment requires some time to run full simulation experiments. Finally, the focus on a single model would restrict the exploration of the solution space, thus formulating the following issue:

Issue 9. *The system design process is mostly neither interactive nor iterative.*

3.6 Conclusion of the first part of the rigor cycle

During the first part of the rigor cycle, described in this chapter, we explored existing theories, models, and methods regarding the synthesis and analysis of systems. We have seen the evolution of SE from early conception to modern use. Simulation has turned out to be an important method to study systems: computing the dynamic aspects of a system helps supporting designers understand the systems better.

In light of the case study discussed in Chapter 2, modern design method that include the use of simulation models revealed to have some limitations. We have identified these limitations as “issues” throughout this chapter. The limitations are specific for the use of simulation models within a multi-actor environment. In the following chapter, we will initiate the design cycle by identifying constructs that can be used to solve these issues.

Chapter 4

Multiple Worlds: a multi-actor simulation-based design method

In Chapter 2 we identified research opportunities to support the design process of large-scale logistics systems, more specifically automated container terminals. In Chapter 3 we carried out an extensive literature study to identify theories that is applicable to exploit the research opportunity we have identified during the case study. We highlighted issues throughout the chapter that can be taken into consideration to propose a design method that integrates different theories to support a multi-actor design process that uses simulation models.

Following the design science research approach, which we discussed in Chapter 1, we need to enter the design cycle that shapes a viable solution. We will present the design cycle in different steps:

1. Present the issues that we have identified during our case study and found literature. These issues reflect requirements that guide us throughout our design cycle towards a suitable design method.
2. Based on these issues, shape a solution by proposing constructs that can be used in the design. These constructs help to propose a solution that satisfies the set of requirements reflected by the issues.
3. Using the constructs, we will compose our new design method using the proposed constructs, which concludes the design cycle.

4.1 Issues in systems engineering

During our case study, we have identified a research opportunity that we have used to guide us through the literature study. The latter suggested a shortcoming of suitable theory to completely solve the issues encountered during the case study. Both from the relevance cycle as from the rigor cycle, we can identify issues that we choose to solve to construct a multimethodological design method.

The set of issues identified during the relevance and rigor cycle are the following:

1. Without a well defined structure, the system design process may not be able to support the abstract, divergent and convergent thinking of system designers.

2. Without the ability to manage different system designs throughout the system design process, designers may not be able to choose among alternatives.

3. Without the ability to document assumptions, decisions and argumentations, actors may not be able to follow and audit the system design process.

4. Without the ability to compare different system designs throughout the system design process, actors may not be able to identify their preferred system design.

5. Actors may have difficulties in keeping the conceptual system designs and detailed system designs mutually consistent.

6. Not every actor is involved in both the conceptual system design as well as the detailed system design process.

7. Without simulation models, actors may not able to predict the influence of their design decisions on the system design.

8. The system design is mostly not presented in a way that all actors can understand.

9. The system design process is mostly neither interactive nor iterative.

4.2 Identifying suitable constructs

In chapter 3, we discussed existing theories that highlighted the issues presented in the previous section. These issues have been identified as relevant in many research projects, thus resulting in much published work. In table 4.1, we present non-exhaustive lists of methods and models that have been presented to tackle each individual issue. We aimed at restricting ourselves to work that is most closely related to the research presented in this dissertation. In our design cycle, we aim at designing a method that tackles these issues. We want to achieve that by integrating method and models presented in the previous chapter and Table 4.1. We present the notion of *construct* to describe related methods or models that can be used to tackle a set of issues. Therefore, to define constructs, we grouped theories from the previous that share a common denominator. The constructs are: "component based modeling", "different levels of specification", "structuring alternatives", and "participatory design".

4.2.1 Component based modeling

Den Hengst *et al.*[Hen07] and Valentin *et al.*[Val05] argued that simulation models take much time and effort to develop. Pre-defined simulation components let users build new simulation models by focusing on the structure without requiring knowledge about the inner behavior of the components. Another important feature of component-based modeling is modularity, which can be facilitated by port-based communication: such an approach has been adopted by Paredis *et al.* [Par01]. Once this approach is adopted, reusability of components becomes highly enhanced.

The notion of components has its origins in software engineering, which by now has become common practice for software development [Som04]. Component-based simulation has been adopted recently (compared to pure software engineering applications) but has quickly become a popular research topic [Ose04]. A comprehensive overview on simulation components is given by Verbraeck [Ver04a] who advocates the construction of a library of *easy to use* components to quickly create alternatives throughout a design process. Indeed, components provide the major advantage of being self-contained, reusable, replaceable and customizable. The use of simulation model components for design is also strongly advised in various other research for the same reasons [Ste95, Qin03, Kin04, Bal98b].

In Section 1.3 we introduced the DEVS formalism that supports hierarchy, modularity, and composability. These are the exact features to be exploited to support simulation-based design [Zei05, Hu05b, Zia07].

4.2.2 Different levels of specification

Specifying a system design gradually in different steps is required to deal with uncertainty, problem complexity, and cognitive limitations of designers [Goe95, Ull88, Hoo91]. A typical design process starts with an incomplete picture of the design problem, which makes is not feasible to directly start with a detailed solution. From a cognitive perspective, Visser [Vis06] notes that designing is an iterative process of generating designs, transform them and evaluate them. The difference between the first and last model is a question of degree of specification, completeness, and abstraction. A similar view is proposed by Goel [Goe95], who writes "Design, at some very abstract level, is the process of transforming one set of representations (the design brief) into another set of representations (the constract documents)". Yet another similar formulation is put forward by Ullman *et al.* [Ull88] as "a design problem typically begins with a set of functional constraints expressed very abstractly; during the design process, the level of abstraction of the design state is progressively reduced until is detailed enough to be manufactured".

During a design process, models can be constructed that are specific to pre-defined abstraction levels. Ullman *et al.* [Ull88] do this by selecting three levels of abstractions: 'abstract', 'intermediate', and 'concrete'. For each level, they define a conceptual model. Similarly, Schimdt and Cagan [Sch95] define an abstraction grammar which makes it possible to model components at different abstraction levels. Paredis *et al.* [Par01] go a step further by constructing simulation models at different levels of abstraction. This solution lets designers analyze simulation results throughout the design process, even in the initial phases characterized by rough and incomplete designs. Further down the process, abstract simulation models are replaced by more concrete implementations, up until the end, where highly detailed simulation models are used to gather precise simulation results.

4.2.3 Structuring alternatives

Generating alternatives is one of the fundamental steps in most design methods, which we discussed extensively in Chapter 3. Human's bounded rationality impedes a large exploration of the design space: design studies are mostly limited to a few competitive alternatives. The large amount of alternatives that could be generated in a multi-actor environment is usually kept to a bare minimum by scrapping (supposedly) unfeasible designs in the early stages of the design process. The need for structuring and comparing alternatives has induced researchers in proposing solutions such as trees, matrices, rankings and charts.

Controlled experiments described by Dwarakanath and Wallace [Dwa95] bring more insights into the way designers think. Designers tend to follow paths along conceptual tree-like structures to assess alternative designs. Branches are made when different alternatives are being considered. Whenever a solution seems undesirable, pruning takes place to remove unwanted paths. The experiments led to a number of observations that stress the fact that designers go through an iterative process of finding alternatives, evaluate them (either formal or on intuition) and eliminate alternatives that perform less than the others. Although this experiment mainly studied informal methods, another experiment, documented by Girod *et al.* [Gir03], discusses that using formal approaches tend to spawn better results. The formal approach forces designers to structurally evaluate their design and document the design process.

4.2.4 Participatory design

The aforementioned constructs support a design process in a multi-actor environment: simulation components help to quickly generate solutions, abstraction levels lead the design process from rough artifacts to detailed solutions, and structuring the solutions supports designers in assessing too many alternatives.

A multi-actor design process is an iterative process where different actors try to achieve their own, sometimes conflicting, goals. A simulation environment supports analyzing decisions which are otherwise assessed based on the designer's implicit knowledge. By having a means to compare different alternatives (pertaining to the different goals of the actors) analytically, grounded decisions can be made and insight can be provided into unforeseen failures or unexpected successes of certain decisions. The collaborative exploration of the solution space entails the learning process which is sought after by traditional soft systems approaches.

Actors involved in a design process are characterized by their goals e.g. introducing innovative solutions, reducing costs, feasible solutions, and adhering to external regulations. In simulation studies, conceptual designs are developed that are the results of negotiations between the actors. Major decisions are made before the actual simulation study has been performed and are based on informal assessments. By bringing formal techniques closer to the conceptual design process, discussions can be grounded and shared understanding can be achieved. This leads to an approach that is 'soft with a hard centre' [Rob01].

An important note is provided by Shipman and McGall [Shi97]. They argue the need for documentation during a multidisciplinary design process. Not only does documentation serve to capture design decisions and rationale for later use, but also as a reference for external parties that want insight into the design process. Capturing the design process serves to support argumentation ("Why did we take this decision?"), documentation ("How did we take this decision?") and communication ("We took this decision.").

4.3 Towards a possible solution

In Chapter 3, we have introduced relevant literature on design methods from a SE perspective. This led us to identify the main constructs needed for a design method, but also various shortcomings of the discussed approaches. The combination of these construct would entail a multimethodological approach. We summarize the constructs that we will use in the design cycle:

Construct 1 Modularity and component based simulation: as proposed by den Hengst *et al.* [Hen07], simulation models require a lot of effort and time to develop. Simulation models that are constructed of pre-defined components can be malleable to support an iterative and interactive design process, without requiring an intervention by simulation experts.

Construct 2 Different levels of specification: in conceptually challenging domains such as engineering design, gradually specifying in more detail a design is essential to cope with large amount of data and complexity.

Construct 3 Structuring alternatives: as fundamental part of design methods, the definition and evaluation of alternatives enables actors to diverge and converge throughout the system design process.

Construct 4 Participatory design: a system design process is seldom performed by a single actor or even a single person. Many actors are involved and all of them try to achieve their own objectives. A system design process should support a certain convergence of interests of all actors.

In the previous chapter, we introduced studies found in literature that present each construct, or part of it, as a solution to the identified issue. We clarify the relationship between issues and constructs in Table 4.2, as identified in studies cited in Chapter 3.

In Table 4.3 we summarize the literature studied in Chapter 3 on why and how each construct is be able to tackle each issue.

4.4 The "Multiple worlds" design method

The raised issues during the relevance and rigor cycle called for an identification of possible constructs that can be combined into a system design method. The combination of these methods should result into a multimethodological method (as defined in Chapter 3) that aims at taking the best of both worlds, being the hard and soft systems approaches.

We will continue this chapter by presenting how we propose to provide a method that integrates the constructs described earlier. A simulation based design method is presented to be used in a multi-actor context. We will start by formalizing the method. We will continue by discussing the semantics of the formalism and discuss the design process that is entailed by the design method.

4.4.1 The formalism

In [Hof92] the need for formalizations is discussed when new methodologies for IS are introduced. This need is present in our case and the arguments for formalization are applicable. The reasons given are to reduce the risk for ambiguity, to facilitate automated support and to define properties. The risk for ambiguity refers to different possible interpretations when there is a lack of formalization. With an absence of formalizations, only reference examples or vague descriptions are available. This can lead to an erroneous application of the given methodology or process description. Eliminating this ambiguity leads us to the second reason: facilitating automated support. A well structured formalization greatly supports an actual implementation. In extreme circumstances, an implementation could be a one-to-one mapping of the formalization to a programming language. Lastly, a formalization helps us define and further elaborate properties. Equivalence with and mapping onto other formalizations could be one of them.

We present a formal description of the conceptual core of the 'Multiple Worlds' design method. This formal description assumes the use of the DEVS formalisms to model the system. Various modeling formalisms are at hand to model complex systems depending on the system domain. Due to this large variety of modeling formalisms, it is unfeasible to apply it to every single one of them. Generalizing the "Multiple Worlds" formalism by abstracting away from a specific formalism would therefore be desirable. However this would require us to precisely specify the requirements of the abstract modeling formalism. In [Van00], the author discusses the large variety of modeling formalisms and discusses the possible mappings between these formalisms. Following the formalism transformation graph, it can be concluded that the expressiveness of DEVS is able to represent many discrete and even continuous modeling formalisms. Our aspiration towards a "Multiple Worlds" formalism independent from the system domain, thus the modeling formalism, can be fulfilled by using the DEVS formalism.

A DEVS atomic model, as defined by Zeigler and Praehofer [Zei00], is a structure

$$M = \langle X, S, Y, \delta_{int}, \delta_{ext}, \delta_{con}, \lambda, ta \rangle \tag{4.1}$$

where

- X: set of external input events
- S: a set of partial states (excluding the elapsed time)
- Y: a set of outputs
- $\delta_{int}: S \rightarrow S$: internal transition function
- $\delta_{ext}: Q \times X^b \rightarrow S$: external transition function where $Q = \{(s, e) \mid s \in S, 0 \leq e \leq ta(e)\}, e$ will be defined later
- X^b is a set of bags over elements in X where $\delta_{ext}(s, e, x) = s$
- $\delta_{con}: S \times X^b \rightarrow S$: confluent transition function
- $\lambda : S \rightarrow Y^b$: output function generating external events at the output
- $ta : S \rightarrow \mathbb{R}_0^+$: time advance function
- e is the elapsed time since last state transition

In the above formalism, we can see that S is the set of states that a model can have. The set of states is defined as the Cartesian product of the sets belonging to each state variable: $S = s_1 \times s_2 \times \ldots \times s_k, k \in \mathbb{N}$.

A DEVS coupled model, as defined by Zeigler and Praehofer [Zei00], is a structure

$$N = \langle X, Y, D, \{M_d \mid d \in D\}, EIC, EOC, IC \rangle \tag{4.2}$$

where

- $X = \{(p, v) \mid p \in IPorts, v \in X_p\}$ is a set of input ports and values
- $Y = \{(p, v) \mid p \in OPorts, v \in Y_p\}$ is a set of output ports and values
- D is a set of the component names

- Components requirements:
  - components are DEVS models, for each $d \in D$,
    $M_d = (X_d, Y_d, S, \delta_{ext}, \delta_{int}, \lambda, ta)$ is a DEVS as defined in 4.1 with
    $X_d = \{(p, v) \mid p \in IPorts_d, v \in X_p\}$
    $Y_d = \{(p, v) \mid p \in OPorts_d, v \in Y_p$
- Coupling requirements:
  - external input coupling connects external inputs to component inputs
    $EIC \subseteq \{((N, ip_n), (d, ip_d)) \mid ip_N \in IPorts, d \in D, ip_d \in IPorts_d\}$
  - external output coupling connects components outputs to external outputs
    $EOC \subseteq \{((d, op_d), (N, op_N)) \mid op_N \in OPorts, d \in D, op_d \in OPorts_d\}$
  - internal coupling connects component outputs to component inputs
    $IC \subseteq \{((a, op_a), b, ip_b)) \mid a, b \in D, op_a \in OPorts_a, ip_b \in IPorts_b\}$
  - However, no direct feedback loops are allowed, i.e., no output port of a component may be connected to an input port of the same component: $((d, opd), (e, ipd)) \in IC$ implies $d \neq e$

A 'multiple worlds' is a structure

$$mwSim = \langle P, A, \mathcal{L}, \zeta, \eta, Z, \mathcal{Z}, \rho \rangle \tag{4.3}$$

where

- P is the set of possible triples (N, L, t) with
  - N is a coupled model as defined in 4.2
  - L is the set of components of N, $L \in \mathcal{L}$ (definition will follow)
  - t is the creation time of N, $t \in \mathbb{R}_0^+$
- A is a tree defined as $(P, <_{t,z,L})$
  - P as defined above
  - $<_{t,z,L}$ is the ordering relationship defined by the creation time, the applied decisions, and the set of components; $z \in \mathcal{Z}$ and $L \in \mathcal{L}$
- $\mathcal{L} = \{L_0, L_1, \ldots, L_k\}$ is the set of m component sets, $k \in \mathbb{N}; \forall L \in \mathcal{L}$ and $\forall c \in L \mid c$ is a coupled model as defined in 4.2
- $\zeta : L_i \to L_j; L_i, L_j \in \mathcal{L}$ is the component transformation function where $\zeta(c_{L_i}) = c_{l_j}; c_{L_i} \in L_i, c_{L_j} \in L_j$

- $\eta : S \times R \to S$ is the initialization selection function that selects a specific state from the set of states such that $\eta(s,r) = s_{initial}$ with $s, s_{initial} \in S, r \in R$ and R is the set of sequences for which each sequence selects a specific states, thus if $s = (w_{s_1}, w_{s_2}, \ldots, w_{s_j}), r = (w_{r_1}, w_{r_2}, \ldots, w_{r_k}), s_{initial} = (w_{s_{initial_1}}, w_{s_{initial_2}}, \ldots, w_{s_{initial_l}})$ then $j, k, l \in \mathbb{N}_0$ and $j = k = l$

- $Z = \{A_{component,r}, R_{component,r}, A_{coupling}, R_{coupling}\}$ is the set of decisions of which
  - $A_{component,r}$ adds the component $\eta(s,r)$; s and r as defined above
  - $R_{component,r}$ removes a specific component
  - $A_{coupling}$ adds a coupling as defined in 4.2
  - $R_{coupling}$ removes a coupling as defined in 4.2

- $\mathcal{Z}$ is the set of all k-tuples $(a_1, a_2, \ldots, a_k)$ with $a_1, a_2, .., a_k \in Z, k \in \mathbb{N}_0$

- $\rho : P \times \mathcal{Z} \to P$ is the branching function where $\rho(p_i, b_j) = p_h$
  - $p_i, p_h \in P \wedge p_i <_{t,z,L} p_h$
  - $b \in \mathcal{Z}$

The structure presented in (4.3) consists of eight main parts: system models, a tree, a set of sets of components, a component transformation function, an initialization function, a set of decisions, the set of all sequences of decisions, and the branching function. To follow the design process as discussed in Chapter 3, we need to have the ability to branch a certain model into various new models containing the possibilities one wants to asses using a predefined set of components. A set of sets of components, denoted $\mathcal{L}$, is provided throughout the design process. Each node, from the set of possible nodes P, has a model constructed using components from a set of components. A component transformation function is used to transform a model specified using one set of components into another specific set of components.

The function ρ, named the branching function, takes as parameters a model and a sequence of decisions from $\mathcal{Z}$ to construct a new model. The set of decisions is applied to the model's structure. Hu *et al.* [Hu05a] state that structural changes can be reduced to four main operations: add a component, remove a component, add a coupling and remove a coupling. Updating a component is not in the previous list of changes because it is equivalent to a combination of the four basic operations. We have to note that adding a component of the specific set of components needs a initialization function that sets the initial state of the model from the set of all possible states S. This can be compared to parameterize a specific component.

The branching function generates a new model each time it is applied to an existing model. To structure this generation of new models we use a tree for which the root node keeps the initial model. Each node in the tree results from changes that were applied to its parent node and provides a model as a basis to its possible children. Leaves of the tree therefore contain models for which no changes have been defined yet.

4.4.2 Semantics of the formalism

In Section 4.4.1 we presented a formalism that requires semantics to be used in a design process. The formalism is composed of the tree with its nodes, the decisions, and the branching function. We will present each construct with its semantics separately and present a complete tree in Section 4.4.2.

The tree and its nodes

The results from the experiments discussed in Section 4.2.3, lead us to propose a conceptual tree as a way to structure a design process. The tree elegantly embodies the generation of alternatives with branches. In light of the aforementioned constructs, the nodes of the tree contain alternative models (each model representing a design) that are constructed using pre-defined components. The breadth of the tree is defined by the number of alternative designs.

As discussed in the formalism, the tree is a structure that is composed of nodes and edges. Each node contains an design: during the design process, a design under investigation can be further specified by deciding upon structure and parameters of the design. This view on design is given by Churchman, who defines the design of a system as 'the design of components and their relationships' [Chu71]. Based on the formalism presented earlier, this definition can be operationalized as: 'the design of a system is the specification of the structure and parameters of that system'.

The use of several sets of components, specific to each node, can support a design process that starts from an underspecified design and goes towards a better detailed design. A design is said to be underspecified when one or more sub-structures or parameters are not defined. Each underspecified design needs to have a simulation model that can generate performance indicators of that design. This, of course, poses a challenge: an underspecified design needs a simulation model that reflects the uncertainty attained by the elements of the design that have not been defined. This means that from a single design, a multitude of designs can spring: for every structural decision or parameter that has not been set, the complete domain of that structural decision and parameter is used to variate the input and to estimate the output, which gives us the range of the parameter.

In the formalism presented earlier, the input is either every decision that can be applied on the model, denoted as input θ, or the input vector for the DEVS model that is denoted as input X that is fed by the experimental frame, which 'specifies the conditions under which the system is observed or experimented with' [Zei00]. It is important to note that in both formalisms, there is a clear distinction between input variables that influence the incoming events (of the experimental frame) and the incoming decisions (of the users). Each design is specified in terms of its structure and parameters that are inputted as design decisions. Supporting underspecified designs means that we can help designers in coping with a potential cognitive overload: taking every decision at once would pose a true challenge to the designers [Law00b]. During the design process, a fixed order is followed of choosing a decision from the complete set of possible decisions D. This order is created by selecting components from several sets of components. The design is defined by its structure and parameterization: the model components it contains, its interconnected state and its parameters. Both the chosen set of components and its structure and parameterization define the relationship a node has with its parent, children and siblings. The relationships are presented by the edges in the tree. The parent-child relationship defines different sets of components whereas at one level, each sibling is an alternation of one another using the same set of components.

It is important to note that both θ as X can contain many possible values that have to be decided upon during the design process. In a multi-actor (or multi-designer as each designer is also an actor) design environment, many struggles occur on deciding upon the contents of θ and, eventually, consensus has to be achieved to decide upon the final set. It is exactly on this challenge that the formalism presented here merges the formal system's perspective with the, rather, informal actor's perspective, which is also discussed by De Bruijn and Herder [Bru09]. Checkland's notion of learning during the process [Che06] is exploited by allowing the designers to explore different values for θ and using the simulation models to study the behavior and predict the output of the system. Designers need to make trade-offs to achieve a design that satisfies everybody involved in the process.

Branching with the decisions

The design process has a limited set of classes of decisions that can be made to establish a relationship between two nodes. When a specification of a child node is going to be established, the decision that can be made on the parent designs that results in the child design is limited to the specialization of the component or parameter that is defined for the given set of components. The specification decisions are needed to construct alternative specified children of a parent node. The branching function needs to be executed on a node to construct new nodes that will have alternative specifications relationships with the initial node. As formalized earlier in Section 4.4.1, the branching function takes a node and a decision as input and outputs a new node that is connected to the input node.

Branching has a peculiar meaning in the design process: the designers are questioning a possibility and question the results of their decisions. Therefore, at each level of the three, a specific answer needs to be questioned: "What will happen to the output measures of the design when I decide upon A", with A being the class of the decisions that need to be made at a specific level in the tree. Again, by reflecting back on the work of Checkland [Che81b], De Bruijn and Herder [Bru09], and Robinson [Rob01], we postulate that each branching is made to allow actors to explore the restricted solution space that remains after every decision and to discuss which design is not suitable for the given problem in accordance with each actor's stake in the process.

The complete tree

The tree guides actors or designers through the design process. The outputs of the upper nodes specify ranges that would preferably contain the more restricted ranges of the lower nodes. We note that this is not always the case: if we would need to ensure that the output range of each underspecified design should contain the union of ranges of the designs that can spring from the initial design, we would always have to present a range that is too broad for any practical use. Therefore, depending on the techniques used to construct the simulation model of the underspecified design, a trade-off has to be made between keeping exact ranges and ranges that are usable in practice. Ultimately, the design process will be about progressively specifying more about the design that restricts the output range to something that *satisfice* (in Simon's terms [Sim96]) the actors involved in the design process.

Figure 4.1 shows a possible design process of three levels, each level having a set of components specified at the left upper corner (being A, B, and C). At each level of the tree, the appropriate class of decision needs to be made to specify the design further. Each decision leads the designers to a node containing a design that is specified in terms of the upper classes and the current class of decisions, whereas the lower decisions are still constant, allowing a ceteris paribus assumption. This helps in the comparison of the nodes at one level and relieves designers from the burden of deciding on each specification at once. Following the formalism, this tree can be defined as follows:

- $N = \{(n_1, t_1, L_A), (n_2, t_2, L_A), (n_3, t_3, L_A), (n_4, t_4, L_B), (n_5, t_5, L_B), (n_6, t_6, L_B), (n_7, t_7, L_B), (n_8, t_8, L_B), (n_9, t_9, L_B), (n_{10}, t_{10}, L_C), (n_{11}, t_{11}, L_C), (n_{11}, t_{11}, L_C), (n_{12}, t_{12}, L_C), (n_{13}, t_{13}, L_C), (n_{14}, t_{14}, L_C), (n_{15}, t_{15}, L_C)\}$.
- $\mathcal{L} = \{L_A, L_B, L_C\}$
- $D = \{d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8, d_9, d_{10}, d_{11}, d_{12}, d_{13}, d_{14}, d_{15}\}$

Issue	Models and methods
Issue 1: unstructured design process	common design methods analyzed and clarified using mathematical measurement theory, Otto [Ott95]; the exploration model of Smithers [Smi89] and an extension of Banares-Alcantara [BA91]; materials and process approach to engineering design, Dieter [Die83]
Issue 2: manage different system designs	axiomatic framework for engineering design, Hazelrigg [Haz98, Haz99]; multiple perspectives in decision-centric design, Mocko [Moc06]
Issue 3: document the process	using audio recordings, Potts and Bruns [Pot88]; using architectural deltas to recover design decisions, Jansen *et al.* [Jan08]; survey on existing documentation methods and tools, Shahin *et al.* [Sha09]
Issue 4: compare different designs	axiomatic framework for engineering design, Hazelrigg [Haz98, Haz99]; simulation study, Banks [Ban00]; methods to support engineering design, Pahl and Beitz [Pah95]
Issue 5: mutually consistent conceptual and detailed designs	axiomatic framework for engineering design, Hazelrigg [Haz98, Haz99]; simulation based design using composable models at different abstraction levels, Paredis *et al.* [Par01]; linking hard to soft systems in simulation studies, Pidd [Pid07]; using soft systems methodology to determine the simulation study objectives, Kotiadis [Kot07]
Issue 6: involvement of actors throughout the process	healthcare management using a soft approach and simulation, Baldwin *et al.* [Bal04]; multiple perspectives in decision-centric design, Mocko [Moc06], soft systems approach for hospital simulation, Lehaney and Paul [Leh96]; simulation modeling for problem understanding, Lehaney *et al.* [Leh08]; using soft systems methodology to determine the simulation study objectives, Kotiadis [Kot07]
Issue 7: predict the influence of design decisions	axiomatic framework for engineering design, Hazelrigg [Haz98, Haz99]; simulation based design using composable models, Paredis *et al.* [Par01]; multiple perspectives in decision-centric design, Mocko [Moc06]; the role of simulation in design studies, Aughenbaugh and Paredis [Aug04]
Issue 8: understandable presentation	detailed 3D visualization for construction projects, Kamat *et al.* [Kam05], simulation output in 3D virtual environments, Akpan and Brooks [Akp05a, Akp05b], aggregating output in 3D visualizations, Dangelmaier *et al.* [Dan08]
Issue 9: interactive and iterative process	soft systems methodology in simulation studies, Kotiadis [Kot07]; soft systems to facilitate simulation studies, Robinson [Rob02]; using simulation components throughout the design process, Paredis *et al.* [Par01]; soft systems approach for hospital simulation, Lehaney and Paul [Leh96]

Table 4.1: Non-exhaustive lists of methods to tackle each issue individually.

	Construct 1: modularity and component based simulation	Construct 2: different levels of specification	Construct 3: structuring alternatives	Construct 4: participatory design
Issue 1: unstructured design process		●	●	
Issue 2: manage different system designs	●		●	
Issue 3: document the process				●
Issue 4: compare different designs	●		●	
Issue 5: mutually consistent conceptual and detailed designs		●		
Issue 6: involvement of actors throughout the process		●		●
Issue 7: predict the influence of design decisions	●			
Issue 8: understandable presentation				●
Issue 9: interactive and iterative process	●		●	●

Table 4.2: The relationships between the issues and the constructs: each construct solves one or more issues in a non-exclusive and non-equally partial manner indicated by a ●.

Construct	Issue	Why	How
1	2	Verbraeck [Ver04a] pinpoints that component based design can be used to quickly assess alternative designs using simulation.	By providing domain specific components that can be used to structure and parameterize a system design.
1	4	By having a way of easily constructing and simulating system designs, comparisons become possible.	Components can be used as discussed above.
1	7	Simulation models are capable of supporting actors in achieving insight into the behavior and output of a system design.	Each component is a simulation model that can operate in connection with other components.
1	9	Components can be used to quickly compose new system designs.	Each component can operate with other components to form a complete simulation model of a system design.
2	1	A design process can be structured that allows actors to gradually go from a underspecified design to a detailed design using classes of specification.	By providing simulation models that support varying level of specification.
2	5	The difference between conceptual design and detailed design is the level of specification. If the design process allows to gradually specifying the design, consistency can be preserved between conceptual and detailed design.	(see previous explanation)
2	6	Without a clear demarcation between the conceptual and detailed design process, all actors can be involved in the complete design process from the beginning.	(see previous explanation)
3	1	As in the first issue, a structure helps designers in managing alternatives.	The structure needs to be appropriate to reflect the diverging and converging nature of design processes.
3	2	Whenever designers have to deal with multiple designs, a clear structure is required to manage the different models.	By providing a way to manage, visualize, and edit different designs.
3	4	Alternatives need to be structured to make an easy comparison.	By providing a clear structure, keeping into consideration the previous points.
3	9	During the design process, new alternatives have to be constructed to assess new designs. Iterative cycles are needed to revise old designs.	A clear structure helps iterate through different designs without loosing track of all alternatives.
4	3	All actors involved in the process should be able to document their assumptions and decision.	By providing the necessary support to document everything that is needed.
4	6	By including every actor in the design process, actors can go through a learning process and negotiate decisions.	By using the structure and simulation models specified in the above points.
4	8	If every actor needs to be included, the way of presenting a design should be comprehensible by all.	Providing 2D and 3D visualization that provide a clear view on the design.
4	9	If every actor needs to be included, the many opinions need to be taken into consideration, resulting in alternative designs.	As pointed out in the explanation for construct four with issue six.

Table 4.3: Why and how the constructs tackle the identified issues.

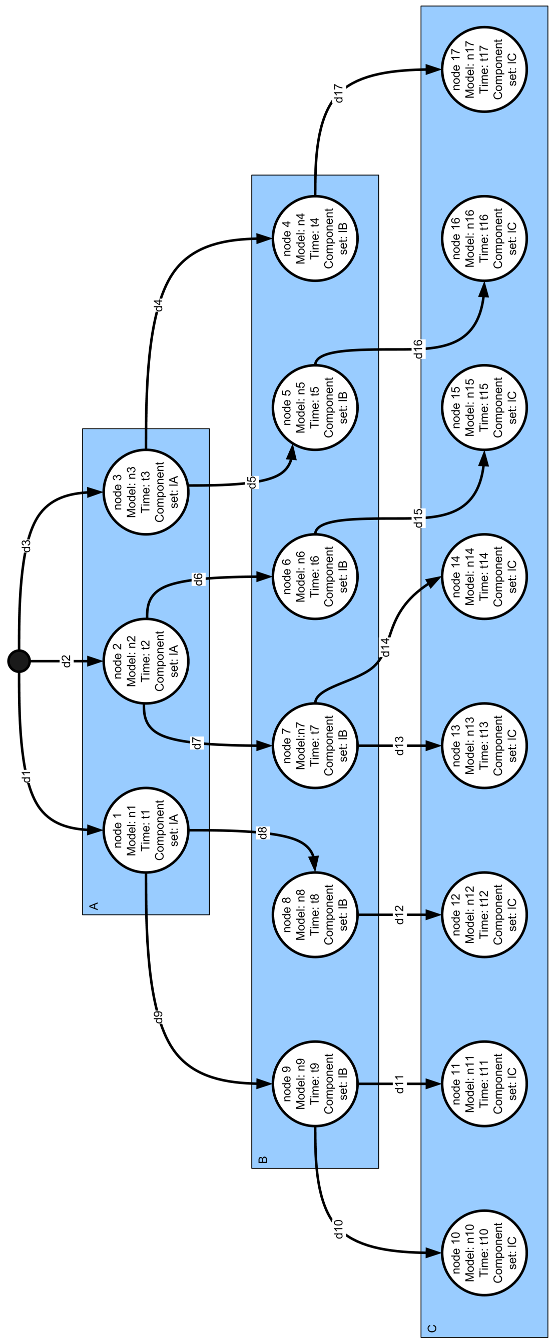


Figure 4.1: The structure of a multiple worlds design process (variablenames are explained in the text)

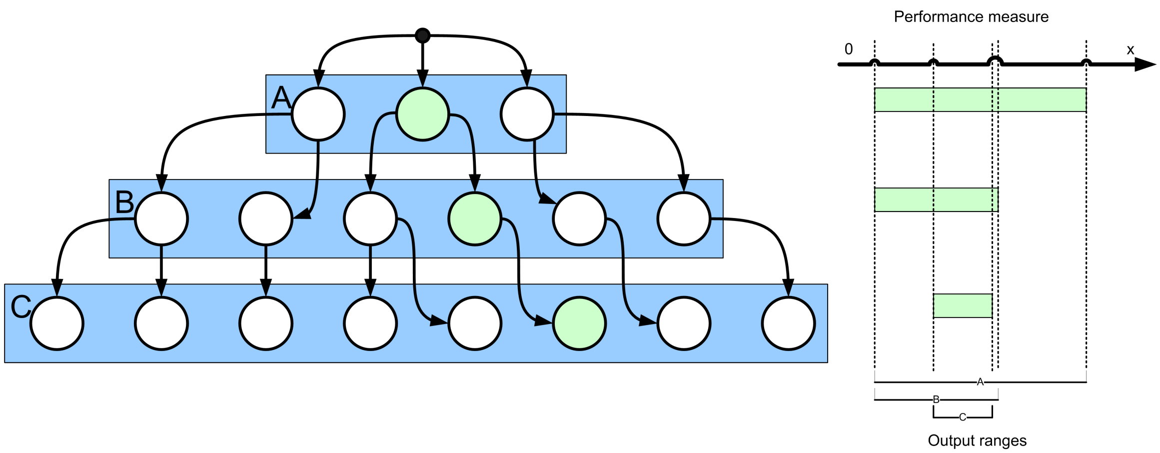


Figure 4.2: Output ranges of the different levels of specification.

Each design has a model that outputs performance indicators that are important to the designers. The width of the output's range is in relation to its model position in the tree: less specified model result in a broader range than the more specified models, which have a narrow range, even up to a point estimate. The models closer to the node of the tree reflect the uncertainty designers have during the initial phases of the design process. Lower down the tree, several decisions are made that restrict considerably the solution space. If we visualize this using output ranges and point estimates, we can take Figure 4.2 as an example. The output can guide the designers through the design process by providing an estimate on where the design is located in the broader range given by less specified models. The designers can steer the design process by inputting decisions that reduce the range to the sub-range of their interest.

The tree guides designers through the design process. The outputs of the upper nodes specify ranges that contain the more restricted ranges of the lower nodes. Therefore, the design process will be about progressively specifying more about the design that restricts the output range to something that satisfice (in Simon's terms) the actors involved in the design process. This process is visualized in Figure 4.3.

4.4.3 Instantiating and using the method

To use the "Multiple Worlds" method, we will need to go through several distinct simulation studies as discussed by Balci [Bal98a]. The simulation study proposed by Balci is shown in Figure 4.4.

To develop the set of components, thus component libraries, of $\mathcal{L}$ we have to go through each distinct step of the simulation life cycle, taking into account a class of problems instead

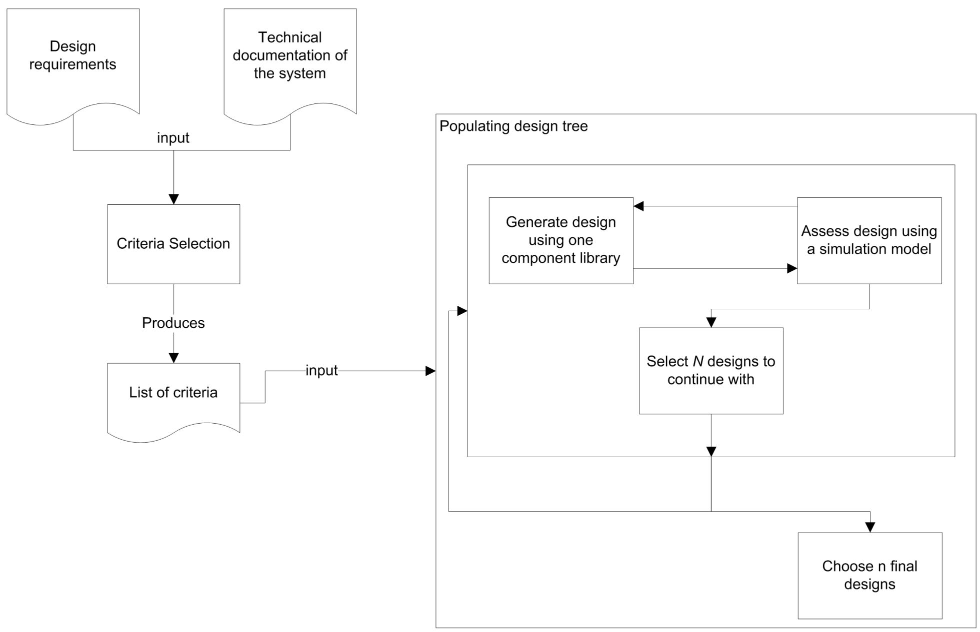


Figure 4.3: A flow chart showing the design process using the multiple worlds method. Taken the design requirements and technical documents of the system, actors define the criteria they are going to use to assess a design. Once the list is ready, the actors enter an iterative cycle of generating alternatives and assessing these alternatives using the criteria. Using several component libraries, they refine their design until a (set of) final design(s) has been selected to conclude the design process.

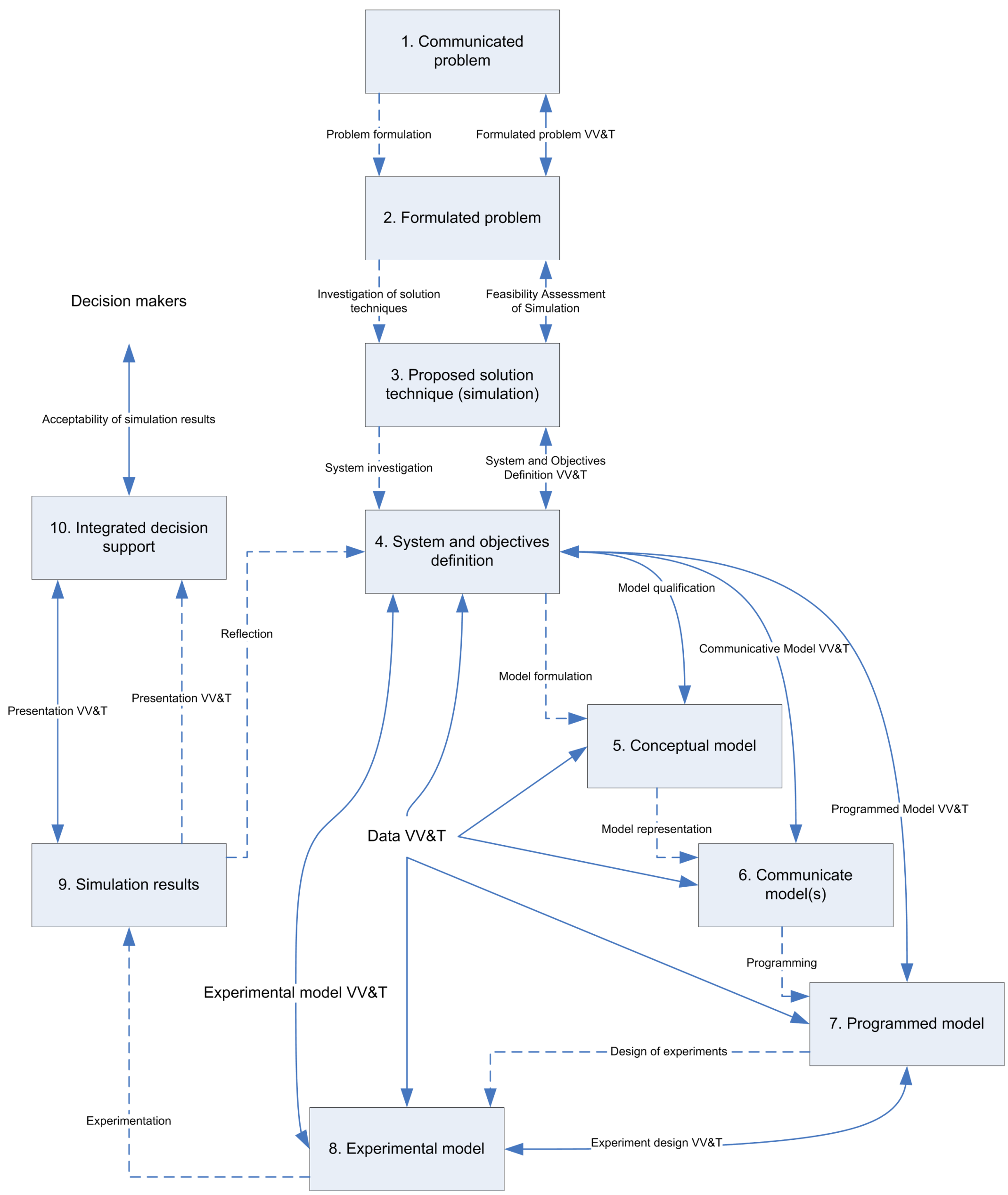


Figure 4.4: The simulation life cycle as proposed by Balci [Bal98a] (adapted from the original figure using the same source; VV&T stands for validation, verification, and testing). In our research, this life cycle will be used in 1) constructing the component libraries and 2) carrying out the design studies using the "Multiple Worlds" method.

of a single problem as described in step 1 and 2 of the life cycle. From the conceptual model onwards (step 5), the modeler cannot take a complete simulation into consideration but only the distinct model components. This means that in step 8, experimental models have to be constructed to validate the model. In simulation studies, a model is constructed of an existing system to analyze the system and propose improvements to the system. One of the validation steps, in this case, would be to compare the performance indicators of the real system with those of the simulation models. When simulation studies are carried out to support the design process, this approach is not feasible due to the lack of an actual system. The validation study becomes an effort to increase the trust the designers have in the model rather by adopting validation techniques on the individual model components, such as techniques discussed by Balci [Bal98a].

The simulation life cycle is also relevant in using the “Multiple worlds” method, although it will be adapted to the divergent nature of the method. The first three steps will remain the same, whereas step four up until step nine will differ. The latter will be carried out for each node, by defining different objectives, constructing different conceptual models using the simulation components, assemble these model components into an experimental model, and, finally, study the simulation results. Step four up to step nine will therefore be part of the cycle identified in Figure 4.3.

4.5 Summary

In Chapter 3, we have seen that designing systems in a multi-actor environment is complex from the system as well as the actor perspective. Although classic operations research (OR) and SE did not fully accomplish to provide successful approaches, they did provide valuable techniques to analyze systems. Due to an increased interest in facilitating the design process from an actor perspective, we do have insights in how actors behave and how we can support them. Multimethodological approaches are being developed and researchers commit themselves in covering both perspectives of complexity in design methods.

We have presented the constructs that are needed tackle the issues identified in Chapter 3. The totality of constructs is grouped into a method that is formal in describing the design artifact (using simulation components) and informal for the design process (actors have to explore the design space, without being restricted by an inflexible design method). Therefore, it is not a matter of “picking sides”: to fully cover the complexity present in a design process, constructs from both approaches can be used.

The “Multiple Worlds” method combines constructs from both the hard as well as the soft system’s perspective that can be used in the design process of container terminals. In the next chapter, we will operationalize this design method using these constructs and present a design environment that has been implemented to support design processes using the “Multiple Worlds” method.

Chapter 5

Using the "Multiple Worlds" method to design automated container terminals

In Chapter 4 we have presented the Multiple Worlds method to support the design process of logistics systems in a multi-actor environment. Following the design science methodology, we want to instantiate the method in a specific domain in order to be able to show that the method can be instantiated and to experiment with the method and feed it back to a business environment. In this chapter, we will present our instantation of the method. This entails the development of a software implementation to support the method and the description of the design process following the Multiple Worlds method. In accordance with the case study discussed in Chapter 2, we have selected automated container terminals to be the logistics system for our instantiation.

In this chapter, we will present an instantiation of the Multiple Worlds method by discussing the supporting software implementation and the design method applied to automated container terminals. We will present each part of the software implementation by first presenting it in a generic manner. Hereafter, we will discuss the application of the implementation specifically to support the design process of automated container terminals. Figure 5.1 shows the generic architecture (without actual implementation) of the software implementation in the upper part, the specific implementation is presented in terms of the actual software components in the lower part. The primary user interface of the software implementation is *MultipleWorldsGUI*: this presents a tree and the functionalities to add and remove new designs. The user commands are send to *MultipleWorldsStructure* that contains the actual tree structure with connections to related components to run and visualize the models. Users need a way of defining system designs: this is reflected by the *Input Component* in the generic architecture. In the implementation, the component *CADTransformer* translates the CAD drawings from AutoCAD to a description that can be used both as a simulation and a visualization model. Once the right description of a design has been

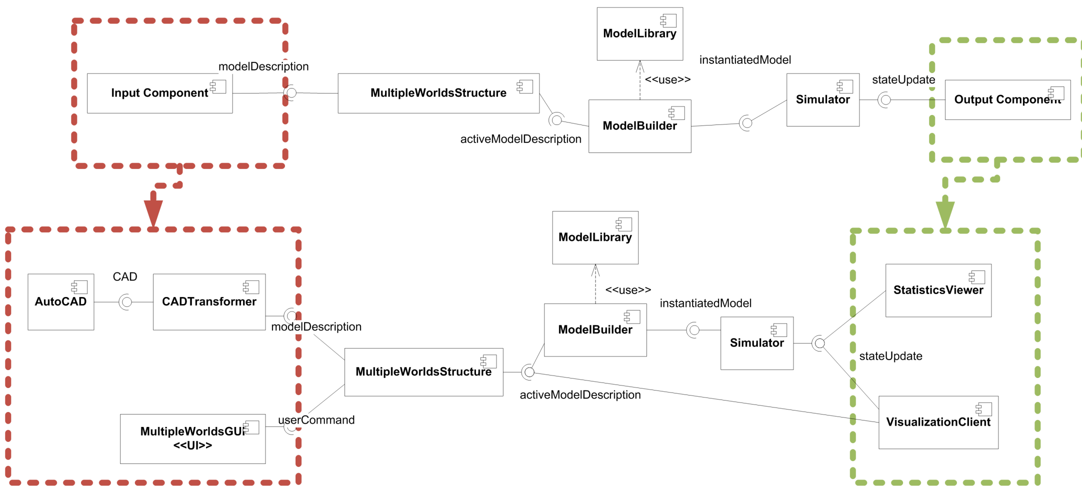


Figure 5.1: The software implementation is, in general, composed of a component to define the system design, a structure to maintain the tree, a library of components and a simulator to run simulation experiments, and, finally, to have a quantitative output using the output component. The software implementation that supports the design process of automated container terminals has several components that fulfill the following tasks: inputting a detailed design, structuring the designs in a design tree, running the simulation experiments, and visualizing the output of those experiments.

constructed, this can be read by the component *ModelBuilder* that instantiate a correct parameterized model using model components contained in *ModelLibrary*. The output of the simulation experiments need to be outputted appropriately. This is reflected by the *Output Component* in the generic architecture. In the implemention, after the model has been instantiated, it can be run and the generated output can be send to *StatisticsViewer* to calculate the statistical output and to *VisualizationClient* to output the run using a 3D environment.

We will discuss each part of the software implementation separately:

- the Multiple Worlds implementation: it consists of the components *MultipleWorldsGUI* and *MultipleWorldsStructure*, and will be presented in Section 5.1;
- the CAD implementation to input detailed designs: it consists of the components *AutoCAD* and *CADTransformer*, and will be presented in Section 5.2;
- the simuation implementation: it consists of the components *ModelBuilder*, *Simulator*, and *ModelLibrary* and *MultipleWorldsStructure*, and will be presented in Section 5.3;

- and the visualization clients consisting of the components *StatisticsViewer* and *VisualizationClient*, and will be presented in Section 5.4.

Hereafter, we will present how the process takes place using this software, based on the discussion of the semantics in Chapter 4. We will conclude this chapter discussing possible alternations on the instantiation and reflect on ways to instantiate the method for other logistics systems.

5.1 Implementation of the "Multiple Worlds" formalism

The major contribution of this research is the "Multple Worlds" method, which allows actors to alternate and iterate during the design process. In the instantiation, two components are responsible for supporting the design method: the component *MultipleWorldsGUI* and the component *MultipleWorldsStructure*. The former provides an interface to the actors to construct the tree as discussed in 4.4.1, the latter provides the internal structure to store the different simulation models and maintain the proper relationships between the models. We will first discuss in more detail the graphical user interface (GUI) in Section 5.1.1 and we will continue by discussing the internal structure in 5.1.2.

5.1.1 The "Multiple Worlds" user interface

The "Multiple Worlds" GUI depicted in Figure 5.2 lets the actors structure either high level or low level models in a tree, and send these models to the "Multiple Worlds" internal structure to run the simulation experiments.

The GUI has been developed in wxWidgets, using a custom *wxTree* widget to visualize and structure the tree. The customized part of the tree holds the necessary references to the stored high level or low level designs (in this part of the software implementation, we prefer using the term "design" instead of "model" as we only store references to either high level parameters or CAD drawings without a constructed simulation model). The major functionality of the GUI is the ability to structure high level and low level designs. Whenever a user wishes to input a new design, a choice has to be made regarding the abstraction level of the design, as depicted in Figure 5.3.

The high level designs are added using customizable windows to input parameters. Such window requiring three parameters is shown in Figure 5.4. We can see that the required input parameters are specified by an input file. However, these parameters must also be relevant to the high level DEVS model that we will discuss in Section 5.3.1. Once the the parameters are specified, they will be stored internally in the *wxTree* widget.

The low level designs are added by selecting the appropriate XML (Extensible Markup Language) description of the design. The XML description is generated from an CAD drawing as described in Section 5.2. Once the XML description has been selected, the

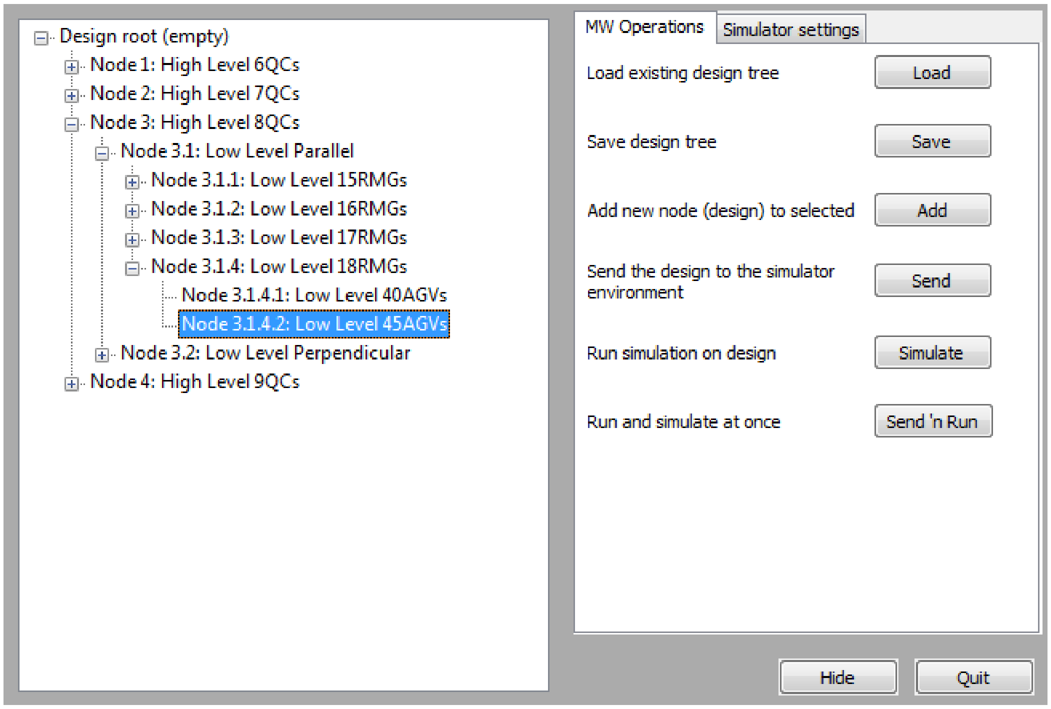


Figure 5.2: The "Multiple Worlds" graphical user interface.

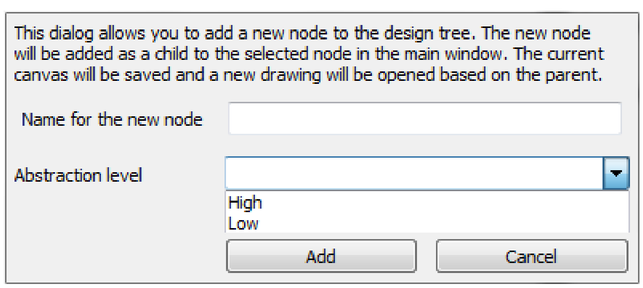
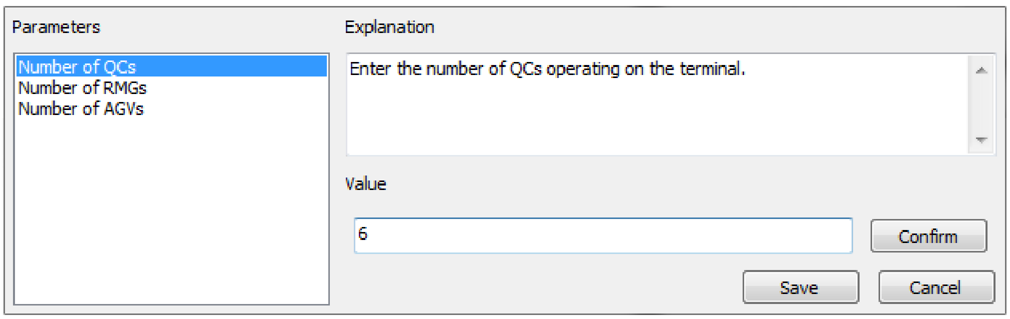
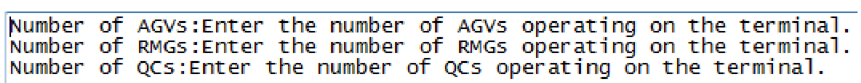


Figure 5.3: Adding a new design: choosing between a high level or a low level model.



(A)



```
Number of AGVs:Enter the number of AGVs operating on the terminal.
Number of RMGs:Enter the number of RMGs operating on the terminal.
Number of QCs:Enter the number of QCs operating on the terminal.
```

(B)

Figure 5.4: Adding a high level design: required parameters (A) based on an input file (B).

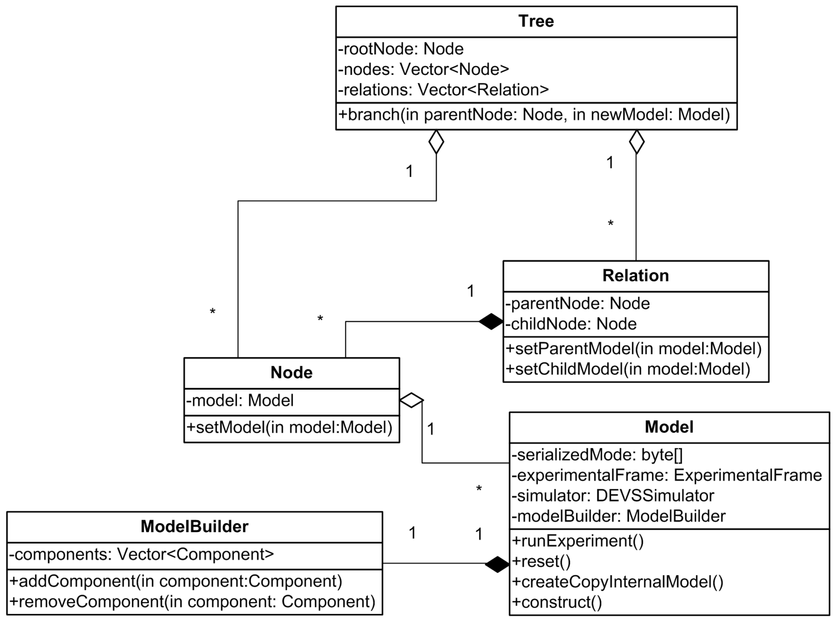


Figure 5.5: The architecture of the “Multiple Worlds” structure is designed to reflect the formalism presented in Section 4.4.1 (we omitted implementation specific details, such as support classes and methods, that are present the coupling with external classes, such as the TCP sockets).

node will contain a reference to that description, which will be ready to be sent to the “Multiple Worlds” internal structure.

The “Multiple Worlds” GUI interfaces with the internal structure by a TCP/IP network connection. Whenever a user wishes to run an experiment, the design (either high level or low level) is sent to the internal structure, which constructs the appropriate simulation model for it. We will discuss the construction of the simulation model in the following section. This decoupling between interface and internal structure allows for a distributed software implementation, with the possibility of running simulation experiment on dedicated computers.

5.1.2 The “Multiple Worlds” internal structure

The “Multiple Worlds” internal structure is responsible for maintaining a tree of simulation models, run the simulation experiments, and save the output in a file or send the output the external clients. The internal structure has an architecture that reflects the formalism presented in Section 4.4.1. This architecture is shown in Figure 5.5.

The classes *Tree*, *Node*, and *Relation* are present to construct the structural equivalent of the tree kept in the "Multiple Worlds" GUI. The class *Tree* keeps a list of nodes and relations between these nodes. To transverse the tree, a reference is kept to the root of the tree: starting from the root, every other node can be reached using the list of relations.

The class *Model* keeps the necessary references to construct a simulation model, run experiments on it, and reset the model to its initial state. The construction of the model is carried out using the class *ModelBuilder*: this class stores all components that needs to be added to construct a model that is consistent with the design that has been received from the user. Each model builder is customized to a component library, which in this case consists of simulation components to construct models for automated container terminals. This means that the model builder is made to construct the necessary couplings whenever a specific component is added to the simulation model. A more elaborate discussion on the model builder for the component library used in this instantiation, is available in Section 5.3.4.

Each model is serialized into a byte string to be able to rerun the model construction process and to carry out the simulation experiments. As soon as the complete list of components that needs to be added to the model, is available, the complete set of objects that constitutes the simulation model is serialized using the Java Object Serialization facility [Ora10]. The serialized model can also be used to store the complete tree into a persistent storage.

5.2 Designing using AutoCAD

CAD environments have long been the preferred environments into which new design drawings can be specified. We have opted to use AutoCAD to be able to specify detailed designs. This choice is based on observations during our case study: the design team at APM Terminals uses AutoCAD for their drawings.

The creation of 3D environments based on CAD drawings can be conducted in several ways. Whyte *et al.* [Why00] present three distinct ways: a library approach, a direct translation, and finally a database approach. The library approach uses a library of components that contains blocks that can be used in a 2D drawing and 3D models that represent the appropriate mapping. On the other hand, the simple translation purely transforms the CAD drawing to a 3D model. The CAD drawings have to be made directly in 3D which can be used directly in a visualization environment. Lastly, the database approach uses a central database with a description of an object from which a 3D model and a CAD drawing can be extracted. Although a direct translation (e.g. drawing directly in 3D) would provide major flexibility, it would require too much effort for the designers that usually concentrate their efforts on 2D schematized designs. On the other hand, the database approach is useful for situations that require a two way data exchange between the 3D virtual environment and the CAD environment. In our case, library approach would best fit our needs: it allows the re-use of existing components, which reduces time and efforts needed to construct the 3D environment. Additionally, this approach is also suitable for constructing a simulation model using a library of model components. Therefore, this approach allows us to construct one description of the container terminal design that can be used to both instantiate a visualization as well as a simulation model.

The predefined components are primarily the different cranes, vehicles and containers that are present in most automated container terminals of our interest. Additional components represent the buildings present on the terminals, such as the central administration offices and the workshops. These can be specified by using AutoCAD blocks that adhere to a naming convention (See Appendix D). This way, the conversion algorithm simply needs to look up specific names to decide which 3D object is required. Currently, the library contains objects for different functions and areas of a container terminal:

1. Sea transportation: different kind of vessels (such as the Emma Maersk) and barges.
2. Sea side equipment: quay cranes for vessels, barge cranes for barges.
3. Horizontal transportation: AGVs, automated shuttle carriers.
4. Stacking area: RMGs, AGV stacks.
5. Land transportation: trucks and trains.
6. Buildings: workshops for equipment, gates for trucks, administration building.
7. Miscellaneous objects: lighting poles, containers.

Further information about position and rotation of these objects can be extracted from the properties specified by the designer for each block.

The *ad hoc* specification possibility is present to visualize the surrounding areas, the areas inside the terminal and the various detailing such as road lining, rail tracks, and jersey barriers. The surrounding areas can be delineated as well, which will be converted to an appropriate set of vertices needed to visualize it in the 3D environment. In case of lining, the procedure is similar, different only in the use of textures and the use of standardized width values. Using jersey barriers, or other forms of 3D objects which placing follows some kind of line, is performed by calculating positions and orientations based on a specified line and the measurements of the chosen object. Currently, the following areas and detailing can be specified:

1. Surrounding areas outside of the container terminal premises, as well as the sea
2. Container terminal premises, which will default to a paved area
3. White or yellow lining for road lining, parking areas, etc
4. Rail tracks for trains, quay cranes and RMGs

5. Barriers such as jersey barriers

The AutoCAD plug-in has been created using ObjectARX [Aut10]. If loaded on a drawing that follows the conventions discussed earlier, it generates an XML file that can be used in the stand-alone visualization environment. Therefore, creating compatible drawings requires little extra effort (naming blocks according to specific conventions, loading the plug-in) compared to creating traditional CAD drawings of container terminals.

5.3 A library of simulation components

The scope of this research is restricted to automated container terminals: a selection of the large variety of material handling equipment can therefore be made. This selection assumes container terminals wherein quay cranes are active at the sea side, AGVs take care of the transporation between sea side and stacks, and between stacks and land side, and finally RMGs handle the containers in the stacks. In automated container terminals, the TOS controls the equipment and is therefore a crucial part for the performance of the terminal.

The discrete event models are defined in the DEVS formalism [Zei00]. In this formalism, two types of models are present, namely atomic models and coupled models. An atomic model has input and output ports to connect with other models and has various transition functions that define the state changes based on events. The state variables are model specific and information considered relevant by the modeler. Coupled models, on the other hand, are solely composed of interconnected atomic models or other coupled models. The input and output ports of coupled models are directly connected to the input or output ports of contained atomic or coupled models. Furthermore, the contained atomic and coupled models can also be connected with eachother by using their input and output ports. Coupled models do not contain state variables of their own. The DEVS formalism has been implemented in the D-SOL [Jac05] as D-SOL ES-DEVS[Sec09]. In this Java-library, every atomic or coupled model has to be derived from the appropriate classes. Figure 5.6 clarifies how a conceptual DEVS model relates to its implementation in D-SOL ES-DEVS by presenting a partial class diagram.

In Figure 5.7, a conceptual model of an automated container terminal has been schematized in a System Entity Structure (SES) [Zei07]. An SES is a specification of structural and specialization relations for a model family. The diagram shows the decomposition of an automated container terminal into different types of equipment and a TOS at different abstraction levels. Further, the TOS is functionally decomposed to control each type of equipment. Finally, each type of equipment is decomposed both on a physical as on a functional level of each type of equipment. The function of each component will be discussed in greater detail in Section 5.3.1 and Section 5.3.2.

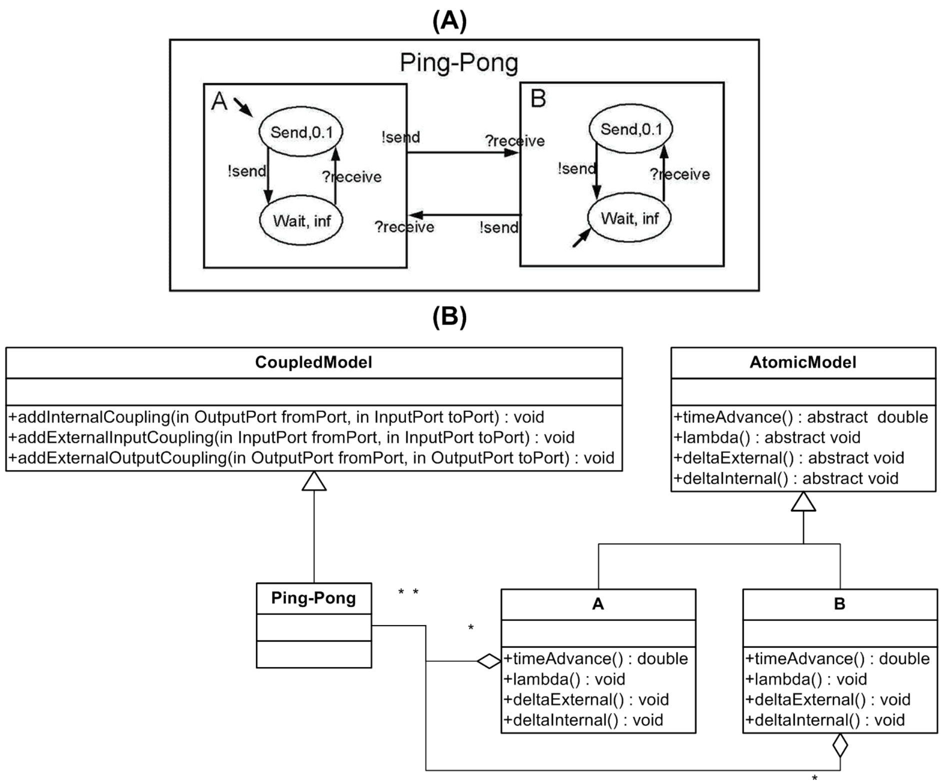


Figure 5.6: The Ping-Pong example model [Zei00] (A) implemented in the D-SOL ES-DEVS library (B).

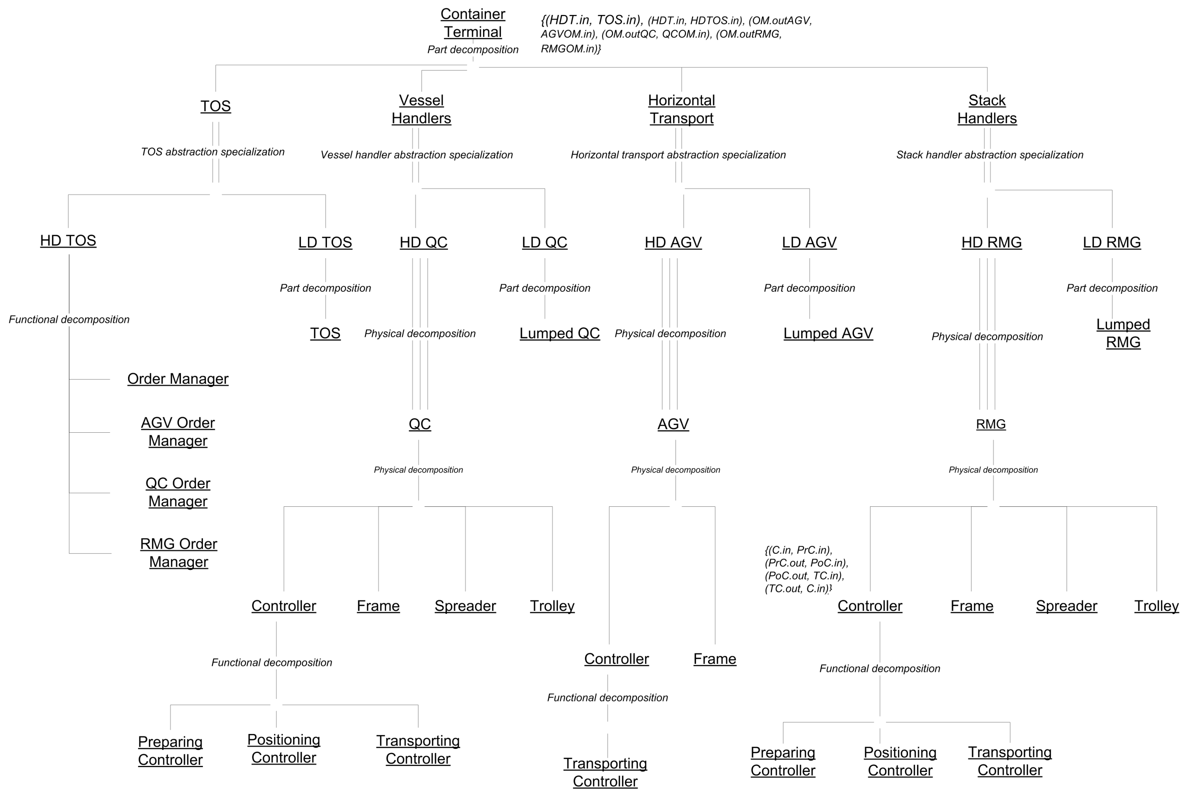


Figure 5.7: The SES of the conceptualized automated container terminal. For clarity reasons, only major couplings have been specified. (Abbreviations: QC for quay crane, AGV for automated guided vehicle, RMG for rail mounted gantry crane, HD for high definition, LD for low definition, TOS for terminal operating system)

In the following two sections, we will present the two component libraries, which we call the "high level" and the "low level" component libraries. If we reflect back on Figure 4.4, referring to the simulation study life cycle, we can show how the development of component libraries differs from a normal simulation study. We begin by arguing that the simulation study of a component library does not start from a problem (part one and two of the simulation study) but from a class of problems. In our case, we identified the need for component libraries to support the design process of automated container terminals. Hereafter, we implicitly answer the third step by proposing the use of the component library as a solution, and define the objectives, as part of the fourth step, generically for the class of problems: "given a set of requirements of the automated container terminals in terms of operations, costs, and environmental impact, we want to design an automated container terminal that satisfies the actors involved in the process". The conceptual and communicative models of each component library, part of step five and six, are presented as the SES in Figure 5.7 and the individual conceptual models of each entity of the SES. Each entity of the SES is programmed using D-SOL ES-DEVS, as part of step 7. Finally, experimental models are used to verify the simulation components. The results of these experimental models, step 8, are used to compare the "high level" models with the "low level" models. In our research, the "high level" models are developed in terms of the ''low level" models to reflect the design process presented in Figure 4.2.

5.3.1 DEVS Specification of the high level model components

The components specified at a high abstraction level provide the ability to define different areas of the terminal given parameters such as the number of specific type of equipment. The components represent complete areas of the terminal such that, by using the components, we can specify a complete automated container terminal by denoting each area. In Figure 5.7, the components specified at a high abstraction level are the components of lumped equipment. The UML diagram of this set of components is shown in Figure 5.8.

The components of lumped equipment represent a DEVS atomic model with the type and number of equipment as parameter. Figure 5.9 shows the lumped quay crane, the models for other types of equipment are similar, only with differing parameters. Lumped equipment can have external events while being in an idle phase or an active phase. If the model is in the idle phase, no specific equipment is active. In the other case, the model is in the active phase, and one or more pieces of equipment are active. To define how long the model should stay in the active phase, we keep a list of equipment with the times they need to process the active order. The times are stochastic and taken from a distribution that belongs to this type of equipment. By picking the minimal time from this list, the phase can come to an end and release that specific equipment. At that point, the model can either go into the idle phase if no equipment needs to perform any action, or again to the active phase if other equipments have to continue. The latter means the model should stay in the active phase for a specific amount of time. To calculate the specific amount of

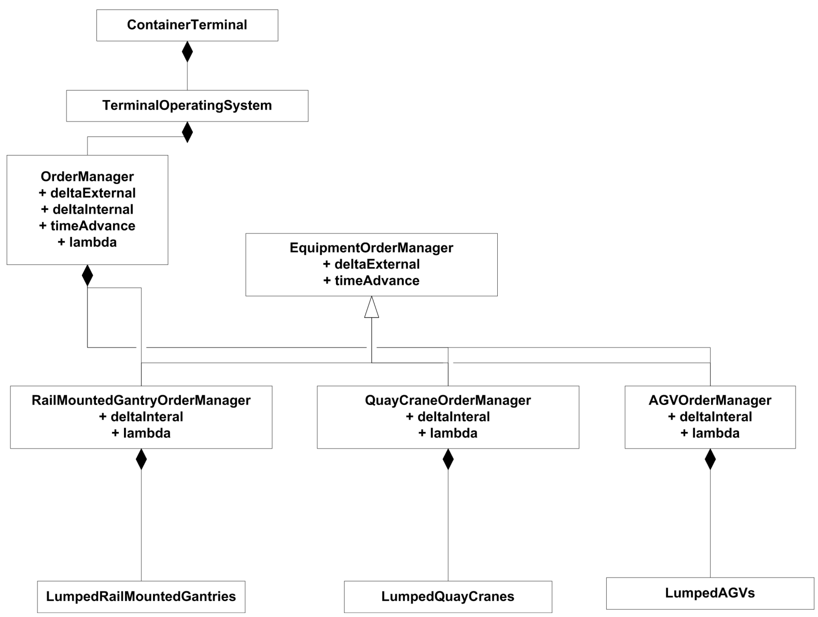


Figure 5.8: The UML class diagram of the set of high level components. Every class is derived from either an atomic model or a coupled model. The latter can be recognized by the absence of methods. The components reflect the LD components illustrated in Figure 5.7

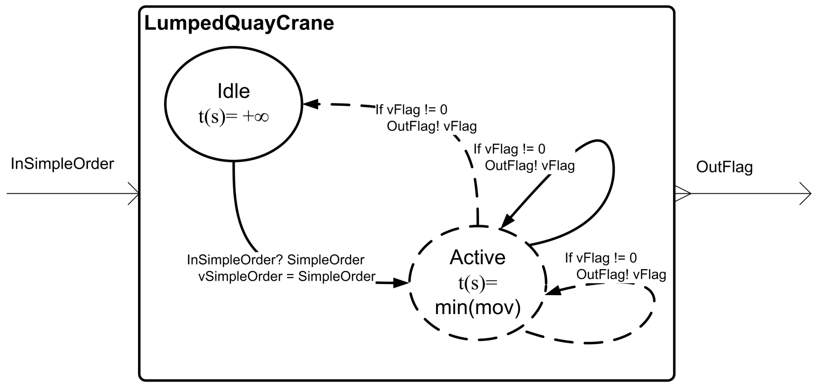


Figure 5.9: The lumped quay crane: an atomic model representing all the quay cranes of a terminal. The equipment with the minimal amount of time needed to complete an action is taken as $min(mov)$ to define the time for that phase.

time, we need to find the new minimum time for the list of equipment and subtract the amount of time the phase has been already active. In pseudo-code, the following happens at every internal event:

Require: $EquipmentTimesList, minTime$
  $remove(EquipmentTimesList, minTime)$
  if $!EquipmentTimesList.isEmpty$ **then**
    for all e in $EquipmentTimesList$ **do**
      $e \leftarrow e - minTime$
    end for
    $minTime \leftarrow min(EquipmentTimesList)$
    $phase.time \leftarrow minTime$
  end if

Figure 5.10 shows the overall structure of a high abstracted model. The TOS used in this model is further explained in Section 5.3.2. The lumped models for RMGs and AGVs have their inner workings similar to the lumped model for the QCs, with the major difference being the distribution from which the stochastic values are taken.

5.3.2 DEVS Specification of the low level model components

There is a direct mapping of the entities identified in the SES on the low level model components expressed in the DEVS formalism, these are modeled as either atomic or coupled DEVS-models. To implement the lowest abstraction level we have to prune the tree to keep the high definition entities. We will first start by presenting the overall model and continue by discussing the detailed models. The overall structure of the model is

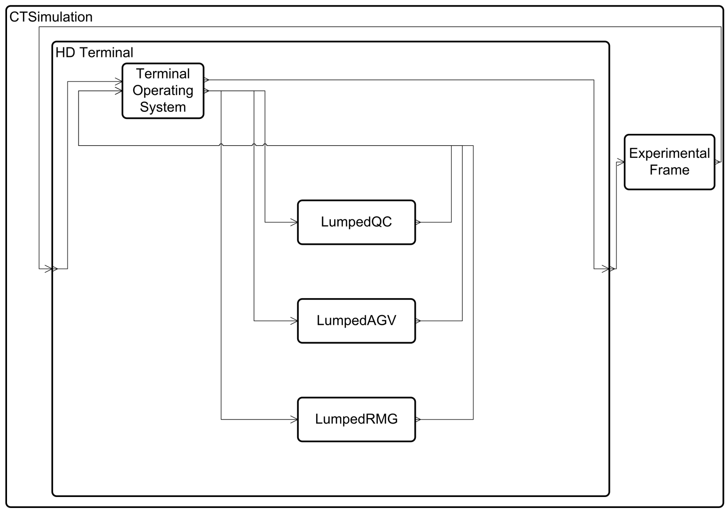


Figure 5.10: The structure of the high level model. The coupled models are shown with their in- and outputports.

presented in Figure 5.11: this figure shows the coupled models. The overarching coupled model contains the model for the actual container terminal and the experimental frame. The model of the container terminal holds the model of the TOS which sends orders to the models of the different equipments.

Both TOS and the equipment models are coupled models. Although this diagram shows only a subset of possible equipment, other model for equipments can be added as well. This can for instance be done for barge cranes, terminal trucks, etc. We will later on see how the TOS takes care of handling new equipment and how the operating logic is abstracted out of the model itself.

Similar to Saanen’s approach [Saa04], the individual components can be grouped into managerial, controlling and physical models. The managerial models are part of the TOS and manage the orders. The controllers receive the orders and translate these orders into specific movement commands that are sent to the physical equipment, that actually perform the commands. As the models pertaining to each group share the same behavior, we will present each group with the models that manage, control and represent the quay cranes.

The TOS coupled model

The TOS contains a general order manager and equipment-specific order managers, as shown in Figure 5.3.2. The order manager receives generic orders (“take the container from place X and put in on place Y”), which we call complex orders, and sending it through to the assigned equipment to perform the specific order, which we call simple

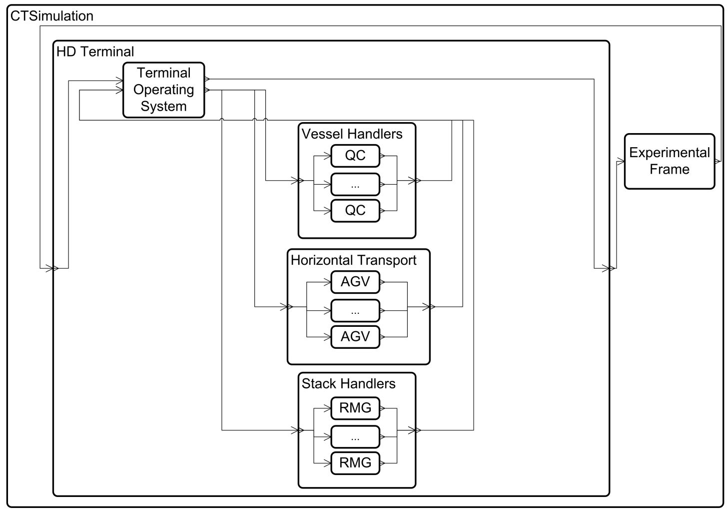


Figure 5.11: The structure of the low level model. The coupled models are shown with their in- and outputports.

orders. The order manager of specific equipment is modeled as a DEVS atomic model that has two phases: an idle and a processing phase. When the model is in phase idle and receives an event, it makes a transition to phase processing, sends a flag out to the relevant port, and makes an internal transition back to phase idle. Flags are received through the input ports connected to the controlled equipment. The order manager of the quay crane is schematized in Figure 5.3.2.

The equipment coupled models

The order goes from the TOS to an equipment. A type of equipment is decomposed into controlling and mechanical models. The behavioral models are called controllers, whereas the physical models depend on the different physical parts of the equipment. The controllers of equipment receive the order and decompose this to control the different physical models. This happens according to different movements that the equipment has to do in order to fulfill the order. To model this, the choice has been made to decompose the controller into subcontrollers that are responsible for the different movements. For a quay crane, this can be a preparing movement ("put the crane into a predefined position which makes it able to freely move"), positioning movement ("put the crane to the exact place where the container has to be picked up"), and finally the transporting movement ("pick the container and transport it to the final destination"). As shown in Figure 5.3.2 (the transporting controller has been taken as an example) , the controllers are fairly simple, having the sole responsibility of sending commands to the physical models, which happens

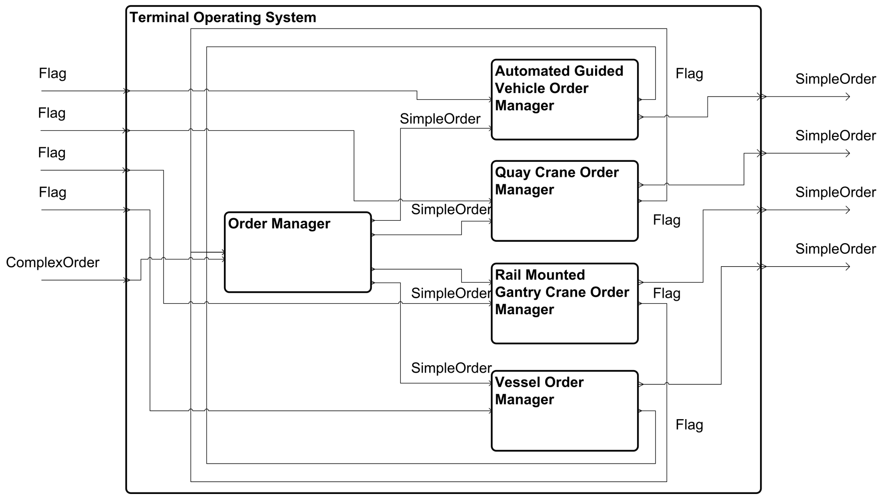


Figure 5.12: The model of the TOS with a general order manager that sends order to the equipment-specific order managers.

in the output function. It has to be noted that the decomposition of the main controller serves as a reference model, but does not imply a restriction for future uses. The controllers can easily be changed by other controllers with different behavior.

The physical models are in charge of performing the physical activities, which is mostly changing positions. In case of a quay crane, the physical models are a moving frame, a spreader and a trolley. As an example, the moving frame is schematized in Figure 5.3.2.

It is interesting to note how this model differs from a more traditional way of modeling moving equipment, which mainly happens in discrete time fashion. In this model, a continuous movement is being discretized into just five distinct phases instead of a large amount of time-segments. Figure 5.3.2 helps clarifying the advantages of such an approach: in discrete time, we have to calculate the new position and speed at every single time step whereas in discrete event, the calculations can be performed three times: one time at the beginning of every phase relevant to the state. The calculations of the acceleration, cruise and deceleration times (which are presented in Figure 5.3.2 as function f) are calculated as follows:

Require: $CurrentSpeed, Rate, Distance$
  $QuadraticSolution < T >$
  $\leftarrow SolveQuadraticEquation(Rate * T^2 - InitialSpeed * T + Distance)$
  $MovementTime \leftarrow Max(QuadraticSolution < T >)$
  return $MovementTime$

These types of calculations are made for each moving equipment and for each subpart of an equipment (for instance a trolley that goes for- and backwards). This saves a large number

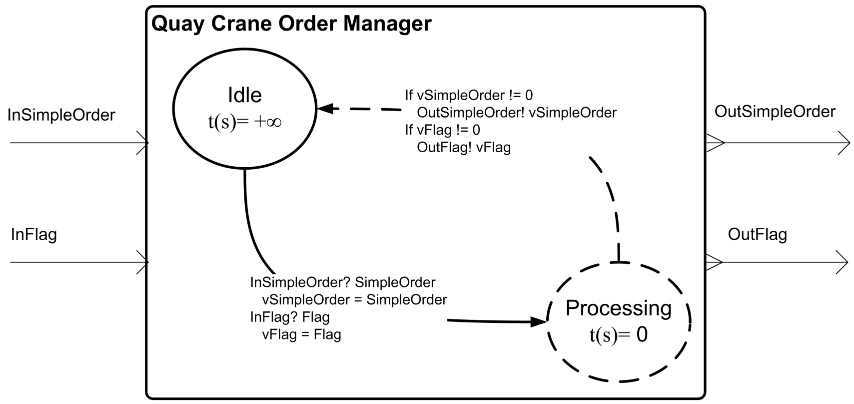


Figure 5.13: The model of the order manager of the quay crane.

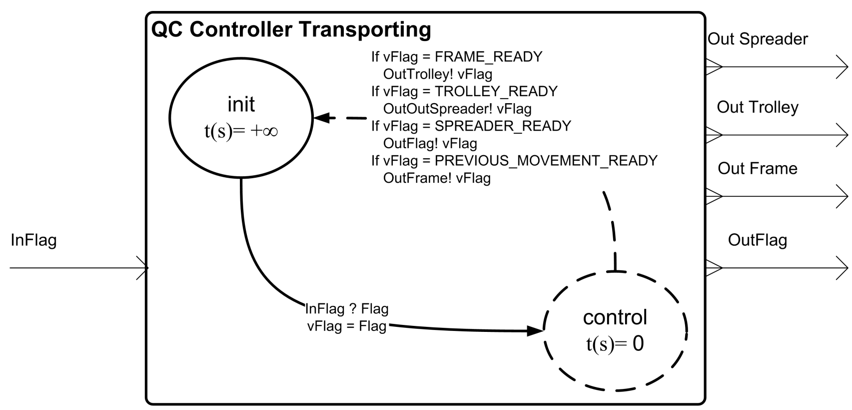


Figure 5.14: The model of a transporting controller is shown here as a reference implementation to all equipment controllers.

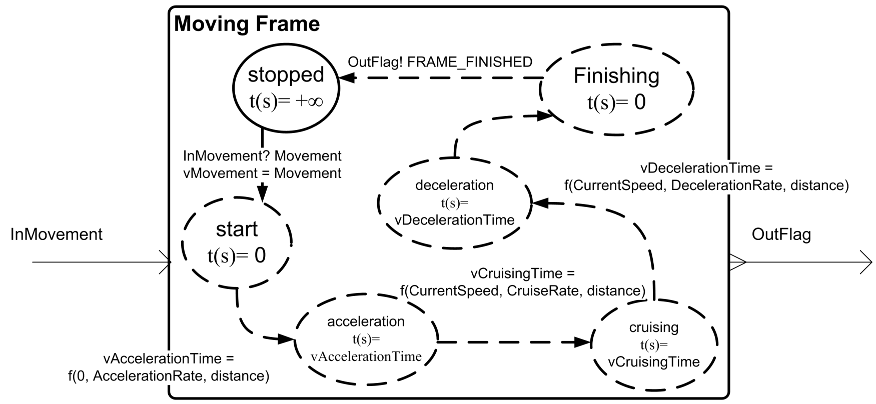


Figure 5.15: The model of a moving frame is shown here as a reference implementation to all the physical models. (f is the function to calculate times based on speed, rate and distance)

of discrete time calculations, which in large numbers can be computationally expensive, that can result in a quicker execution of the model without loss in precision.

5.3.3 Component transformation

The "Multiple Worlds" formalism requires a "component transformation function" between the defined sets of components. In our instantiation, this means that a relation should be defined between the high level models and the low level models.

The following function can be defined, taking into account the nomenclature used in Figure 5.7:

$$\zeta(\eta(c_{HD}, (n))) = n\eta(c_{LD}) \text{ with } n \in \mathbb{N}$$

The *LumpedQC*, *LumpedRMG*, and *LumpedAGV* take the number of equipment as the main input variable. This variable defines the capacity of that subsystem. When applying the component transformation function, we want to go from the lumped models to the detailed models. This means that the parameters taken earlier will be translated by adding as many components of the same type of equipment as defined by the parameter mentioned earlier. The transformation therefore makes sure that the parameterization of the lumped model will impact the structure of the detailed models.

The instantiation of the model, denoted as η in the above function, will be discussed as part of the model building process discussed in the following section.

During the component transformation process, it is important that the results of each node in the "Multiple Worlds" tree leads you as shown in Figure 4.2. As mentioned before, we achieved this by deriving the high level models from the low level models described

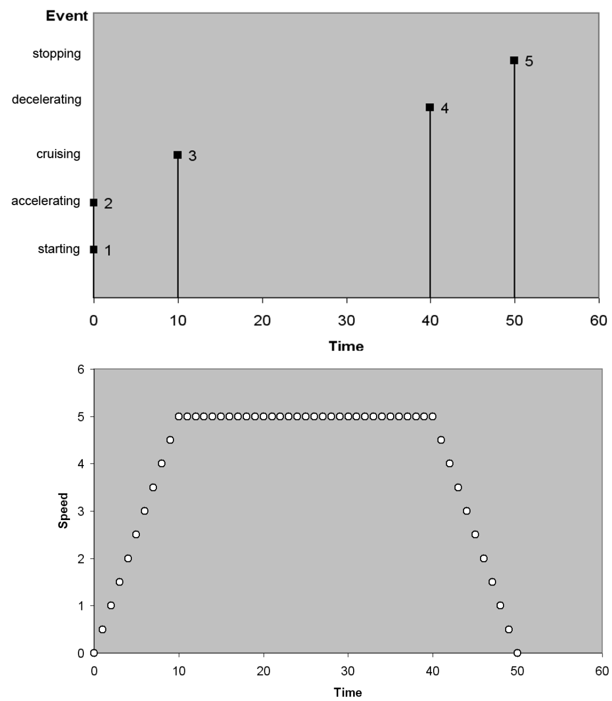


Figure 5.16: The difference between time based and event based modeling: the model of a moving vehicle can be done in fewer calculations.

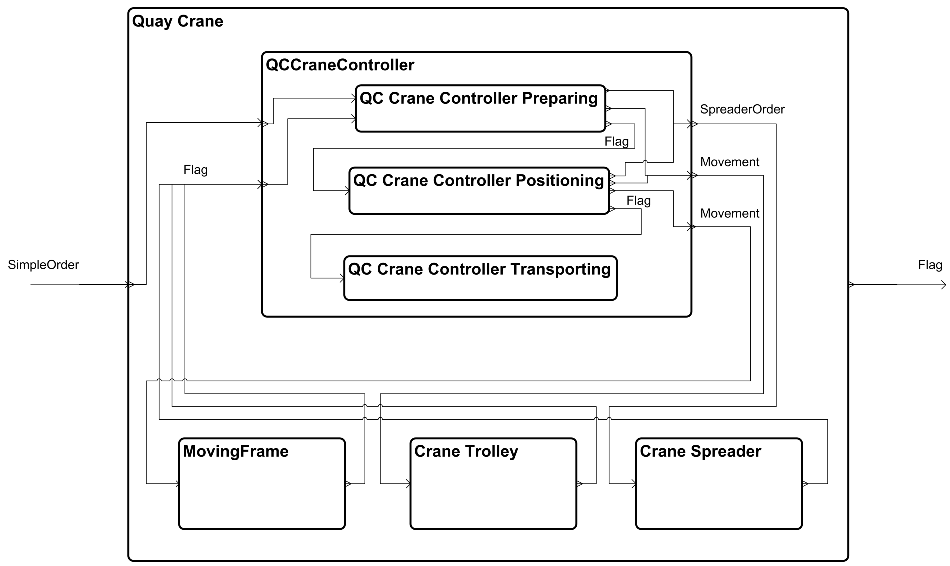


Figure 5.17: The model of a quay crane contains a controller, a movingframe, a trolley, and a spreader.

earlier. Therefore, the results from the high level models present a less precise indication of the performance of a terminal compared to the results of the low level models. Using the low level components, decisions have to be taken on more detailed aspects of the terminals: this allows for models that are far more detailed in providing results.

5.3.4 Model building process

A dynamic build-up mechanism of the container terminal model is able to add and remove equipment from the model and construct and remove the couplings between the TOS and the equipment. The container terminal model does this by receiving events that define the addition and removal of equipment. This results in the creation and destruction of the model instance of the equipment and in the creation and destruction of the coupling from this new model instance to the rest of the container terminal model. This mechanism corresponds to variable structure as described by Hu *et al.* [Hu05b].

We can use the running example of quay cranes to explain this mechanism. The container terminal receives an event to create a new quay crane. The creation of the quay crane model is performed based on the received parameters. This model is directly connected to the TOS which on its turn will secure the connection with the right order manager. In a similar way, the removal of the quay crane takes place, by having an event to remove a certain quay crane which is present in the container terminal.

Although this mechanism will mostly be used in the initial creation of a container terminal, thus sending events at simulation time 0, other possible scenarios can be thought of. During a long term simulation of a container terminal, a decision could be made to add a number of equipments after a couple of (simulated) years. With a static model, this would be a difficult task. However, with the mechanism presented here, an event can be sent for the creation for which the container terminal will just continue to work with the extra equipment.

5.3.5 Java implementation

The DEVS model presented here, has been implemented in D-SOL-DEVS. Although a one-to-one mapping of the DEVS conceptual model in a class model would have been a possibility, careful consideration is put into exploiting the object oriented environment provided by Java. Abstraction, as found in object oriented programming, is a powerful mechanism that can be exploited in the implementation of DEVS models without conflicting with the formalism. As many atomic models can have some communinalities (for instance a common external transition function), reimplementing the same function multiple times, would become cumbersome and error-prone.

Throughout the discussion of the DEVS model, there was a clear grouping of similar models: managers, controllers and physical equipment all share some similitarities. These groupings can give use the possibilities to abstract different functions on the managers, controllers and physical models. In the class diagram presented in Figure 5.3.5 these abstractions are presented through the inheritance relationship. Each ordermanager inherits the deltaExternal method (External transition function) and the timeAdvance function from the abstract class *EquipmentOrderManager*. This is due to the basic behavior that has been presented in Figure 5.3.2. In a similar way, the controllers (e.g. *RailMountedGantryControllingPreparing* and *AutomatedGuidedVehicleController*) inherit their behavior from abstract controllers *CraneControllingPreparing* and *VehicleController* respectively. Finally, equipments sharing same controllers and physical models, can be abstracted by using the abstract classes of the containing models.

The Terminal Operating System

In an automated container terminal, the TOS runs the main algorithms to control the equipment. In the model presented here, the TOS contains an order manager and the various equipment-specific order managers. The order manager keeps a list of orders and always picks the order first in the list. This list is ordered according to an instance specific algorithm which allows the user of the component library to implement a specific order allocation algorithm by re-implementing the ordering method. The same approach was chosen for the equipment allocation, by keeping a list of equipment which will be ordered according to a specific algorithm.

The task of the order manager is to take complex orders and transform them into multiple simple orders that can be sent to the equipment managers that will handle the orders. To make the transformation from complex order to simple order possibile, the order manager has access to a class that implements the *TOSLogic* interface. This makes it possible to assess different types of algorithms (e.g. scheduling of equipment, stacking of containers, etc) without changing anything of the container terminal model.

The *TOSLogic* interface contains an enumerator of order types, a function to create simple orders, one to create order stacks and finally two functions to check whether an order is compatible with a selected instance of equipment. The most important function is the one responsible of creating a simple order. To achieve this, the implementation of the interface can access the list of available (and unavailable) equipment, and all current orders. The interface supposes also an implementation of a datastructure reflecting the layout of the container terminal. In the reference implementation, a graph has been used where each node restricts access to certain type of equipment so that the paths of equipment can be deduced.

5.4 Visualization in a 3D virtual environment

Most modern simulation packages for material flow and production processes contain an animation component to support the communication of output data to the users i.e. for verification and validation of the simulation models, and presentation to external parties such as customers [Roh00]. Traditionally, 2D canvasses have been used to animate the icons representing the different entities in the simulation. In recent years however, a shift has been made from 2D canvasses to more appealing 3D renderings. Although these 3D renderings are mostly meant for marketing purposes, studies have shown that 3D renderings can also increase insight into the dynamics of the simulation model, thus increasing the support for understanding the output data and validating the simulation models [Akp05a, Akp05b].

Using animation to visualize the output of discrete event simulation is particularly challenging due to their inherent difference in the way time is managed. Discrete event simulation models’ states are described in terms of the consequences of events [Nan81]. The flow of the simulation clock would therefore perform jumps of variable sizes, depending from one event to the next, and processing the event at hand. Animation techniques, on the other hand, aim at presenting a continuous flow of events, discretized in small time steps only limited by computational capabilities. The flow of the animation clock therefore mimics the wall clock, in certain cases only differing in a scaling factor to speed up or slow down the animation. While simulators aim at performing a simulation run as fast as possible using a logical clock, animations are meant at running at a speed desired by the viewer or user, with as little jerky movements as possible. This inherent difference causes conflicts that leads to synchronization issues between simulator and animation, and

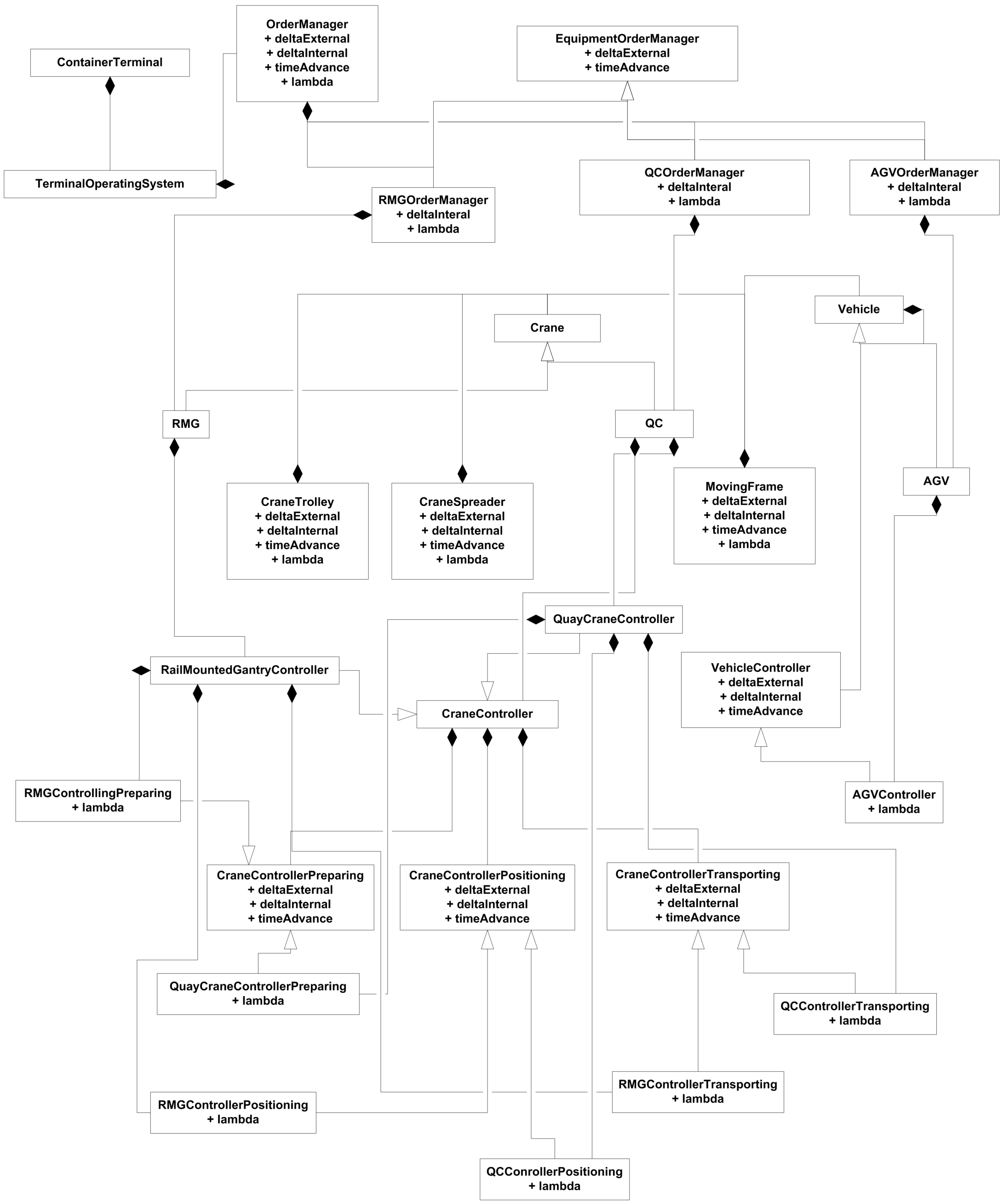


Figure 5.18: The UML class diagram of the DEVS model. Every class is derived from either an atomic model or a coupled model. The latter can be recognized by the absence of methods. The components reflect the HD components illustrated in Figure 5.7

unwanted slowdowns or speedups of the animation.

Traditional approaches for implementing animation components of simulation packages result in tightly coupled environments. Both parts must be aware of specifications of the other part in order to work together in harmony. To achieve this, simulation modelers must often be aware of the visualization part and the other way around: animation designers must cooperate closely with modelers. This results in models containing specific references to the animation, such as references to animation classes in Java libraries [Jac02], specialized procedures [Cap06], or fully integrated simulation and animation objects [Peg07]. This tight coupling can however have multiple disadvantages for flexibility and clarity, clearly lacking separation of concerns. Tightly coupled simulation environments are difficult to adapt to new visualization engines: as visualization engines are rapidly improving, having the flexibility to use a new generation engine can often be interesting. In tightly coupled environments, models contain often too much information meant for the animation, which makes complicated models even harder to understand.

To construct this coupling concessions have to be made to the quality of either the animation (e.g. non-fluent animation) or to the quality of the simulation model (e.g. entangled simulation model and animation code). In most cases, the result that has to be presented (end result) to the final users (i.e. clients) has priority, so that simulation modelers end up by fine-tuning their model code to have an acceptable animation. In COTS packages such as Rockwell Arena [Roc10], and Applied AutoMod [App10] this coupling has already been made by providing existing blocks to users. We, on the other hand, focus on the developers of these building blocks.

Prioritizing the end result leads to inflexible environments and models that lack clarity. Models tend mostly to be big (in terms of lines of code) and complicated: the extra code needed for animation can be considered as a burden. The extra code is often also written with a specific animation component in mind, making the complete environment inflexible, bound to be used with that specific animation component. This is not desired whenever new, more advanced animation components are available. It would therefore be desirable to have a modeling environment where no attention has to be spent on the animation and where new animation components can be easily added. Nonetheless, it would be naïve to think that we could have a simulation environment that understands what the modeler is implementing and creates appropriate animations on-the-fly (although that should be a future we should aim for). However, a clear demarcation of simulation model and animation code can already be considered a step towards both flexibility and clarity.

5.4.1 Implementation

In traditional DEVS models, the flow of information goes solely through the ports, including data for statistics gathering and animation. In approaches such as discussed by Wainer and Liu [Wai09], a trace file is generated by outputting data at points of interest. To fulfill our first requirement however, we would want a clear demarcation of model and animation code, preferably having no reference to animation components. This avoids entangled model and animation code that makes the whole more difficult to understand and maintain. To attain this, we have constructed the model probe: an approach that exploits Java's reflection capabilities and the publish-subscribe mechanism.

The atomic model and coupled model classes are both derived from the Abstract-DEVSModel-class that exploits the Java's reflection mechanism to collect every variable that is part of a given instantiation of a class. The state can be deduced by taking the subset of the variables contained by this object. This subset must be taken as the object also contains variables that are useful for the internal logic of the DEVS implementation. Careful consideration is therefore needed to differentiate state variables from variables inherited from parent classes. This problem can however be tackled by taking the complete collection of variables and subtracting variables inherited from parent classes, those that therefore are not part of the state variables of the model.

Once the set of state variables has been determined, events can be fired that inform subscribed objects at each state change. To publish the state, different implementations are needed for atomic models and coupled models. Atomics models need only to publish their own state, thus firing the local set of state variables. Coupled models, on the other hand, have a state that is contained in the atomic and coupled models. Due to the hierarchical structure, this can be achieved by having a recursive function.

The animation component, which needs to be coupled to the simulation environment, is able to receive event changes by subscribing to the appropriate event generator. Once this has been done, the choice remains on what to do with these events. Although we have chosen to use the event changes for animation, one can choose to run statistical analyses with the data, or other undetermined purposes. In either case, we can define two ways of using the data: inside or outside the simulation environment.

Simulation environment with an embedded animation component are mostly compiled into a single executable. This practically means in our case implementing the animation component in Java. To achieve this, a class can be specified that receives the events and displays them in whatever way is desired to the user. This type of animation component can be useful for quick and easy ways of presenting the behavior of the model, such as in the debugging and verification stage.

Outside animation components grant much more flexibility for the animation component as it is completely decoupled for the simulation environment. To achieve this, a straightforward solution would be a network socket connection. In the simulation environment, an internet component has to be defined that receives the state changes and transforms them using a communication protocol. At the other end of the network connection, the same protocol will therefore be used to interpret the messages for animation purposes. The implementation of the animation can herewith be real-time or post-run, depending on the requirements for the animation. In either case, this fulfills our second requirement to be able to animate discrete event simulation. The third requirement is finally fulfilled by defining the intermediate language between simulation and animation

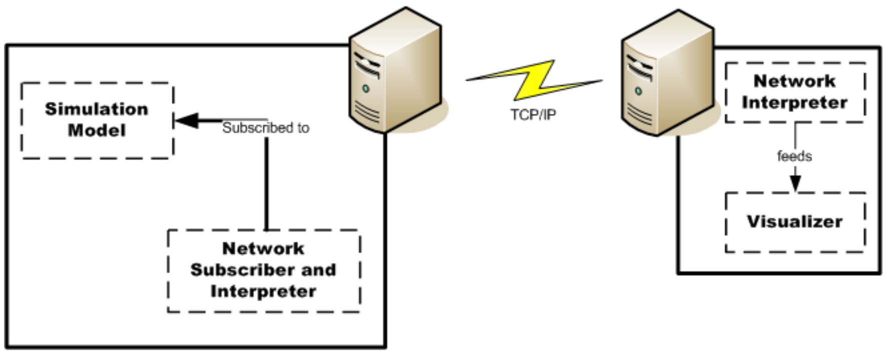


Figure 5.19: A diagram of the loosely coupled visualization architecture.

component that is used to communicate the state changes. This allows us to define new animation components as long as they can interpret the intermediate language.

In Figure 5.19 we can see how a typical setup can look like with animation components outside the simulation environment. The simulation model can run on a single server with a subscribed object that creates messages to send over a network. The visualizers, or other interested data collectors, can run on various different computers and connect to the server through a network. This loosely coupled architecture can be interesting whenever the simulation model or visualization component is computationally heavy.

To use the data from the model probe for animation, we have chosen to implement a post-run animation. This is described by Strassburger *et al.* [Str05] and basically functions as a recorder of events that are used for animation in a later stage. Again by Strassburger *et al.* [Str05] however, mechanisms have been defined to implement temporally parallel coupling between simulation and animation with variable buffer size. These mechanisms could be implemented in our case as well. The rationale behind the choice for a post-run animation is given by the advantages this approach brings to the users. By having a recording of a complete simulation run, users can navigate easily through the whole run. Fast forward, skip and repeat become easily manageable operations that possibly support users in the process of understanding the simulation output. However not a strict requirement, simulators should in this case be able to complete a run in a relatively short amount of time.

The animation component is loosely coupled with the simulation component through a TCP/IP connection. It receives messages containing the state changes of the simulation run. The animation component has therefore to construct the animation from these state changes. This can for instance be achieved through linear interpolation, or any other higher degree function that can describe the trajectory, based on the context of the animation. This allows us to animate cranes and vehicles, and to change the positions of containers. As all states are saved, it is also possible to skip to a specific instance of the simulation run as well as repeating passed events.

5.5 Statistical output

Output measures of simulation models are traditionally presented in terms of KPIs. Saanen [Saa04] discusses an extensive list of KPIs that are of importance during the design process of automated container terminals. These KPIs mainly focus on the operational aspect of a container terminal. An important part of many KPIs is the one indicating the number of containers using the unit TEU. TEU is mostly discussed to express KPIs such as the productivity in a year time (How many TEU per year?), the storage capacity, and the peak capacity. Due to the importance of the quay on the productivity of the overall container terminal, the number of productive moves per quay crane is measured as well.

In our software implementation, we have pursued the possibility of supporting multiple actors. The 3D visualization environment presents a container terminal design using an understandable communication medium. This is geared towards a multitude of actors with several, differing backgrounds. In case of presenting KPIs, this becomes more challenging as the perspective of each actor is expressed in KPIs that are unique for that actor.

To tackle this problem, we have chosen to use a raw trace output of the simulation model to calculate KPIs depending on the actor's demands. This output, which is also used to show the animation discussed earlier, only contains basic operational events that took place during the simulation runs. Using this output and the initial design, an output can generated customized to the needs of the actors identified in Chapter 2. This can be achieved as follows for the different actors:

The financial actor in mainly interested in figures such as CAPEX, OPEX, revenue, and cash flow [Hu08]. The CAPEX can be calculated based on the design specification. The OPEX can be calculated on the operational output transformed using costs of operations (e.g. energy and labor). Finally, revenue and cash flow can be calculated using both the design specification and operational output.

The operations actor is mainly interested in figures concerning the operational performance of the terminal. These KPIs are extensively discussed by Saanen [Saa04] and can be calculated using the operational output of the simulation model.

The environmental actor is mainly interested in the environmental footprint of a container terminal. This is highly dependent on the operations of a container terminal and the design specification. CO_2 emissions can be calculated based on the type of equipment and the operational performance [Gee11].

Although we have restricted the transform the output towards the perspectives of the above actors, the approach can be used to serve the needs of other actors as well.

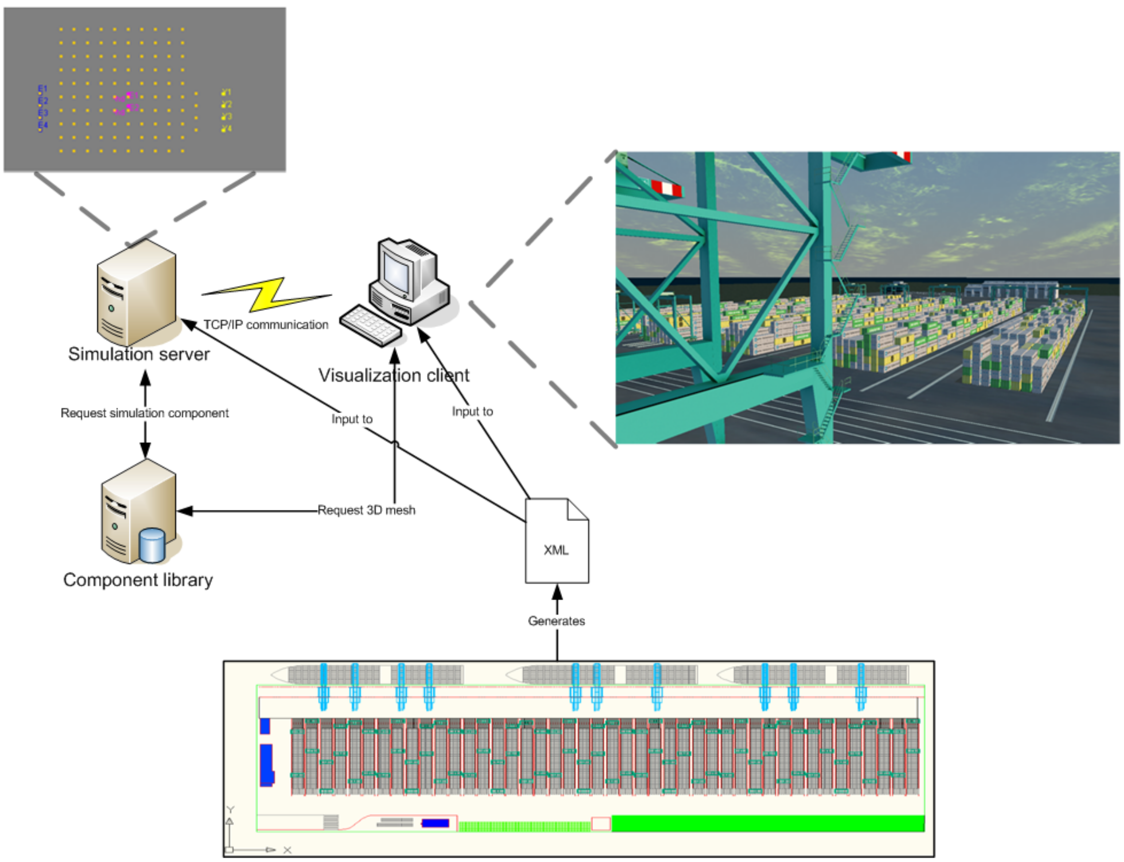


Figure 5.20: The schematized view of the architecture: we generate an intermediate XML from a CAD drawing; the XML file is used as input for the simulation server and the visualization client; the simulation server and visualization client request the needed components from the component library; the simulation server has a build-in 2D Java canvas visualization; the visualization client has a 3D virtual environment with animation.

5.6 The design process using the supporting software implementation

The software implementation presented so far is part of the instantiation of the Multple Worlds design method. The software implementation itself provides the technical infrastructure needed to design a container terminal using our design method. To complete our instantiation of the method, we need to describe the design process itself: how can actors use the environment to design an automated container terminal?

We will describe the design process following the flow chart presented in Figure 5.21 based on Figure 4.3. The process starts with the specification of the design requirements and the collection of documentation regarding equipment, terminal plot, demand forecasts, etc. These documents define the investment size, the physical possibilities, and the required throughput. Once this information is collected, designers can identify the list of criteria. The list of criteria is needed to compare two alternative designs in a concise way: using the criteria, a multi-criteria decision analysis can be carried out at regular intervals during the design process.

Once the criteria have been selected, the first high level design can be constructed. At this level, constructing a high number of designs is encouraged: the number of parameters is relatively low and the time to run the simulation model is short. In our instantiation, the parameters are limited to the number of quay cranes, RMGs, and AGVs. Limiting the design decisions to the sea side and yard allowed for a manageable simulation model in terms of model construction time while extensive enough to carry out experiments for evaluation purposes.

Using the results of the high level models, designers can choose to design using the low level models and go into more detail for the analyses. The low level models can be used to assess the layout of the terminal and the types of equipment used on the terminal. The latter can differ in costs, performance and environmental footprint. The number, types, and position of quay cranes, RMGs, and AGVs are deciding in several individual levels. This allows the designers to concentrate on one type of equipment at a time. The tree structure used in the design method provides the possibility to go back to higher levels in order to alter decisions on the different levels and discover the interactions between the different subsystems. After each level, designers can evaluate their designs using a multi-criteria decision analysis as mentioned before. Once the actors have extensively assessed different design options based on their individual and collective preferences, they can work towards a set of final designs which can conclude the design process.

5.7 A design environment for logistics systems

Throughout the chapter, we have discussed the parts of this instantiation that might be considered non-specific to automated container terminals. In reflection on the method presented in Chapter 4, we might want to consider the parts of the instantiation that are applicable to logistics systems instead of specifically to automated container terminals.

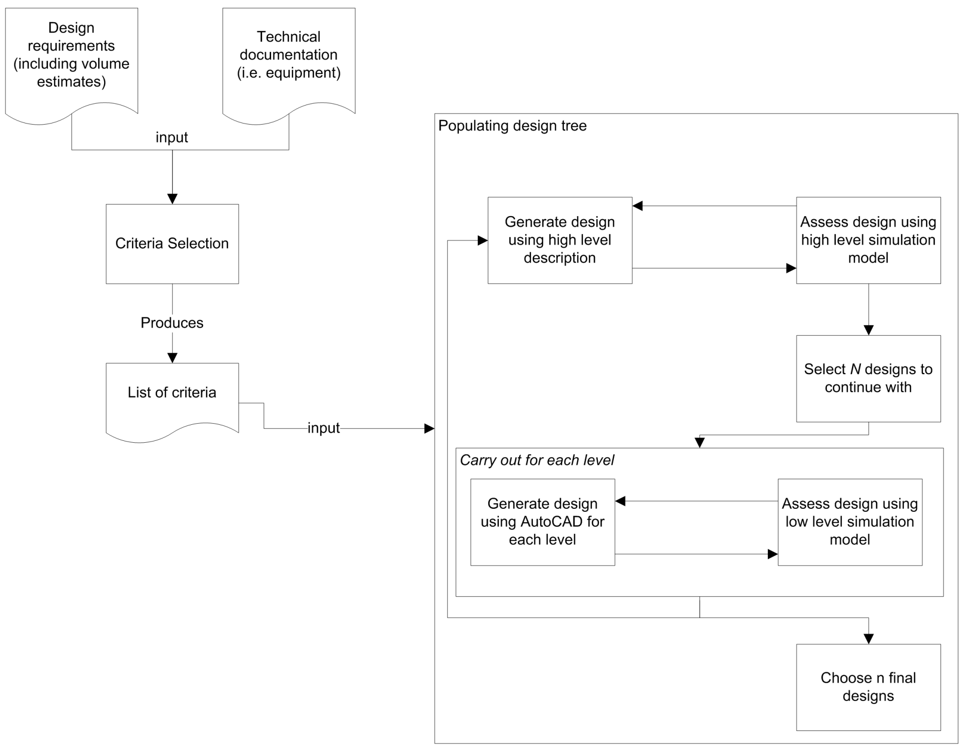


Figure 5.21: Similarly to Figure 4.3, this flow chart shows the design process using the multiple worlds method. Taken the design requirements and technical documents for automated container terminals, actors define the criteria they are going to use to assess a design. Once the list is ready, the actors enter an iterative cycle of generating alternatives and assessing these alternatives using the criteria. Going from high level models to low level models, they refine their design until a (set of) final design(s) has been selected to conclude the design process.

The architecture presented in Figure 5.1 illustrates the components that have been defined to construct an instantiation of the "Multiple Worlds" method and also illustrates how these components have been implemented to instantiate the method for supporting the design process of automated container terminals. To reflect on this, we need to divide this discussion in two parts: the software environment and the design process.

The software environment consists of software components that support the use of the "Multiple Worlds" design method. Apart from the input and output controllers, the software environment has been designed without taking into consideration the specific case of automated container terminals. The implementation of the structured tree to manage and store several designs is usable for other domains using the same software. Although the library of simulation components is specific to automated container terminals, its interface with the overall environment is not. The library communicates using an interface that defines components and its parameters. Likewise, the interface that is used to ascertain the communication between the two controllers (input and output) with the core of the environment is designed as using an ontology that has been discussed earlier. In the specific case of automated container terminals, the choice has been made to use AutoCAD as an input controller and a 3D visualization environment as an output controller. As such, these parts of the environment can be considered specific to this instantiation.

In terms of the design process, it follows the method presented in Chapter 4. The process has been instantiated based on automated container terminals. The instantiation requires defining the decisions that need to be taken during the process and the order in which these decisions can be taken. Careful consideration of the different steps in the process (defined by the several set of components used throughout the process) is required.

5.8 Reflection on the instantiation of the method

In this chapter, we have presented an instantiation of the design method presented in Chapter 4. The method has been instantiated to support the design of automated container terminals, which is a specific logistics system. Although close consideration has been put into implementing the method by strictly following the specification of Chapter 4, instantiation specific choices can be identified. These choices can be attributed to pragmatic reasons: constructs of the programming environment and considerations in improving the user experience. These choices can be seen in architecture of the "Multiple Worlds" internal structure, as discussed in Section 5.1.2, in the use of AutoCAD, as discussed in Section 5.2, and the high and low levels components identified for the simulation model, as discussed in Section 5.3.

The instantiation of the "Multiple Worlds" design method to support the design of other types of logistics systems requires a careful analysis that is guided by the following questions:

1. How can the design process be phased in different levels of abstraction?

2. How can the logistics system be decomposed in components?

3. How can the logistics system be visualized?

4. What are the perspectives needed to support all relevant actors in the design process?

5. How can the users input designs at different level of abstractions?

We have seen in this chapter that we have answered these questions as follows:

1. Following the description of the case study in Chapter 2, we have chosen to mimic the different phases followed within APM Terminals to design container terminals. During the initial phases, rough estimations are made following a fast analytical model. After this, more detailed analyses are made using simulation models. We have implemented this by introducing the high level and low level designs.

2. We have chosen to decompose the system following a physical decomposition of the technical system, being the container terminal. The physical decomposition is achieved by modeling the equipment described in Chapter 2 as individual simulation models.

3. We have visualized the automated container terminals using a 3D virtual environment. This choice has been based motivated in Section 5.4.

4. In Chapter 2, we have identified three major perspectives: business, operations, and environmental. The output that results from the simulation experiment using the component library discussed in 5.3 indicate the operational performance of the terminal. These results can be analytically mapped to output the financial and environmental aspects of the container terminal, as discussed in Section 5.5.

5. We have chosen to implement a customized user interface to input the parameters to construct high level models. For the low level models, we have chosen to use AutoCAD, following the current practice of designing container terminals in APM Terminals.

In Chapter 4, we identified issues that we chose to solve using our method. To demonstrate that the "Multiple Worlds" method solves these issues, we related each issue to a construct of the method in Table 4.3. Similarly, we reflect on the instantiation of the method and relate each issue to an aspect of the instantiation in Table 5.1.

The method presented in Chapter 4 is constructed of different parts: the formalism, the semantics of the formalism and the design process. As part of the formalism, this chapter discussed the component libraries $\mathcal{L}$ in Section 5.3, the component transformation function ζ in Section 5.3.3, the model building process defining the initialization selection function η in Section 5.3.4, and, finally, the decisions and the branching function ρ as presented

Issue	How
Issue 1: Unstructured design process	The process is structured as shown in Section 5.6, mainly using the design editor discussed in Section 5.1.1.
Issue 2: Manage different system designs	The design editor that supports the users in maintaining different system designs is discussed in Section 5.1.1.
Issue 3: Document the process	The design editor supports the users in storing each intermediate design in a tree. Moreover, the process stimulates the users in documenting each node.
Issue 4: Compare different designs	The design editor allows users to construct different designs. The visualization client and statistical output shows the output of several simulation models.
Issue 5: Mutually consistent conceptual and detailed designs	The design editor stimulates users in constructing designs in as children to appropriate parent nodes.
Issue 6: Involvement of actors throughout the process	The design process and the several forms of output support actors throughout the process.
Issue 7: Predict the influence of design decisions	Using the simulation models, design decisions can be assessed using quantitative output.
Issue 8: Understandable presentation	Several methods for presenting the output are available, including output in 3D virtual environments.
Issue 9: Interactive and iterative process	The design editor and process stimulates actors in iterating on their designs using the interactive tools that are part of the design environment.

Table 5.1: The issues identified in Chapter 4 in relation to aspects of the instantiation of the method.

in the “Multiple Worlds” user interface presented in Section 5.1. On the process side, we have shown how the method can be instantiated in Section 5.6.

Vahidov [Vah05] states that an instantiation, as part of the design science methodology, should contain a working prototype that exhibits the major essential features and processes supported. The prototype can demonstrate the feasibility of the solution, and can also be used in experiment assessments of the value of the concept. We have constructed the instantiation of the “Multiple Worlds” design method to demonstrate the technical feasibility of the method and to describe its deployment to support the design process of automated container terminals. In the next chapter, we will use this instantiation to carry out an evaluation. The evaluation will be used to assess the usability of the artifact and to evaluate the potential use of the artifact in a business environment.

Chapter 6

Evaluating the method

Hevner [Hev10] emphasizes the importance of evaluation by stating that "evaluation is a crucial component in the design science research process". He follows by exposing the two philosophical groupings that determine how the actual investigation takes place. As discussed in Chapter 1, the two major epistemological stances taken in information research, are positivism and interpretivism. The first "suggests that the merit and worth of an information resource can in principle be measured with all observations yielding the same result; rational persons can and should agree on what attributes of a resource are important to measure and what results of these measurements would be identified as a most desirable, correct, or positive outcome" [Fri97]. The latter "says that what is observed about a resource depends in fundamental ways on the observer" [Fri97]. In Chapter 1, we took s critical realist ontological and interpretivist epistemological stance. In this chapter, we will discuss the implications it has for our evaluation and how this has been established in our research.

6.1 Validation following an interpretivist epistemology

Gonzalez [Gon10] argues that an interpretivist epistemology requires DSISR research to have both a rigorous as a relevant contribution. Rigor is achieved through the effective and transparent use of existing research. Therefore, part of validating a DSISR contribution means "documenting the construct definitions, the model descriptions, and the methodical choices used in building the artifacts". Following Iivari [Iiv07], this is accomplished by describing the practical problems and opportunities, comparing the new artifact with existing ones, explaining analogies and metaphors that have been used in the construction of the artifact, and identifying the kernel theories. For this research, this is presented in Table 6.1.

Hevner [Hev10] follows a positivist view when describing the evaluation of DSRIS ar-

Practical problems and opportunities	The research opportunity has been established during our case study, discussed in Chapter 2.
Comparable artifacts	Comparable artifacts have been discussed in Chapter 3. Moreover, in Chapter 4, we discussed artifacts that contain elements that have been used in our method. Finally, in Chapter 5, we discussed our instantiation that refers back to comparable software environments that allow similar functionality.
Analogies and metaphors	In Chapter 5, we established an ontology for automated container terminals that has been used to construct our simulation model library.
Kernel theories	In Chapter 3, we identified the theories from hard and soft systems thinking and from M&S that went into our method. More specifically, Table 4.3 discusses which theory contributed to solving each issue we identified during the relevance and rigor cycle.

Table 6.1: Iivari's [Iiv07] four aspects to establish the transparancy of a DSRIS contribution

tifacts. This entails using quantitative methods that yield precise statistical analysis of performance over time, which implies a strict means-ends utility. The fallacies of this approach would lie in a weak link of the instantiation with the underlying theories, the temporary nature of the validation (i.e. Popper's provisionally acceptance of a theory until a possible refutation [Pop72]), and, finally, the contextual limitations of a DSRIS artifact [Gon10].

Contrarily, interpretivist evaluation strives at providing a rich understanding on "how an IT artifact is really appropriated and used and what its effects are, without confining the focus on the given ends of its initial construction" [Iiv07]. To accompany this stance, Iivari argues that any DSRIS artifact should be tested extensively in laboratory and experimental situations before evaluating it in real business environments.

6.2 Evaluating the "Multiple Worlds" design method

Within the design cycle of the DSRIS methodology, evaluation plays a crucial role. In this cyclic view, evaluation is carried out to enter new iterations of the design cycle based on shortcomings or improvements identified during the evaluation stage. This cycle puts Simon's notion of design into practice by stimulating an iterative design process where the artifact is refined until satisfaction. Iivari's emphasis on carrying out evaluations in laboratory and experimental situations requires that we carefully frame the goal of the evaluation in our research. Nunamaker and Chen [Nun90] posit that the evaluation tests should focus on the performance and usability of the new information system.

Siau and Rossi [Sia07] elaborate on evaluation techniques for system analysis and design methods. They present several evaluation techniques plotted on a two dimensional diagram: ontological and epistemological. In the quadrant that reflects the ontological and epistemological stance taken in this research, techniques are presented that focus on interacting with study participants and try to minimize objective separateness [Cre97]. We can posit group interviews [Fre91] as being in line with these techniques whenever that data generated from this technique is qualitative.

In this research, usability tests with graduate students allow us to experiment with the artifact (method and software environment). Using group interviews, we collect a rich dataset that feeds the design cycle for further iterations.

6.2.1 Usability experiment

To carry out the qualitative usability evaluation, we have organized a workshop with 9 graduate students from Delft University of Technology. The students were divided in groups of 3 to conduct a design study using the proposed method. To simulate a multi-actor environment, we assigned a role for each actor: the business analyst, the performance engineer, and the environmental analyst. The roles were accompanied by role descriptions to stimulate the specific behavior expected from that actor. Each role had specific goals that are congruent with reality and each goal was quantified in terms of costs, operational performance indicators or pollution threshold. This subdivision of roles was used to stimulate the discussions and disagreements that are predominant in multi-actor environments.

The experiment was subdivided in 6 phases spread over a 4.5 hours time span. During the first phase, a domain expert from APM Terminals gave a general presentation on the design process of automated container terminals. The presentation thoroughly explained important concepts to design container terminals, such as calculating the required throughput of a terminal given its physical proportions. The various types of equipment, such as cranes and vehicles were presented with technical specifications. During and after the presentation, the participants had the possibility to ask questions to understand the subject matter. During the second phase, a demonstration was given on the support environment and on how to use it. In the third phase the participative design process required the teams to create a high level design specifying the number of equipment and total capital investment of the terminal. The fourth up until the sixth phase consisted of using the detailed simulation model to analyze the performance of the specific functions of the terminal: the loading and unloading on the quay, the horizontal transportation using automated vehicles and the storage at the yard. During the last phase, the participants had the opportunity to freely explore the design space by climbing back up the design tree and try different alternatives for their designs. A fictive table of equipment was presented that contained three types of each class of equipment: quay cranes, RMGs and AGVs.

The specification of the equipment is fictive, but is balanced to prevent straightforward combinations. The types of equipment and their specification are listed in Table 6.2.

Equipment type	Equipment speed (km/h)	OPEX (Euro per operational minute)	CAPEX (Euro per unit)	Pollution (gram CO_2 per operational minute)
QC type 1	5.5 (frame)	2.45	9500000	1100.54
QC type 2	6 (frame)	2.1	9600000	1265.6
QC type 3	6 (frame)	2.2	9200000	1152
RMG type 1	4.7 (frame)	0.8	3000000	210.53
RMG type 2	6 (frame)	0.762	3200000	225.3
RMG type 3	5.2 (frame)	0.72	3240000	215.9
AGV type 1	25	0.18	550000	95.12
AGV type 2	21	0.25	650000	90.36
AGV type 3	21	0.2	580000	92.6

Table 6.2: The specification of the equipment used in the usability experiment.

Besides the design environment, the participants were equipped with a technical infrastructure to support and stimulate group work. To support the collaboration among multiple participants, we used interactive whiteboards and 3D visualizations. Every group could use 2 interactive whiteboards each connected to a different computer: one for the design environment and on to display an Excel spreadsheet to keep track of the financial, operational, and environmental aspects. A second Excel spreadsheet was provided to construct a matrix to carry out a multi-criteria decision analysis. The interactive whiteboards were used to display the animation output of the simulation runs and to discuss the designs in AutoCAD. The physical setup and the participants working with the environment are shown in Figure 6.1.

6.2.2 Results and discussion

The main source of data collection for this usability experiment was a semi-structured group interview. To give the participants time to formulate their answers, we prepared a questionnaire (see Appendix B using the questions of the semi-structured interview. The following statements were presented and participants could respond on a 7-point Likert scale (from 1, totally disagree to 7, totally agree):

- The method supported me in inputting from my actor's perspective.
- The method supported me in protecting both my interests and the collective interests.

Figure 6.1: (A) the physical setup for each group of participants, (b) the participants at work.

- The method supported me in following the design process without losing sight of my goals.
- The method supported the construction of enough alternative designs.
- The method enabled the specification of the decisions using the predefined components and parameters.
- The method enabled the comparison of designs using the results of the simulation model presented through the animation.
- The method enabled the comparison of designs using the results of the simulation model presented through the statistics.

Figure 6.2 shows the results of this questionnaire. From an actor perspective, students indicated that the environment supported the group better than the individual stakeholder. The approach also guided the group in their design effort and helped them to structure their design decisions. Finally, students gave a higher score to the support by statistics than by animation. The semi-structured interview provided more insights on the participant's experiences. There was a consensus on the fact that the method helped cooperate with different actors from different perspectives and that the possibility of exploring different design alternatives helped the participants carrying out their task. However, the software environment to support the method still has some issues. The major problem is AutoCAD: using this design environment, the participants have to follow a strict way of drawing using pre-existing blocks and a certain naming convention. If one deviates from this, the simulation environment cannot run correctly. Another minor issue was the limited output:

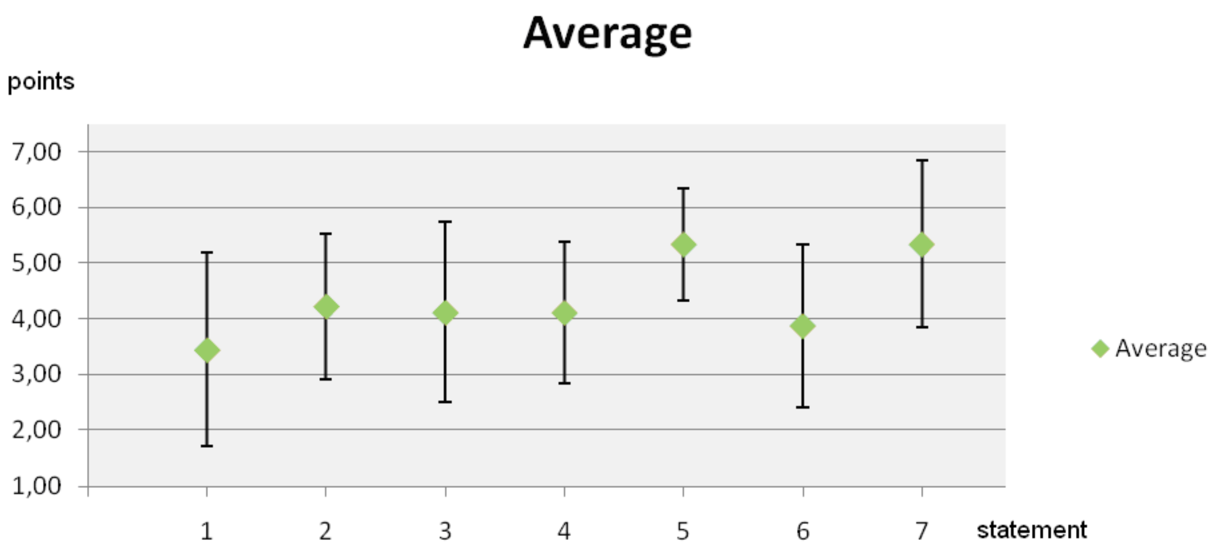


Figure 6.2: Results of the initial questionnaire on a 7-point Likert scale.

although the participants could use the statistics and visualization to study the simulation output, they suggested having a more extensive set of statistics to get a more detailed view on the performance of the system.

Initially, we expected that the large number of alternatives would increase the complexity of design study. On the contrary, we found that the participants were effectively using the design tree to structure their decision process and to follow the design process. The participants constructed a large number of design alternatives and used the simulation models to quickly assess the implications of their decisions. This led them to discuss the designs and argue for a certain decision based on a better shared understanding of implications of that decision. The animations and statistical results were used to understand the performance of the system from the different actor perspectives, thus helping the group to create a more holistic view of the design choices they faced.

6.2.3 Evaluating the design and the design process

Each group that took part in the usability experiment handed over its designs and description of the design process at the end of the experiment. We selected the most extensive design process (in terms of number of nodes) to base a discussion on the results of the usability experiment with an expert in automated container terminals of APM Terminals. As the expert was also involved during the usability experiment, the discussion was not restricted solely to the results of a single group, but was taken more broadly to encompass all groups. We asked the expert to evaluate the results in terms of quality of the design and quality of the process.

The evaluation criteria selected by the participants to compare designs based on the multi-decision criteria analysis, were: total pollution, total CAPEX, total OPEX, total revenue, and "time until profit". Based on these criteria, they constructed the tree displayed in Figure 6.3.

The expert began his analysis by noting the different steps that are made throughout the design process. Subdividing the design process into the different steps, called abstraction levels in our method, allows for a structured process and simplifies the alternation of designs at each level. The solution space is large in this type of design projects and this method shows that the structured approach guides the designers through the process.

The expert continued his analysis by remarking the extensive first level of the tree. The selected tree shows an alternation of four different designs from which one design has been chosen to continue the design process. The expert explained that in practice, a comparison is made of fewer designs, typically two. This larger comparison, thanks to the high level simulation models, was appreciated by the expert as allowed the exploration for non-trivial designs. The method allows users to explore a larger portion of the solution space, which is considered to be a favorable feature. The expert remarked that in practice, designers follow the stream towards one design. A final remark on this concerned the comparison of the designs: using a multi-criteria decision analysis is considered to be a valuable addition to the method.

6.3 Expert evaluation

During the design cycle, we focused on the usability of the "Multiple Worlds" design method and its supporting software implementation. This allowed us to identify potential improvements that influence the efficiency of the proposed method. However, to close the relevance cycle, we need to evaluate the effectiveness of the method. This entails evaluating the use of the method in a business environment.

The interpretivist stance of our research calls for an evaluation method that is able to construct a rich picture on the effectiveness of the design method. Generally, this could only be acquired *a posteriori*: based on a number of applications of the method, a general statement on the effectiveness could be made. However, this is not desired for two reasons:

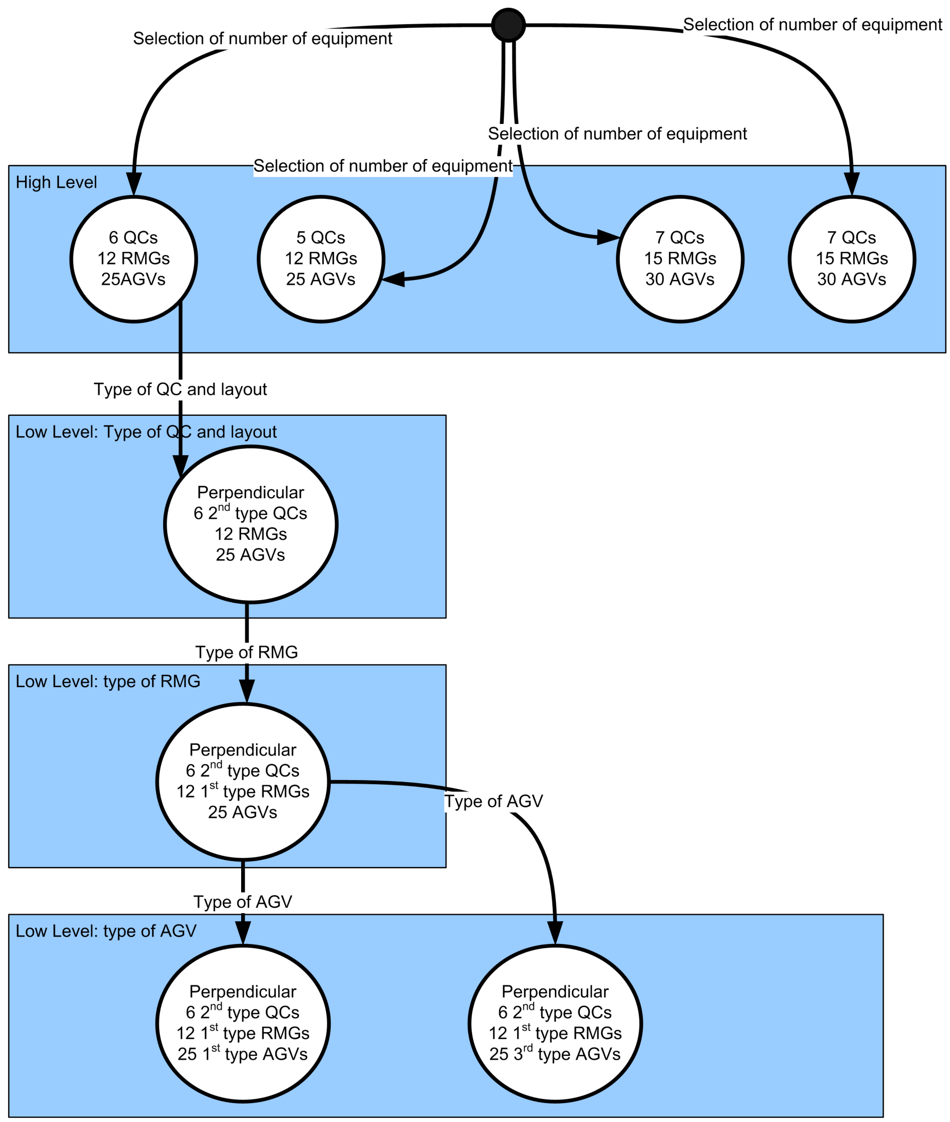


Figure 6.3: A tree constructed by a group of participants of the usability experiment. The reference to the types of equipment can be found in Table 6.2.

1) design projects of container terminals have a long throughput time, which means we do not have any real data for empirical analysis, and 2) employing a design method that is not accompanied by any statement of effectiveness has a high risk. Therefore, an *a priori* evaluation is considered in this research.

Following the relevance cycle, we aim at acquiring informed opinions about the artifact and suggestions for possible improvements. According to Sandelowksi [San98], expert's opinions can be used for precisely these aspects of a research project. Galbraith *et al.* [Gal06] carried out an empirical study to analyze the predictive ability of experts for technology success. Although they concluded that experts have little predictive power on commercial success, a sub-sample of the experts, namely scientist and engineers, are able to carry out an evaluation on the technical dimensions of a proposed technical solution.

Following Sandelowski [San98] we differentiate between different kind of experts, namely domain experts and methodological experts. This is also related to Galbraith *et al.* [Gal06], who conclude that scientists and technical experts bring value to an assessment. In our research, we can relate to this by selecting experts that have a scientific background in the field of M&S for logistics systems and by selecting practitioners that have a substantial experience in the design of (automated) container terminals.

6.3.1 Semi-structured interview

To carry out an expert evaluation, we opted for a semi-structured interview [Lin02]. To formulate the questions, which can be found in Appendix C, we differentiated between method and instantiation, in accordance with our research methodology. We focused on several aspects: level of expertise of the participants, perceived problems they encounter in their organization, perceived problems with the proposed method and instantiation and possible adoption of the method and instantiation in their current organization. Although the aim was not to strictly adhere to a certain evaluation model, inspiration was taken from the Technology Acceptance Model [Dav89] and the Value-Frequency Model [Bri09] to construct the questions. This was due to the fact that both models are inherently quantitative, thus contrasting with our epistemological stance. However, both models do provide perspectives that are valuable to analyze the effectiveness or usefulness of IS.

To carry out the interviews, we selected a number of experts from different organizations. We requested the participation of professors in the field logistics and simulation, managers in companies specialized in design projects using simulation, and companies involved in the design of automated container terminals. We selected individuals that can be considered experts based on years of experience (more than 10 years or professional experience) and based on academic degree (M.Sc. or Ph.D. in a relevant field). The experts are all currently employed at a management level on projects concerning the design of logistics systems, mostly container terminals. To ascertain their expertise, several questions were asked concerning their past projects. The list of experts can be found in Appendix C, subdivided in domain experts and methodological experts.

The interviews were all recorded using a voice recorder after receiving permission from the participants. During the interview, notes were taken to highlight important points of the answers. A summary of the interview was later written out, taking into consideration the objectives of the interviews. Confirmation of this summary was requested to all participants to avoid wrong interpretations on the researcher's behalf.

6.3.2 Interview results

The interviewees presented, not surprisingly given their line of business, that simulation models are used often to support decision making in logistics management. To design logistics systems using simulation models, designers need to be familiar with the use of simulation models and need to trust the results of those models. Familiarity with simulation models means to be aware that models are built to answer specific questions, within a range of uncertainty, and with several assumptions. However, this is one of the first hurdles encountered during design processes supported by simulation models: designers are often reluctant to use models they do not understand, partially or fully. The risk of presenting simulation models that do not conform to the expectations of designers, is that they will leave the simulation models aside and continue without. Several interviewees pointed out that the introduction of a new method that uses simulation models, will therefore meet resistance. The resistance is therefore due to "fear of the unknown" instead of any intrinsic quality of the design method.

Several interviewees emphasized that simulation models should be used earlier in the design process to reduce risks and "bad decisions". Many initial decisions during a design process are made based on best practices, *guesstimates* and "gut feeling". The consequences of these decisions are felt later in the simulation process when it is more difficult to introduce new changes. The use of simulation models in the initial stages of the design is, in practice, often reduced to a specific subsystem. This does not capture the interrelation of the design decisions with other subsystems, thus denying the access to a holistic view that is much needed throughout a design process.

Using simulation models from the beginning and throughout the design process, allows designers to iterate on their design. The presented design method, particularly its tree structure, helps structuring the design process for different levels of abstractions and simplifies alternating between several designs. However, the actors in the design process should be fully aware of the questions they are allowed to ask in every level of the process, as the simulation models are built at that level of abstraction. Several interviewees particularly liked the idea of iterating and alternating (testing an alternative and see what the consequences are) throughout the design process, which allows improving the design throughout the process. A feedback-loop is mostly missing in traditional design approaches, something that can be solved with the proposed method. A note should be put on the use of simulation models from the very beginning: using simulation models from the very beginning can help, but awareness should be brought to the fact designers are dealing with models and not the real system. It is also important to note that modern logistics systems tend to get too complex to grasp without supporting simulation models.

On the practical use of simulation models, attention should be brought to the possibility of having simulation components ready before or during the design process: developing models is a costly and time-consuming effort. Therefore, the use of simulation models should be justified with regards to the problem at hand. This means that the monetary investment should be considered carefully, especially comparing it to less costly models, such as analytical models. On the latter, one should consider that analytical models cannot easily capture the dynamics of the system at hand and the interactions between and interrelation with the different subsystems of a logistics system. A risk is attached to using models at a too high level of abstraction: models lacking too many details have a risk of becoming useless during the design process as the results of such models cannot be interpreted correctly.

Using simulation models to guide the process has advantages from a collaboration perspective. Discussing design decisions based on quantitative data supports the comparison of several decisions and makes the choice between these decisions easier, given a set of decision criteria. The proposed design method therefore would support precisely this aspect of a collaborative design process. An interviewee pointed out that KPIs can streamline the discussion. Each actor, who has a specific background and interest in the design process, needs to be able to collaborate from his specific perspective. One of the experts phrased this as follows: "when you start to consider the various dimensions of a design, finding a common communication mechanism is an important thing; this tool can fill that goal.". The method would support the design process in helping the actors understand the problem: most simulation projects are not about finding the optimal solution but are, in constrast, about having actors understand the system and make decisions that satisfy their requirements. Collaboration is felt as a requirement in design processes: mostly stakeholders work independently and come together in a later stage. Collaboration between different actors in a design process is hard: a design needs to be accepted by all actors involved based on criteria (e.g. staying within budget and time). An integral approach and a structured process, using a tree, allows actors to come together and have design method that supports them. Comparison is possible if a set of decision criteria is set *a priori*, which is something that has to be ascertained.

Some criticism has been uttered on the use of AutoCAD. The steep learning curve of the software package and the required level of detail would be inappropriate for the type of design processes that have been described in this research. Similarly, designing container terminals focuses mainly on fine-tuning the behavioral aspects (such as the control system) and less about physical properties (such as spatial relation). On the same note, one expert suggested to implement this method using a human facilitator that guides the design process and is proficient with the software environment. If not, the usability of the software environment could become a barrier to using the design method.

On a final note, the interviews led to novel applications of (parts of) the design environment. A documented and structured design process can function as a knowledge management system: decisions that have a tendency of recurring throughout multiple design projects can be stored for future reference. The application domain of the design method should not be restricted to solely automated container terminals. The method has the potential of being used in every large-scale design project, particularly engineering projects but also for other domains where simulation models are being used. Moreover, the simulation models could be used further in the lifecycle of the system for operational purposes (such as managing a container terminal). Lastly, the simulation models could be used for training purposes as well.

6.4 Reflection and summary

In this chapter, we have focused on three parts: evaluating the artifact as part of the design cycle; validating the DSRIS contribution by describing the practical problems and opportunities, comparing the new artifact with existing ones, explaining analogies and metaphors that have been used in the construction of the artifact, and identifying the kernel theories, as part of the rigor cycle; and, finally, feeding the artifact back to the business environment by carrying out interviews with domain experts. We reflect back on Table 4.3 presented in Chapter 4 and Table 5.1 presented in Chapter 5 to illustrate that the three activities presented in this chapter result from the work presented in the previous chapters. This reflection is presented in Table 6.3.

In Chapter 5, we reflected on how the different parts of the method have been implemented to support the design process of automated container terminals. Similarly, we want to show how the instantiation of the method has been used during the evaluation described in this chapter. In Chapter 4, we showed the construction of a tree in Figure 4.1. During the evaluation workshop, a tree has been constructed for automated container terminals, shown in Figure 6.3. This tree has been constructed by using a well defined process, explained in Section 6.2.1, that is based on the flow chart presented in Figure 5.21 of Chapter 5.

Issue formulation	Usability experiment	Expert evaluation
Issue 1: Structured process	The structured approach helped participants follow the design process.	The structured approach is perceived useful in collaborating and comparing results.
Issue 2: Manage different system designs	Participants were capable of comparing different designs.	Quantitatively comparing alternatives helps collaborating between different actors.
Issue 3: Document the process	(not evaluated)	Storing a design process can be useful for future reference.
Issue 4: Compare different designs	Participants used the comparison to make a decision on certain designs decisions.	Quantitatively comparing alternatives helps collaborating between different actors.
Issue 5: Mutually consistent conceptual and detailed designs	(not evaluated)	Attention should be given to the choice between analytical models (high level) and simulation models (detailed design).
Issue 6: Involvement of actors throughout the process	Each participant was involved in the design process.	Collaboration is required throughout a design process.
Issue 7: Predict the influence of design decisions	The results from the simulation models were used to make comparisons.	KPIs help streamline the discussion between several actors.
Issue 8: Understandable presentation	The 3D and statistical output were used to compare designs.	(not evaluated)
Issue 9: Interactive and iterative process	Participants were able to design collaboratively.	Iteration is required in a design process.

Table 6.3: The issues identified in Chapter 4 in relation to aspects of the evaluation of the method using results from the usability experiment and the expert evaluation.

Chapter 7

Epilogue

Designing logistics systems is a challenging endeavor that requires the contribution of many experts from different fields. In this research, we have presented a design method that tackles a set of issues that manifest themselves in multi-actor design processes that are supported by simulation models. The latter quantifies the influence of design decisions on the system, but also presents a number of drawbacks in terms of required effort and specialized knowledge needed to use the models.

To carry out the research presented in this dissertation, we have followed the design science methodology. This methodology guides the researcher through several cycles guaranteeing a balance between rigor and relevance of a research contribution comprised of a novel artifact. In our research, we went through the relevance cycle in Chapter 2 and carried out a case study at APM Terminals regarding the design process of automated container terminals. The relevance cycle lead to a research opportunity that we examined in our rigor cycle in Chapter 3. During our rigor cycle, we reviewed the existing body of knowledge on SE, and M&S and we identified a set of issues that we chose to cover in order to enter the design cycle. The main contribution of this research has been presented in Chapter 4: a design method for logistics systems that covers the issues identified in Chapter 3. In Chapter 5, we instantiated this method to support the design process of automated container terminals. To achieve this, we implemented the formalism discussed in Chapter 4. The instantiation is comprised of two parts: the supporting software implementation and the process description. As the final step of our research, we carried out the evaluation step of the design cycle, described in Chapter 6. Hereafter, we closed the rigor cycle by structuring our contribution and the relevance cycle by carrying out interviews with domain experts.

In this chapter, we will summarize our findings in light of our research objectives and our research questions, discussed in Chapter 1. We will conclude this dissertation by discussing future work that can spring from our findings.

7.1 Research findings

We began our research pursuing the research objective presented in Chapter 1. To satisfy this objective, we defined three research questions that we used as a guide throughout our research. The first research question (RQ 1) has been mainly covered in our case study, in which we analyzed the current use of simulation models in a representative case such as the design of automated container terminals. The second research question (RQ 2) covered existing methods and models that are known to cover several aspects of designing systems in multi-actor environments using simulation models. Lastly, the third research question (RQ 3) referred to the design of a design method that incorporates the findings from RQ 1 and RQ 2. These questions reflect our research methodology: design science in information systems research. This methodology presents a three-cycle view to research: the relevance cycle to identify problems or research opportunities in business environments, the rigor cycle to identify the kernel theories appropriate to carry out the design cycle.

To answer the first research question, we carried out a case study at APM Terminals as part of the relevance cycle. During this case study, we analyzed their approach in designing automated container terminals. The analysis encompassed both the technical as the actor perspective. From the technical perspective, we demonstrated the technical complexity of designing automated container terminals: choosing between several competing types of equipment, identifying the required numbers, and developing the appropriate control system for the complete system. From the actor perspective, we clarified the role of each actor throughout the design process and how this role affects his interests in the design process. To face this complexity, design studies using simulation models are carried out. The current use of simulation models is positioned in the evaluation phase of the project lifecycle. We concluded that a repositioning of simulation models throughout the design process could benefit the design study.

With the knowledge and research opportunity identified during the relevance cycle, we entered the rigor cycle. During this cycle, we reviewed existing theory on system science, and M&S to cover RQ 2. With the notions of hard and soft systems, we identified several uses of simulation models. We concluded that the classical view of simulation models as an evaluation tool is being replaced by the use of simulation models as a way to collaborate on design studies by using the results of the simulation runs to guide the design process. To use modeling and simulation throughout the design process, several issues need to be addressed. Although existing methods address these issues separately, no method has been identified that covers each issue. Therefore, we concluded that we need to use these issues in our design cycle.

Our design cycle is presented as the resulting method in Chapter 4, the instantiation in Chapter 5, and, partially, in Chapter 6 that covers the evaluation. These chapters cover RQ 3. The presented method is formally presented as the “Multiple Worlds” formalism in Chapter 4 and the associated semantics. The method has been designed based on the theories and issues identified during the rigor cycle. We instantiated the method to support the design process of automated container terminals. This instantiation allowed us to evaluate the method in a laboratory setting by organizing a design workshop. The results from this workshop brought to light benefits and drawbacks of the method regarding usability of the method and supporting environment, the quality of the process and the resulting design. We followed our interpretative standpoint by collecting qualitative data that provides a rich picture in order to re-enter a new design cycle as part of future research. Finally, we closed the relevance cycle by conducting expert interviews assessing the usefulness of the method in business environments and we closed the rigor cycle by demonstrating the transparency of our contribution.

From the final research steps, the evaluation and the closing of the relevance cycle, we can conclude that we addressed an issue that is present in current business environments. The data collected during the workshop suggests that we covered the relevant aspects of a design study in a multi-actor environment and enables us to refine the design method further. As remarked during the expert interviews, the early use of simulation models has the potential of reducing decisions that are based on an, otherwise, inprecise estimation of the system's performance. Moreover, using simulation models to support the collaboration of multiple actors can streamline the discussion, which could be based on KPIs instead of opinions. However, the workshop and the expert interviews brought to light some limitations of the current method and its implementations. The method is based on simulation models to assess the designs: this can meet resistance from the people involved in the process. If the people do not accept the results, the method will not be strictly followed. Which brings us to a following issue: if the method and its supporting environment are felt to impede the process as they require a steep learning curve, it will not be used. However, to tackle this issue, a facilitator can be used that guides the designers throughout the process. This finally brings us to the last issue, namely the choice for an input environment of the designs. In our instantiation, we chose to use AutoCAD reflecting the current use of it at APM Terminals, where we carried out our case study. However, the level of detail and expertise required for this design environment is considered to be a sub-optimal match to the design method and its instantiation presented in this dissertation.

7.2 Reflection

In this dissertation, a multi-actor simulation-based design method has been introduced for logistics systems. The research has been carried out following the design science methodology and the chosen application domain is automated container terminals. We reflect on this research from three perspectives: its research methodology, its results and its applicability on other logistics systems.

The design science methodology provides a way of balancing rigor with relevance in scientific research projects on information systems. We have chosen to carry out a case study at APM Terminals to study a business environment, which, in DSIRS terms, is needed to derive the business requirements. In this research, this did not result in identifying a concrete problem, but instead helped us formulate a research opportunity. During the expert interviews, the presented method was discussed in light of other possibilities that were not formulated earlier during the research project. Thus DSIRS is suited for exploratory research and not limited to research projects with well defined problems.

A moment of reflection is also required on the results of the research. As discussed in the first chapter, we do not aim at providing definitive answers that compare the presented method with existing methods, which could be considered a positivist approach. We took an interpretivist stance and collected data that could provide us with the means to continue the research following DSIRS. To do so, we focused on distinct aspects of the method that can be improved in further iterations. The expert interview as part of the relevance cycle helped us further understand the needs of a business environment and the circumstances in which the method might be beneficial.

Finally, we are also interested in the application of the method on other logistics systems besides automated container terminals. Although the method is domain independent as formulated in Chapter 4, the instantiation is domain dependent. In the concluding part of Chapter 5, we discussed the relevant questions on instantiating the method to design other types of logistics systems. We argue that using these questions as a starting point and using the presented instantiation as a reference, new instantiations could be developed for other types of logistics systems.

7.3 Future research

Future work can be identified in three aspects: support multiple actors on different perspectives, extending the existing component library, and apply the method in other domains.

During the relevance and rigor cycle, we acknowledged the fact that designing logistics systems is an effort that requires the input and insight of several actors. However, collaboration is required between these actors, making it a daunting task. Simulation models are often presented from one specific perspective, in our case the operational perspective, and the results are transformed to serve other actors with a different perspective as well. In our research, this meant that we simulated the operational aspect of the system and transformed it into a financial and environmental picture. However, this assumes that a logical transformation can be made between two different perspectives in terms of KPIs. In case this is not possible, a model needs to be made in terms of the other perspectives. This issue has been addressed by Tekinay *et al.* [Tek10] and several possible solutions have been proposed. The results from this research may improve the collaboration between different actors in a simulation based design study.

Regarding the extension of the component library, we believe that the current component library can be used to design complete container terminals by developing the land-side components. This means the inclusion of components to handle the rail and truck operations. Moreover, the existing ship-to-shore crane models can be used for developing barge crane models as well. A complete component library would mean that we can refine the design further in the design cycle and use the resulting design environments in commercial projects.

Lastly, the method can be applied for designing other logistics systems besides container terminals. Straightforward applications would be the application on similar technical systems such as rail networks, distribution lines in warehouses, and baggage handling systems in airports. However, a step further could be made, going outside the scope of this research, and considering the application of the method on assembly lines of factories, energy networks, and on supply chain networks. Many multi-actor design studies can prosper by using simulation, thus requiring a simulation based design method such as the one proposed in this dissertation.

Appendix A

Interview regarding the design process at APM Terminals

This interview was part of the evaluation phase of the 'Virtual Terminal' (VT) developed for APM Terminals. The VT is a visualization software tool for automated container terminals. The tool is intended as support during the design of automated container terminals. Four important aspects characterize the VT: the automated extraction of the virtual environment description from design drawings, a realistic 3D visualization of the virtual environment, the possibility to structure information into context and finally the means for communication and presentation. Creating a virtual environment is a tedious task which usually involves manual labor requiring months of work. During the design process of a container terminal, this waiting-time would be a huge set-back. Commonly, 3D impressions are made only once the design process is over and the design of the new container terminal is finalized. However, during the design process many design drawings are made in authoring tools as AutoCAD. Having a mean of translating these drawings to descriptions of virtual environments, can therefore facilitate the task of constructing the virtual environment and significantly shorten the time needed for it. To achieve this, an AutoCAD-plugin has been developed which translates the design drawings into an XML-file that serves as a description for the virtual environment. The translation is based on an ontology for automated container terminals: this ontology allows a mapping of entities in the 2D design drawing to entities in the virtual environment. This approach on translation can best be described as an extension to the library based approach on data translation from CAD to VR as described by Whyte *et al.* [Why00].

With a valid description of a virtual environment, a realistic 3D presentation is possible. To achieve this, 3D models have been developed based on blueprints of equipment and detailed photographic material which resulted in high-detailed models. These models are thus visualized in the VT. The VT has been used as the visualization client in the instantiation of the design method described in this dissertation.

Following Keen and Sol [Kee08], we evaluated the VT on usability, usefulness and usage. We described this evaluation in Fumarola and Versteegt [Fum11]. Part of this evaluation provided us insight into the design process of automated container terminals. We used the following set of questions to carry out interviews with 5 employees that are highly involved in these processes.

On the original situation:

- How were the designs of new container terminal presented to external actors/stakeholders?
- How time and effort consuming was it to prepare such a presentation?
- How clear was this presentation material to external actors/ stakeholders? Did you encounter difficulties in explaining a new design?
- Briefly describe your experiences with this way of presenting: Which were the weak points? Which were the strong points?

On the usefulness:

- Can you identify the phases in a design study for a new container terminal and pinpoint in which phases you find the Virtual Terminal to be useful?
- Can you identify the actors in a design study for a new container terminal and pinpoint for which actors you find the Virtual Terminal to be useful?
- How well is the Virtual Terminal appropriate in presenting new container terminal designs to external stakeholders?
- Which benefits does the Virtual Terminal provide over traditional (previous) presentation materials?
- How well does the Virtual Terminal perform in presenting new container terminal designs? Do external stakeholders perceive a better understanding of the design?
- Briefly describe your experiences with the Virtual Terminal: Which were the weak points? Which were the strong points?

On the usage:

- How is the Virtual Terminal being used in APMT?
- With regard to the phases identified in the previous question, do you think it supports the phases well?
- With regard to the actors identified in the previous question, do you think it supports the actors well?

- What are your expectations on the use of the Virtual Terminal in 1 year and in 5 years?
- What do you expect as an impeding factor in the future usage of the Virtual Terminal?
- Do you consider the Virtual Terminal as a competitive advantage towards other container terminal operators? Discuss why/why not.

As a final set of questions, we would want to explore how the Virtual Terminal can be improved for future use:

- What would you see as an improvement to the Virtual Terminal?
- Do you think the use of the Virtual Terminal should be extended? Do you think it should be used for non-automated terminals?
- How well would a dynamic terminal improve the VT? Do you think moving equipment would improve the final product? How detailed should these movements be?
- Discuss other points that could be improved about the VT.

We interviewed the following people:

- Cornelis Versteegt, Project Manager Maasvlakte II at APM Terminals (on April 8th, 2010 at APMT's Maasvlakte II office in Rotterdam (the Netherlands))
- Ross Clarke, head of Design & Operations at APM Terminals (on August 26th, 2010 at APMT's headquarters in The Hague (the Netherlands))
- Rik Geurtsen, General Manager Maasvlakte II at APM Terminals (on April 8th, 2010 at APMT's Maasvlakte II office in Rotterdam (the Netherlands))
- Dong Gui, General Manager Design & Innovation Department at APM Terminals (on August 26th, 2010 at APMT's headquarters in The Hague (the Netherlands))

The contents of these interviews have been recorded with permission of the participants. The summaries of the interviews have been used in our research.

Appendix B

Usability experiments of the “Multiple Worlds” design method

B.1 Experiment set-up

This document describes different aspects of the experiment to carry out with novice users (students): the scenario and tasks, the technical setup, the design process, (technical) support, and data collection.

B.1.1 Preparation

1. 3 interactive whiteboards with HP laptops
2. 3 interactive whiteboards with Dell laptops
3. Asking everyone if pictures and video may be taken
4. dividing the roles and groups (make sure one proficient user per group)

B.1.2 Scenario and tasks

- The design task: design an automated container terminal given an area (basic CAD drawing), a budget limit, a desired throughput, and a maximum pollution rate.
- Each actor gets a specific goal (at random) to account for his interests: budget, performance, and pollution. Each actor strives to protect its own interests: low budget, high performance and low pollution (precise numbers will be provided, using APMT presentation).
- Budget is split into CAPEX and OPEX

  - CAPEX: NrRMG * UnitPriceRMG + NrQC * UnitPriceQC + NrAGV * UnitPriceAGV + FixedCostsTerminal
  - Performance in terms of moves/hour, and equipment usage can be translated to OPEX.

- Pollution is calculated per equipment (e.g. "50 tonnes of CO2 emissions per straddle carrier, per year", Kalmar) using the performance measures (same as with OPEX).
- An appropriate spreadsheet is provided to calculate every KPI that is not directly outputted by the simulation model.

B.1.3 Technical set-up

- 1 beamer for presentation purposes:
  - Introductory presentation
  - For each part of the design process, a new presentation is given.
- Each group gets 2 smartboards:
  - 1 smartboard to design in AutoCAD and to visualize the 3D visualization and statistics
  - 1 smartboard to keep track of the design process using documentation and smartboard software:
- At each branch of the tree, users have to document (in a predefined Excel table) the following:
- Which decisions did they make?
- Which actors support/oppose each decision?
- Why did they make the decision?
- What do they expect from that decision?

B.1.4 Design process

The design process will be phased as follows:

1. First phase (30min):
   (a) (Expert) Introductory presentation on container terminals:
      i. What is a container terminal?

   ii. What are the different functions of a terminal?
   iii. Presentation of the different actors present in the design process and which interests they have.
   iv. Presentation of the performance matrix and how it can be used to compare different possibilities.
 (b) (Participants)
   i. Selection of the criteria for the columns of the performance matrix.
 (c) (Researcher) Introductory presentation on the method:
   i. What is it?
   ii. How does it work? Which different software components are there?

2. Second phase (30min)
   (a) (Expert) How are the different functions taken care of? (areas of the terminal)
      i. Presentation of each type of equipment used in the different areas of a terminal
      ii. Providing the rules of thumbs to calculate roughly the numbers of equipment needed. Explore the neighborhood of these figures.
   (b) (Researcher)
      i. Demonstration on how the first set of decisions can be inputted into the software
   (c) (Participants)
      i. Construction of various alternatives
      ii. Selection using the performance matrix to make a selection of alternatives.
3. Pause (10min)
4. Third phase (40min)
   (a) (Expert) What are the different types of QCs? What are the important points to take into consideration? (moves/hour)
   (b) (Researcher) Guidance on how to construct a CAD using the predefined drawing. Guidance on how to run the model and visualize the output.
   (c) (Participants)
      i. Construction of the model, assessments using the simulation.
      ii. Selection using the performance matrix to make a selection of alternatives.
5. Comparable to fourth, except for RMGs instead of QCs.

6. Pause (10min) between fourth and fifth.
7. Comparable to fourth, except for AGVs instead of QCs.
8. Free roaming phase (40min): The groups are allowed to improve the existing designs and create new ones according to their best judgment. Iterate over the existing designs.

B.2 Questionnaire

Thank you for participating in our experiment to assess the usability of the Multiple Worlds method and the supporting design environment. The Multiple Worlds method (simply called "design method" or "method" from now on) is intended to support designers in exploring the solution space of a given problem, in this case, the design of an automated container terminal. The design method constitutes the complete set of activities you have been through during the workshop: discussing and inputting different designs, comparing the alternatives using the supporting tools, and making a final decision on which design you consider to be the best alternative.

Your contribution to this experiment is really helpful to understand how the method can be used in a design process and how it can be improved in future research. To end the workshop, we would want to have your input on the design method. This questionnaire is needed to collect data on your past experiences in the design of logistics systems and your experiences using the proposed method.

B.2.1 You and your past experiences

The following questions are focused on your background and prior experiences.
What is your age?
What is your field of expertise (i.e. education, degree)?
How many years have you been professionally active in your area of expertise?

Please circle the appropriate answer:

1. Did you have any experience in modeling and simulation? (e.g. building and using simulation models for decision making) Yes / No
   If "Yes", state how many years of experience and in which role (e.g. student, professional):

2. Did you have any experience in the design of logistics systems? (e.g. design courses, involvement in projects on the subject) Yes / No

If "Yes", state how many years of experience and in which role (e.g. student, professional):

3. Did you have any experience in the design of container terminals in particular? Yes / No
   If "Yes", state how many years of experience and in which role (e.g. student, professional):

4. Did you have any experience in the use of CAD tools for 2D layout design? Yes / No
   If "Yes", state how many years of experience and in which role (e.g. student, professional):

B.2.2 Assessing the design method

The following statements are presented to have an initial assessment on the usability of the design method. These statements serve as a "warm-up" to a group discussion that will allow you to provide more information on your experiences as a user of the design method.

We would like you to respond to the following statements on a scale from 1 meaning "I totally disagree" to 7 meaning "I totally agree". Please put a cross in the appropriate box next to each statement.

	1	2	3	4	5	6	7
The method supported me in inputting from my actor's perspective.							
The method supported me in protecting both my interests and the collective interests.							
The method supported me in following the design process without losing sight of my goals.							
The method supported the construction of enough alternative designs.							
The method enabled the specification of the decisions using the predefined components and parameters.							
The method enabled the comparison of designs using the results of the simulation model presented through the animation.							
The method enabled the comparison of designs using the results of the simulation model presented through the statistics.							

B.2.3 Thank you

Thank you for providing the above information. We will now have a final discussion that will give you the opportunity to describe your experiences.

Appendix C

Expert interview: participants and protocol

C.1 Participants

The following people participated to the expert evaluation:

- Rienk Bijlsma, general director at Systems Navigator (Delft, the Netherlands), a company specialized in simulation models for logistics, material handling, and business process simulation. (domain expert)
- Vincent de Gast, project manager at Systems Navigator. (domain expert)
- René de Koster, professor of logistics and operations management at Rotterdam School of Management at Erasmus University Rotterdam (the Netherlands). (methodological expert)
- Rich Ceci, Vice President Information Technology at Global Container Terminals (New Jersey, USA). (domain expert)
- Yvo Saanen, technical director at TBA (Delft, the Netherlands), a company specialized in the design of container terminals using simulation models. (domain and methodological expert)
- Robin Audenaerdt, experienced project manager at ECT Euromax Terminal (Rotterdam, the Netherlands), APM Terminals (Rotterdam, the Netherlands), Port of Gothenborg (Sweden), and Cargonaut (Schiphol, the Netherlands) in projects concerning the design of logistics systems. (domain expert)
- Cornelis Versteegt, project manager in the Maasvlakte II-team of APM Terminals (Rotterdam, the Netherlands), specialized in the design of automated logistics systems. (domain and methodological expert)

- Arne Hedstrm, senior adviser at Danaher Motion Särö AB (Sweden), an R&D company specialized in automated equipment (including equipment for automated container terminals). (domain expert)

C.2 Evaluation protocol

- Preparation material
  - Video: a video showing the MW design environment: AutoCAD, visualization, simulation
  - Documentation: paper explaining the method and the experiment
- Interview: a one hour interview in person or through Skype

C.3 Interview questions

The general questions of the interview

1. What are your past experiences in designing container terminal?
2. What are your past experiences in using modeling and simulation to design logistics systems?
3. What are your past experiences in designing in a multi-stakeholder environment?
4. How many multi-stakeholder design projects using M&S did you carry out in the last 5 years?
5. What are some perceived problems in your current organization/projects that could be tackled using the multiple worlds **method**?
6. What are some perceived problems in your current organization/projects that could be tackled using the multiple worlds **environment**?
7. What benefits or drawbacks do you see in using the **method** to support collaboration with multiple stakeholders in your current organization or project?
8. What benefits or drawbacks do you see in using the **environment** to support collaboration with multiple stakeholders in your current organization or project?
9. What benefits or drawbacks to you see in using the **method** to explore the design space using modeling and simulation in your current organization or project?
10. What benefits or drawbacks to you see in using the **environment** to explore the design space using modeling and simulation in your current organization or project?

11. From a scale from one to five, going from highly unlikely to highly likely, would you propose an implementation of this method for one of your following projects? Could you explain the rating?

Appendix D

Naming convention of the components used to design automated container terminals

Table D.1 contains the actual name, the CAD block name, and the description of the components used to design an automated container terminal. Several blocks and all layers are only present for visualization purposes, therefore ignored by the model builder of simulation models: this is noted in the last column.

Name	Block name	Description	Only visualization
Quay Crane	VT_QC_01	Super Post Panamax with 30m crane guage	no
RMG	VT_RMG_01	8-wide, 1 over 5 high rail mounted gantry crane with a 25.5m crane gauge	no
Shuttle Carriers	VT_SC_01	1+1 over Shuttle Carrier	no
Lift-AGV	VT_AGV_01	Lifting AGV without container	no
Lift-AGV	VT_AGV_02	Lifting AGV with 40'container	yes
Vessel	VT_VESSEL_01	M type (Length: 295m, width: 32m) without container	no
Vessel	VT_VESSEL_02_A	E type (Length: 397.5m, width: 56.5m) container on it	yes
Vessel	VT_VESSEL_02_B	E type (Length: 397.5m, width: 56.5m) without container	yes
Vessel	VT_VESSEL_03_A	Feeder vessel	yes
Hatch Covers	VT_HATCHCOVER_03	Hatch Cover Stacking 3 high	yes
Container stack	VT_STACK_01	Dry container stacking block	no
Transfer point - waterside	VT_TPW_01	RMG waterside transfer point, for SC operation	no
Transfer point - landside	VT_TPL_01	RMG landside transfer point, for truck exchange	no
Apron	VT_APRON_1ST_01	Apron image, first 3m deep	yes
Apron	VT_APRON_2ND_01	Apron image, under the quay cranes for Shuttle carrier operation, 30.48m deep	yes
Apron	VT_APRON_3RD_01	Apron image, back reach for Shuttle carrier operation, 31m deep	no
Lighting Tower	VT_LT_01	Lighting Tower 30m high	yes
Lighting Tower	VT_LT_02	Lighting Tower 50m high	yes
Administration Building	VT_ADMIN_01	Building image of Virginia terminal (36.6m x 37.5m)	yes
Workshop	VT_WS_01	Tall workshop unit for SC (13.5m x 37.5m)	yes
Workshop	VT_WS_02	Short workshop unit for SC (13.5m x 37.5m)	yes
Gate	VT_GT_01	Gate unit with 4 lanes (21m x 28 m)	yes
Closed Gate	VT_CGT_01	OCR housing unit with 1 lane (7m x 12m)	yes
Jersey Barrier	VT_JB	1 pair of Jersey Barriers	yes
Other Object	VT_OB_xxxx	Sufix defines the height of the object (e.g. VT_OB_10000 : object has height of 10m)	yes
Boundary	VT_BOUNDARY	Paved area. This object defines the oeprational area of the terminal	no
Boundary	VT_ADDITIONAL_BOUNDARY_xx	Additional paved area, has numbering in the suffix. (e.g. VT_ADDITIONAL_BOUNDARY_01)	no
Surroundings	VT_SURROUNDINGS_xx	Land object, has numbering in the suffix (e.g. VT_SURROUNDINGS_01)	yes
Green Field	VT_GF_xx	Greenfield has numbering in the suffix (e.g. VT_GF_01)	yes
White lining	VT_LINE	Lines on this layer appear as white lines on the ground level	yes
Crane Rail	VT_RAIL	Lines on this layer appear as single rails in the virtual environment	yes
Road	VT_ROAD	Lines on this layer appear as 2-lane roads	yes

Table D.1: Naming convention used for tagging components.

Bibliography

[A. 11] A. P. Moller-Maersk Group. Company profile. `http://www.maersk.com`, January 5 2011.

[Ack73] R. L. Ackoff. Science in the systems age: Beyond IE, OR, and MS. *Operations Research*, vol. 21:pp. 661–671, 1973.

[Ack79] R. L. Ackoff. The future of operational research is past. *The Journal of the Operational Research Society*, vol. 30(2):pp. 93–104, 1979.

[Age04] H. Agerschou. *Planning and design of ports and marine terminals*. Thomas Telford, Ltd, United Kingdom, second edn., 2004.

[Akp05a] J. I. Akpan and R. J. Brooks. Experimental investigation of the impacts of virtual reality on discrete-event simulation. In M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, editors, *Proceedings of the 2005 Winter Simulation Conference*, pp. 1968–1975. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2005.

[Akp05b] J. I. Akpan and R. J. Brooks. Practitioners? Perception of the impacts of virtual reality on discrete-event simulation. In M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, editors, *Proceedings of the 2005 Winter Simulation Conference*, pp. 1976–1984. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2005.

[Ale64] C. Alexander. *Notes on the synthesis of form*. Harvard University Press, MA, USA, 1964.

[APM07] APM Terminals. APM Terminals Virginia container terminal opens; new technology to benefit safety, security, efficiency. `http://www.apmterminals.com/`

uploadedFiles/corporate/Media_Center/Press_Release_Articles/APMTVA\%20PRESS\%20RELEASE.pdf, September 7 2007.

[APM10] APM Terminals. Global port industry has passed "stress test" of 2009; lower cost and staying close to the customers has gained in importance. http://www.apmterminals.com/uploadedFiles/corporate/Media_Center/Press_Releases/100625_TransFin_2010_Conference.pdf, September 6 2010.

[APM11a] APM Terminals. Company profile. http://www.apmterminals.com, January 5 2011.

[APM11b] APM Terminals. Media kit 2011. http://www.apmterminals.com/uploadedFiles/corporate/Media_Center/110112_APM_Terminals_Media_Kit.pdf, January 2011.

[App10] Applied Materials. AutoMod. http://www.appliedmaterials.com/, November 10 2010.

[Aug04] J. M. Aughenbaugh and C. J. J. Paredis. The role and limitations of modeling and simulation in systems design. In *ASME 2004 International Mechanical Engineering Congress and Exposition*, pp. 13–22. American Society Of Mechanical Engineers, 2004.

[Aut10] Autodesk. ObjectARX. http://www.autodesk.com, April 8 2010.

[BA91] R. Banares-Alcantara. Representing the engineering design process: two hypotheses. *Computer-Aided Design*, vol. 23(9):pp. 595–603, 1991.

[Bai10] Baird Maritime. Shanghai tipped to overtake singapore port. http://www.bairdmaritime.com/index.php?option=com_content&view=article&id=7333:shanghai-tipped-to-overtake-singapore-port\&catid=68\&Itemid=59, September 6 2010.

[Bak09] P. Baker and M. Canessa. Warehouse design: a structured approach. *European Journal of Operational Research*, vol. 193(2):pp. 425–436, 2009.

[Bal98a] O. Balci. Verification, validation, and testing. In J. Banks, editor, *Handbook of simulation: principles, methodology, advances, applications, and practice*. John Wiley & Sons, NY, USA, 1998.

[Bal98b] O. Balci, A. I. Bertelrud, C. M. Esterbrook, and R. E. Nance. Visual simulation environment. In D. J. Medeiros, E. Watson, J. Carson, and M. S. Manivannan, editors, *Proceedings of the 1998 Winter Simulation Conference*, pp. 279–287. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 1998.

[Bal04] L. P. Baldwin, T. Eldabi, and R. J. Paul. Simulation in healthcare management: a soft approach (MAPIU). *Simulation Modelling Practice and Theory*, vol. 12:pp. 541–557, 2004.

[Ban00] J. Banks. Introduction to simulation. In J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, editors, *Proceedings of the 2000 Winter Simulation Conference*, pp. 9–16. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2000.

[Bar03] F. J. Barros. Dynamic structure multiparadigm modeling and simulation. *ACM Transaction on Modeling Computer Simulation*, vol. 13(3):pp. 259–275, 2003.

[BBC08] BBC News. Baggage halted at new £4.3bn T5. `http://news.bbc.co.uk/2/hi/uk_news/7317352.stm`, March 28 2008.

[BBC09] BBC News. Uk railways 'must boost capacity'. `http://news.bbc.co.uk/2/hi/uk_news/8071249.stm`, May 28 2009.

[Bec07] J. Becker and B. Niehaves. Epistemological perspectives on IS research: a framework for analysing and systematizing epistemological assumptions. *Information Systems Journal*, vol. 17(2):pp. 197–214, 2007.

[Ben98] J. Bennett and D. Worthington. An example of a good but partially successful OR engagement: improving outpatient clinic operations. *Interfaces*, vol. 28:pp. 56–69, 1998.

[Ber50] L. von Bertalanffy. An outline of general system theory. *The British Journal for the Philosophy of Science*, vol. 1(2):pp. 134–165, 1950.

[Boe88] B. W. Boehm. A spiral model of software development and enhancement. *Computer*, vol. 21(5):pp. 61–72, 1988.

[Bor75] I. Borovits and P. Ein-Dor. Computer simulation of a seaport container terminal. *Simulation*, vol. 25(2):pp. 141–144, 1975.

[Bri96] S. Brinkkemper. Method engineering: engineering of information systems development methods and tools. *Information and Software Technology*, vol. 38:pp. 275–280, 1996.

[Bri09] R. O. Briggs, J. D. Murphy, T. F. Carlisle, and A. J. Davis. Predicting change: a study of the value frequency model for change of practice. In *Proceedings of the 42nd Hawaii International Conference on System Sciences*. 2009.

[Bru00] H. de Bruijn and E. ten Heuvelhof. *Networks and decision making*. Lemma, the Netherlands, 2000.

[Bru08] H. de Bruijn and E. ten Heuvelhof. *Management of networks: on multi-actor decision making*. Routledge, United Kingdom, 2008.

[Bru09] H. de Bruijn and P. M. Herder. System and actor perspectives on sociotechnical systems. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 39:pp. 981–992, 2009.

[Bud92] R. Budde, K. Kautz, K. Kuhlenkamp, and H. Zullighoven. What is prototyping? *Information Technology & People*, vol. 6(2):pp. 89–95, 1992.

[Cap06] F. Caputo, G. di Gironimo, and A. Marzano. A structured approach to simulate manufacturing systems in virtual environment. In *Proceedings of the XVIII Congreso Internacional de Ingenieria Grafica*. INGEGRAF, Spain, 2006.

[Car04] J. Carson II. Introduction to modeling and simulation. In R. G. Ingalis, M. D. Rossetti, J. S. Smith, and B. A. Peters, editors, *Proceedings of the 2004 Winter Simulation Conference*, pp. 9–16. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2004.

[Cha96] F. Chance, J. Robinson, and J. Fowler. Supporting manufacturing with simulation: model design, development, and deployment. In J. M. Charnes, D. J. Morrice, D. T. Brunner, and J. J. Swain, editors, *Proceedings of the 1996 Winter Simulation Conference*, pp. 114–121. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 1996.

[Che78] P. B. Checkland. The origins and nature of hard systems thinking. *Journal of Applied Systems Analysis*, vol. 5:pp. 99–110, 1978.

[Che81a] P. B. Checkland. *Soft systems methodology : a 30-year retrospective*. John Wiley & Sons, NJ, USA, 1981.

[Che81b] P. B. Checkland. *Systems thinking, systems practice*. John Wiley & Sons, NJ, USA, 1981.

[Che04] W. Chen and R. Hirschheim. A paradigmatic and methodological examination of information systems research from 1991 to 2001. *Information Systems Journal*, vol. 14(3):pp. 197–235, 2004.

[Che06] P. B. Checkland. *Learning for action: a short definitive account of soft systems methodology, and its use for practitioners, teachers and students*. John Wiley & Sons, NJ, USA, 2006.

[Che10] P. B. Checkland and J. Poulter. Soft systems methodology. In M. Reynolds and S. Holwell, editors, *Systems approaches to managing change: a practical guide*. Springer, United Kingdom, 2010.

[Chu71] C. W. Churchman. *The design of inquiring systems: basic concepts of systems and organizations*. Basic Books, NY, USA, 1971.

[Cla95] T. D. J. Clark, R. W. Zmud, and G. E. McCray. The outsourcing of information services: transforming the nature of business in the information industry. *Journal of Information Technology*, vol. 10:pp. 221–237, 1995.

[Cou10] Council of Supply Chain Management Professionals. CSCMP supply chain management definitions. `http://cscmp.org/aboutcscmp/definitions.asp`, November 11 2010.

[Cre97] J. W. Creswell. *Qualitative inquiry and research design: choosing among five traditions*. Sage Publications, Inc., CA, USA, 1997.

[Cro93] N. Cross. Science and design methodology: a review. *Research in Engineering Design*, vol. 5(2):pp. 63–69, 1993.

[Dan08] W. Dangelmaier, M. Fischer, D. Huber, C. Laroque, and T. Süß. Aggregated 3d-visualization of a distributed simulation experiment of a queuing system. In S. J. Mason, R. Hill, L. Moench, and O. Rose, editors, *Proceedings of the 2008 Conference on Winter Simulation Conference*, pp. 2012–2020. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2008.

[Dav89] F. D. Davis. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, vol. 13:pp. 319–340, 1989.

[Dep10] Department of Defense. Systems Engineering. `http://www.acq.osd.mil/se/`, September 6 2010.

[Der09] T. Derksen. *Modelling modes of operation: a DSS and a decision making process on the selection of the mode of operation for APM Terminals*. Master's thesis, Delft University of Technology, the Netherlands, 2009.

[Dic68] G. W. Dickson. Management information-decision systems: a new era ahead? *Business Horizons*, vol. 11(6):pp. 17–26, 1968.

[Die83] G. E. Dieter. *Engineering design: a materials and process approach*. McGraw-Hill, NY, USA, 1983.

[Dob01] P. Dobson. The philosophy of critical realism: an opportunity for information systems research. *Information Systems Frontiers*, vol. 3(2):pp. 199–210, 2001.

[Dob02] P. Dobson. Critical realism and information systems research: why bother with philosophy. *Information Research*, vol. 7(2), 2002.

[Dob04] P. Dobson and P. Love. Realist and postmodernist perspectives on information systems research: points of connection. *Australasian Journal of Information Systems*, vol. 12(1), 2004.

[Don95] T. Donaldson and L. E. Preston. The stakeholder theory of the corporation: concepts, evidence, and implications. *The Academy of Management Review*, vol. 20(1):pp. 65–91, 1995.

[Dub03] L. Dube and G. Pare. Riger in information systems positivist case research: current practices, trends, and recommendations. *MIS Quarterly*, vol. 27(4):pp. 597–635, 2003.

[Dun72] F. Dunford. Basis for simulation model of container terminal. *Journal of the Transportation Engineering Division*, vol. 98(3):pp. 607–615, 1972.

[Dwa95] S. Dwarakanatha and K. M. Wallacea. Decision-making in engineering design: observations from design experiments. *Journal of Engineering Design*, vol. 6(3):pp. 191–206, 1995.

[Dym05] C. L. Dym, A. M. Agogino, D. D. Frey, and L. J. Leifer. Engineering design thinking, teaching, and learning. *Journal of Engineering Education*, vol. 94:pp. 103–120, 2005.

[Ens10] B. Enserink, J. F. M. Koppenjan, W. A. H. Thissen, D. P. Kamps, and G. Bekebrede. *Analyse van complexe omgevingen.* Delft University of Technology, 2010.

[Est08] J. A. Estefan. Survey of model-based systems engineering methodologies. Tech. rep., INCOSE, CA, USA, 2008.

[Eur10] Europe Container Terminals. Rotterdam Terminals. `http://www.ect.nl/`, September 6 2010.

[Eur11a] Europe Container Terminals. Delta terminal. `http://www.ect.nl/Terminals/rotterdamterminals/deltaterminal/Pages/default.aspx`, January 8 2011.

[Eur11b] Europe Container Terminals. Euromax terminal. `http://www.ect.nl/Terminals/rotterdamterminals/euromaxterminal/Pages/default.aspx`, January 8 2011.

[Fal99] D. J. Falconer and D. R. Mackay. Ontological problems of pluralist research methodologies. In *Proceedings of the Fourth 1999 Americas Conference on Information Systems*, pp. 624–626. 1999.

[Far09] J. Farré. *Modelling processes in the operation of an automated container terminal.* Master's thesis, Universitat Politècnica de Catalunya, Spain, 2009.

[Flo90] R. L. Flood. *Liberating system theory.* Plenum Publishing Corporation, NY, USA, 1990.

[For92] K. Forsberg and H. Mooz. The relationship of systems engineering to the project cycle. *Engineering Management Journal*, vol. 4(3):pp. 36–43, 1992.

[Fre91] J. H. Frey and A. Fontana. The group interview in social research. *The Social Science Journal*, vol. 28(2):pp. 175–187, 1991.

[Fri97] C. P. Friedman and J. C. Wyatt. *Evaluation methods in medical informatics.* Springer-Verlag, Germany, 1997.

[Fry95] C. Frydman, N. Giambiasi, G. Schmitt, and M. Le Goc. An intelligent model for assisted process control. In *IEEE International Conference on Systems, Man and Cybernetics*, pp. 3352–3357. IEEE Computer Society Press, 1995.

[Fum11] M. Fumarola and C. Versteegt. Supporting automated container terminal design processes with 3D virtual environments. In H. Yand and S. Yuen, editors, *Handbook of research on practices and outcomes in virtual worlds and environment.* IGI Global, PA, USA, 2011.

[Gal06] C. S. Galbraith, S. B. Ehrlich, and A. F. DeNoble. Predicting technology success: identifying key predictors and assessing expert evaluation for advanced technologies. *The Journal of Technology Transfer*, vol. 31:pp. 673–684, 2006.

[Gee11] H. Geerlings and R. van Duin. A new method for assessing CO_2-emissions from container terminals: a promising approach applied in rotterdam. *Journal of Cleaner production*, vol. 19(6–7):pp. 657–666, 2011.

[Gig91] J. P. van Gigch. *System design modeling and metamodeling (the language of science).* Plenum Publishing Corporation, NY, USA, 1991.

[Gir03] M. Girod, A. C. Elliott, N. D. Burns, and I. C. Wright. Decision making in conceptual engineering design: an empirical investigation. *Journal of Engineering Manufacture*, vol. 217(9):pp. 1215–1228, 2003.

[Goe95] V. Goel. *Sketches of thought.* The MIT Press, MA, USA, 1995.

[Gon10] R. A. Gonzalez. *A framework for ICT-supported coordination in crisis response.* Ph.D. thesis, Delft University of Technology, the Netherlands, 2010.

[Goo57] H. H. Goode and R. E. Machol. *System engineering: an introduction to the design of large-scale systems.* McGraw-Hill, NY, USA, 1957.

[Goo06] A. S. Goodman and M. Hastak. *Infrastructure planning handbook: planning, engineering, and economics*. McGraw-Hill Professional, NY, USA, 2006.

[Gra71] A. Gramsci. *Prison notebooks*. Lawrence & Wishart, United Kingdom, 1971.

[Gre06] S. Gregor. The nature of theory in information systems. *MIS Quarterly*, vol. 30(3):pp. 611–642, 2006.

[Gre07] S. Gregor and D. Jones. The anatomy of a design theory. *Journal of the Association for Information Systems*, vol. 8(5):pp. 312–335, 2007.

[Hal62] A. D. Hall. *A methodology for systems engineering*. Van Nostrand Reinhold, NY, USA, 1962.

[Haz98] G. A. Hazelrigg. A framework for decision-based engineering design. *Journal of Mechanical Design*, vol. 120:pp. 653–659, 1998.

[Haz99] G. A. Hazelrigg. An axiomatic framework for engineering design. *Systems Engineering*, vol. 121:pp. 342–348, 1999.

[Hen07] M. den Hengst, G. J. de Vreede, and R. Maghnouji. Using soft OR principles for collaborative simulation: a case study in the Dutch airline industry. *Journal of the Operational Research Society*, vol. 58:pp. 669–682, 2007.

[Her05] L. M. Hermans. *Actor analysis for water resources management: putting the promise into practice*. Ph.D. thesis, Delft University of Technology, the Netherlands, 2005.

[Hev04] A. R. Hevner, S. T. March, J. Park, and S. Ram. Design science in information systems research. *MIS Quarterly*, vol. 28(1):pp. 75–105, 2004.

[Hev07] A. H. Hevner. A three cycle view of design science research. *Scandinavian Journal of Information Systems*, vol. 19(2):pp. 87–92, 2007.

[Hev10] A. R. Hevner and S. Chatterjee. *Design research in information systems: theory and practice*. Springer-Verlag, Germany, 2010.

[HHL11] HHLA. Container Terminal Altenwerder. `http://www.hhla.de/Altenwerder-CTA.64.0.html`, January 8 2011.

[Hil02] S. P. J. Hilkens. *Pre-Stacking at a large transshipment terminal: trade-off between the costs of segregation and quay crane productivity*. Master's thesis, Technical University Eindhoven, the Netherlands, 2002.

[Hir95] R. Hirschheim, H. K. Klein, and K. Lyytinen. *Information systems development and data modeling: conceptual and philosophical foundations*. Cambridge University Press, United Kingdom, 1995.

[Hir03] R. Hirschheim and H. K. Klein. Crisis in the IS field? A critical reflection on the state of the discipline. *Journal of the Association for Information Systems*, vol. 4(10):pp. 237–293, 2003.

[Hit08] D. K. Hitchins. *Systems engineering: a 21st century systems methodology*. John Wiley & Sons Inc, NY, USA, 2008.

[Hlu93] V. Hlupic. *Simulation modeling software approaches to manufacturing problems*. Ph.D. thesis, London School of Economics, United Kingdom, 1993.

[Hof92] A. H. M. ter Hofstede and T. P. van der Weide. Formalisation of techniques: chopping down the methodology jungle. *Information and Software Technology*, vol. 34(1):pp. 57–65, 1992.

[Hoo91] S. P. Hoover and J. R. Rinderle. Models and abstractions in design. *Design Studies*, vol. 12:pp. 237–245, 1991.

[Hor10] R. Horonjeff and F. McKelvey. *Planning and design of airports*. McGraw-Hill Professional, NY, USA, fourth edn., 2010.

[Hou07] S. P. A. van Houten. *A suite for developing and using business games: supporting supply chain business games in a distributed context*. Ph.D. thesis, Delft University of Technology, the Netherlands, 2007.

[Hu05a] X. Hu, B. Zeigler, and S. Mittal. Variable structure in DEVS component-based modeling and simulation. *Simulation*, vol. 81(2):pp. 91–102, 2005.

[Hu05b] X. Hu and B. P. Zeigler. Model continuity in the design of dynamic distributed real-time systems. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 35(6):pp. 867–878, 2005.

[Hu08] H. Hu. *Choosing the optimal mode of operation for marine container terminals*. Master's thesis, Delft University of Technology, the Netherlands, 2008.

[Hub88] V. Hubka and W. E. Eder. *Theory of technical systems*. Springer-Verlag, Germany, 1988.

[Hut11] Hutchison Port Holdings Group. London thamesport. `http://www.londonthamesport.co.uk`, January 8 2011.

[Iiv07] J. Iivari. A paradigmatic analysis of information systems as a design science. *Scandinavian Journal of Information Systems*, vol. 19(2):pp. 39–64, 2007.

[INC10] INCOSE. International Council on Systems Engineering. `http://www.incose.org/`, September 6 2010.

[Jac84] M. C. Jackson and P. Keys. Towards a system of systems methodologies. *The Journal of the Operational Research Society*, vol. 35:pp. 473–486, 1984.

[Jac02] P. H. M. Jacobs, N. A. Lang, and A. Verbraeck. D-SOL: a distributed java based discrete event simulation architecture. In E. Yücesan, C. H. Chen, J. L. Snowdon, , and J. M. Charnes, editors, *Proceedings of the 2002 Winter Simulation Conference*, pp. 793–800. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2002.

[Jac05] P. H. M. Jacobs. *The D-SOL simulation suite: enabling multi-formalism simulation in a distributed context*. Ph.D. thesis, Delft University of Technology, the Netherlands, 2005.

[Jan08] A. Jansen, J. Bosch, and P. Avgeriou. Documenting after the fact: recovering architectural design decisions. *Journal of Systems and Software*, vol. 81(4):pp. 536–557, 2008.

[Jen69] G. M. Jenkins. The systems approach. In J. Beishon and G. P. Peters, editors, *Systems behaviour*. Harper & Row, United Kingdom, second edn., 1969.

[Kam05] V. R. Kamat and J. C. Martinez. Dynamic 3D visualization of articulated construction equipment. *Journal of Computing in Civil Engineering*, vol. 19(4):pp. 356–368, 2005.

[Kan99] I. Kant. *Critique of pure reason (The Cambridge edition of the works of Immanuel Kant in translation)*. Cambridge University Press, United Kingdom, 1999.

[Kee08] P. G. W. Keen and H. G. Sol. *Decision enhancement services: rehearsing the future for decisions that matter*. IOS Press, the Netherlands, 2008.

[Kim02] K. H. Kim, S. J. Wang, Y. M. Park, C. H. Yang, and J. W. Bae. A simulation study on operation rules for automated container yards. In *Proceedings of the 7th Annual International Conference on Industrial Engineering*. 2002.

[Kin04] E. Kindler, T. Coudert, and P. Berruet. Component-based simulation for a reconfiguration study of transitic systems. *Simulation*, vol. 80:pp. 153–163, 2004.

[Kli69] G. J. Klir. *An approach to general systems theory*. Van Nostrand Reinhold, NY, USA, 1969.

[Kli85] G. J. Klir. *Architecture of systems complexity*. Sauders, NY, USA, 1985.

[Kot07] K. Kotiadis. Using soft systems methodology to determine the simulation study objectives. *Journal of Simulation*, vol. 1:pp. 215–222, 2007.

[Law00a] A. M. Law and W. D. Kelton. *Simulation Modelling and Analysis*. McGraw-Hill Higher Education, NY, USA, 2000.

[Law00b] B. Lawson. *Design strategies*, chap. 11, pp. 185–204. Architectural Press, NY, USA, 2000.

[Leh95] B. Lehaney and V. Hlupic. Simulation modeling for resource allocation and planning in the health sector. *Journal of the Royal Society of Health*, vol. 115:pp. 382–385, 1995.

[Leh96] B. Lehaney and R. J. Paul. The use of soft systems methodology in the development of a simulation of outpatient services at Watford General Hospital. *Journal of the Operational Research Society*, vol. 47:pp. 864–870, 1996.

[Leh08] B. Lehaney, D. Malindzak, and Z. Khan. Simulation modelling for problem understanding: a case study in the East Slovakia coal industry. *Journal of the Operational Research Society*, vol. 59:pp. 1332–1339, 2008.

[Lei09] T. E. van der Lei. *Relating actor analysis methods to policy problems*. Ph.D. thesis, Delft University of Technology, the Netherlands, 2009.

[Lei10] T. E. van der Lei, G. L. Kolfschoten, and P. J. Beers. Complexity in multi-actor system research: towards a meta-analysis of recent studies. *Journal Design Research*, vol. 8(4):pp. 317–342, 2010.

[Leo01] C. Y. Leong. *Simulation study of dynamic AGV-container job deployment scheme*. Master's thesis, National University of Singapore, Singapore, 2001.

[Lim03] J. K. Lim, K. H. Kim, K. Yoshimoto, and J. H. Lee. A dispatching method for automated guided vehicles by using a bidding concept. *OR Spectrum*, vol. 25:pp. 25–44, 2003.

[Lin02] T. R. Lindlof and B. C. Taylor. *Qualitative communication research methods*. SAGE Publications, CA, USA, second edn., 2002.

[Liu04] C. I. Liua, H. Julab, K. Vukadinovicc, and P. Ioannou. Automated guided vehicle system for two container yard layouts. *Transportation Research Part C: Emerging Technologies*, vol. 12:pp. 349–368, 2004.

[Mac10] P. Machira. Improve port services, Kawambwa tells Tanzania ports authority. `http://www.ippmedia.com/frontend/index.php?l=20623`, September 5 2010.

[Mar95] S. T. March and G. F. Smith. Design and natural science research on information technology. *Decision Support Systems*, vol. 15(4):pp. 251–266, 1995.

[Mar02] M. L. Markus, A. Majchrzak, and L. Gasser. A design theory for systems that support emergent knowledge processes. *MIS Quarterly*, vol. 26(3):pp. 179–212, 2002.

[Mea09] R. Meade. APM Terminals gears up for next phase of Maasvlakte 2 project. *Lloyd's list*, vol. 60(42), 2009.

[Mey08] W. K. Meyers, G. P. J. Dijkema, M. P. C. Weijnen, and K. Brown. Infrastructure master planning and infrastructure interaction: the case of south east queensland. In *Proceedings of The Twelfth Annual Conference of the International Research Society for Public Management*, pp. 26–28. 2008.

[Min94] H. Mintzberg. *The rise and fall of strategic planning.* Prentice Hall, NJ, USA, 1994.

[Min03] J. Mingers. A classification of the philosophical assumptions of management science methods. *Journal of the Operational Research Society*, vol. 54(6):pp. 559–570, 2003.

[Min04a] J. Mingers. Realizing information systems: critical realism as an underpinning philosophy for information systems. *Information and Organization*, vol. 14(2):pp. 87–103, 2004.

[Min04b] J. Mingers and J. Rosenhead. Problem structuring methods in action. *European Journal of Operational Research*, vol. 152:pp. 530–554, 2004.

[Min10] J. Mingers and L. White. A review of the recent contribution of systems thinking to operational research and management science. *European Journal of Operational Research*, vol. 207(3):pp. 1147–1161, 2010.

[Moc06] G. M. Mocko. *A knowledge framework for integrating multiple perspectives in decision-centric design.* Ph.D. thesis, Georgia Institute of Technology, GA, USA, 2006.

[Mun02] I. Munro and J. Mingers. The use of multimethodology in practice: results of a survey of practitioners. *Journal of the Operational Research Society*, vol. 53:pp. 369–378, 2002.

[Nam01] K. C. Nam. Evaluation of handling systems for container terminals. *Journal of Waterway, Port, Coastal and Ocean Engineering*, vol. 127(3):pp. 171–175, 2001.

[Nan81] R. E. Nance. The time and state relationships in simulation modeling. *Communications of the ACM*, vol. 24(4):pp. 173–179, 1981.

[Nan83] R. E. Nance. A tutorial view of simulation model development. In S. Roberts, J. Banks, and B. Schneiser, editors, *Proceedings of the 1983 Winter Simulation Conference*, pp. 325–331. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 1983.

[Nan97] J. Nandhakumar and M. Jones. Too close for comfort? distance and engagement in interpretive information systems research. *Information Systems Journal*, vol. 7(2):pp. 109–131, 1997.

[Nat07] National Aeronautics and Space Administration. *Systems engineering handbook*. NASA Center for AeroSpace Information, MD, USA, 2007.

[Neu08] R. de Neufville. Low-cost airports for low-cost airlines: flexible design to manage the risks. *Transportation Planning and Technology*, vol. 31(1):pp. 35–68, 2008.

[Nie07] B. Niehaves. On epistemological pluralism in design science. *Scandinavian Journal of Information Systems*, vol. 19(2):pp. 99–110, 2007.

[Nun90] J. F. J. Nunamaker and M. Chen. Systems development in information systems research. In *Proceedings of the Twenty-Third Annual Hawaii International Conference on System Sciences*, pp. 631–640. 1990.

[Nun91] J. F. J. Nunamaker, M. Chen, and T. D. M. Purdin. Systems development in information systems research. *Journal of Management Information Systems*, vol. 7(3):pp. 89–106, 1991.

[Ö71] T. I. Ören. *General systems theory implementor*. Ph.D. thesis, University of Arizona, AZ, USA, 1971.

[Ö06] T. I. Ören and L. Yilmaz. Synergy of systems engineering and modeling and simulation. In *Proceedings of the 2006 International Conference on Modeling and Simulation - Methodology, Tools, Software Applications (M&S MTSA)*, pp. 10–17. 2006.

[OMG10] OMG. OMG systems modeling language. `http://www.omgsysml.org/`, November 10 2010.

[Ora10] Oracle Java. Object serialization. `http://download.oracle.com/javase/1.4.2/docs/guide/serialization/index.html`, February 21 2010.

[Orl91] W. J. Orlikowski and J. J. Baroudi. Studying information technology in organisations: research approaches and assumptions. *Information Systems Research*, vol. 2:pp. 1–28, 1991.

[Ose04] N. Oses, M. Pidd, and R. J. Brooks. Critical issues in the development of component-based discrete simulation. *Simulation Modelling Practice and Theory*, vol. 12(7-8):pp. 495–514, 2004.

[Ott95] K. N. Otto. Measurement methods for product evaluation. *Research in engineering design*, vol. 7(2):pp. 86–101, 1995.

[Oya09] V. Oya. *Analysis of ICT solutions integration for tracking purposes in container terminal management and operation.* Master's thesis, Universitat Politècnica de Catalunya, Spain, 2009.

[Pah95] G. Pahl and W. Beitz. *Engineering design: a systematic approach.* Springer-Verlag, Germany, second edn., 1995.

[Par01] C. J. J. Paredis, A. Diaz-Calderon, R. Sinha, and P. K. Khosla. Composable models for simulation-based design. *Engineering with Computers*, vol. 17(2):pp. 112–128, 2001.

[Pat11] Patrick. Brisbane Terminal. `http://www.patrick.com.au/index.php?id=60`, January 8 2011.

[Pea07] R. S. Peak, R. M. Burkhart, S. A. Friedenthal, M. W. Wilson, M. Bajaj, and I. Kim. Simulation-based design using SysML–part 2: celebrating diversity by example. In *INCOSE International Symposium.* 2007.

[Peg07] C. D. Pegden. SIMIO: a new simulation system based on intelligent objects. In S. G. Henderson, B. Biller, M. H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, editors, *Proceedings of the 2007 Winter Simulation Conference*, pp. 2293–2300. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2007.

[Pid07] M. Pidd. Making sure you tackle the right problem: linking hard and soft methods in simulation practice. In S. G. Henderson, B. Biller, M. H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, editors, *Proceedings of the 2007 Winter Simulation Conference*, pp. 195–204. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2007.

[Pie01] M. A. Piera and A. Guasch. Reusable simulation harbour models to improve terminal logistics performance. In *Proceedings of the International Workshop on Harbour, Maritime and Multimodal Logistics Modeling and Simulation.* 2001.

[Pie05] B.-J. Pielage. *Conceptual design of automated freight transport systems.* Ph.D. thesis, Delft University of Technology, the Netherlands, 2005.

[Pii09] K. Piirainen, G. L. Kolfschoten, and S. G. Lukosch. Unraveling challenges in collaborative design: a literature study. *Lecture Notes in Computer Science*, vol. 5784:pp. 247–261, 2009.

[Pop72] K. R. Popper. *Objective knowledge: an evolutionary approach*. Oxford University Press, United Kingdom, 1972.

[Pot88] C. Potts and G. Bruns. Recording the reasons for design decisions. In *Proceedings of the 10th international conference on Software engineering*, ICSE '88, pp. 418–427. IEEE Computer Society Press, 1988.

[Pru10] E. Pruyt. Multi-actor systems and ethics. *International Transactions in Operational Research*, vol. 17(4):pp. 507–520, 2010.

[Put81] H. Putnam. *Reason, truth and history*. Cambridge University Press, United Kingdom, 1981.

[Qin03] S. F. Qin, R. Harrisonb, A. A. Westb, I. N. Jordanovc, and D. K. Wrighta. Conceptual design of industrial systems: an approach to support collaboration. *Computers in Industry*, vol. 50:pp. 153–164, 2003.

[Reu10] Reuters. APM Terminals sees 2010 container port volume growth. `http://uk.reuters.com/article/2010/06/23/apm-terminals-idUKLDE65M1XS20100623`, September 6 2010.

[Ric07] H. Richardson and B. Robinson. The mysterious case of the missing paradigm: a review of critical information systems research 1991-2001. *Information Systems Journal*, vol. 17(3):pp. 251–270, 2007.

[Rij99] J. Rijsenbrij. Balanced scales!? Inaugural speech, 1999.

[Rob01] S. Robinson. Soft with a hard centre: discrete-event simulation in facilitation. *The Journal of the Operational Research Society*, vol. 52:pp. 905–915, 2001.

[Rob02] S. Robinson. Modes of simulation practice: approaches to business and military simulation. *Simulation Modelling Practice and Theory*, vol. 10(8):pp. 513–523, 2002.

[Rob04] S. Robinson, R. E. Nance, R. J. Paul, M. Pidd, and S. J. E. Taylor. Simulation model reuse: definitions, benefits and obstacles. *Simulation Modelling Practice and Theory*, vol. 12:pp. 479–494, 2004.

[Rob07] S. Robinson. PSMs: looking in from the outside. *Journal of the Operational Research Society*, vol. 58:pp. 689–691, 2007.

[Roc10] Rockwell. Arena. `http://www.arenasimulation.com/`, November 10 2010.

[Roh00] M. W. Rohrer. Seeing is believing: The importance of visualization in manufacturing simulation. In J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, editors, *Proceedings of the 2000 Winter Simulation Conference*, pp. 1211–1216. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2000.

[Ros01] J. Rosenhead and J. Mingers. *Rational analysis for a problematic world revisited: problem structuring methods for complexity, uncertainty and conflict.* John Wiley & Sons, NJ, USA, 2001.

[Roy70] W. Royce. Managing the development of large software systems. In *Proceedings of IEEE WESCON*, pp. 1–9. 1970.

[Saa04] Y. A. Saanen. *An approach for designing robotized marine container terminals.* Ph.D. thesis, Delft University of Technology, the Netherlands, 2004.

[Sag00] A. P. Sage and J. E. J. Armstrong. *Introduction to systems engineering.* John Wiley & Sons Inc, NY, USA, 2000.

[San98] M. Sandelowski. The call to experts in qualitative research. *Research in nursing & health*, vol. 21:pp. 467–471, 1998.

[Sar06] R. G. Sargent, R. E. Nance, C. Overstreet, S. Robinson, and J. Talbot. The simulation project life-cycle: models and realities. In L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, editors, *Proceedings of the 2006 Winter Simulation Conference*, pp. 863–871. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2006.

[Sch70] E. Schmeltzer and R. A. Peavy. Prospects and problems of the container revolution. *Journal of Maritime Law and Commerce*, vol. 203:p. 203, 1970.

[Sch95] L. C. Schmidt and J. Cagan. Recursive annealing: a computational model for machine design. *Research in Engineering Design*, vol. 7:pp. 102–125, 1995.

[Sea95] J. R. Searle. *The Construction of Social Reality.* The Free Press, NY, USA, 1995.

[Sec09] M. D. Seck and A. Verbraeck. DEVS in DSOL: adding DEVS operational semantics to a generic event-scheduling simulation environment. In *Proceedings of SCS 2009 Summer Computer Simulation Conference*. The Society for Computer Simulation International, July 2009.

[Sha98] R. E. Shannon. Introduction to the art and science of simulation. In D. J. Medeiros, E. F. Watson, J. S. Carson, and M. S. Manivannan, editors, *Proceedings of the 1998 Winter Simulation Conference*, pp. 7–14. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 1998.

[Sha09] M. Shahin, L. Peng, and M. R. Khayyambashi. Architectural design decision: Existing models and tools. In *Software Architecture, 2009 European Conference on Software Architecture. WICSA/ECSA 2009. Joint Working IEEE/IFIP Conference on*, pp. 293 –296. 2009.

[Shi97] F. M. Shipman III and R. J. McCall. Decision making in conceptual engineering design: an empirical investigation. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing*, vol. 11(2):pp. 141–154, 1997.

[Shu07] F. Shu, W. Mi, and Z. Xu. The information sharing platform for port container terminal logistics using virtual reality. In *Proceedings of the 2007 IEEE International conference on automation and logistics*, pp. 2570–2575. 2007.

[Sia07] K. Siau and M. Rossi. Evaluation techniques for systems analysis and design modelling methods: a review and comparative analysis. *Information Systems Journal*, vol. 20:pp. 1–20, 2007.

[Sim96] H. A. Simon. *The sciences of the artificial*. The MIT Press, MA, USA, third edn., 1996.

[Smi89] T. Smithers. Al-based design versus geometry-based design or why design cannot be supported by geometry alone. *Computer-Aided Design*, vol. 21(3):pp. 141–150, 1989.

[Sol82] H. G. Sol. *Simulation in information systems development*. Ph.D. thesis, Rijksuniversiteit Groningen, the Netherlands, 1982.

[Som04] I. Sommerville. *Software engineering*. Addison-Wesley, MA, USA, seventh edn., 2004.

[Ste95] J. L. Stein, J. L. Stein, and L. S. Louca. A component-based modeling approach for system design: theory and implementation. In *Proceedings of the 1995 International Conference on Bond Graph Modeling and Simulation*, pp. 109–115. 1995.

[Ste05] D. Steenken, S. Voss, and R. Stahlbock. Container terminal operation and operations research: a classification and literature review. In H. O. Gunther and K. H. Kim, editors, *Container terminals and automated transport systems*. Springer-Verlag, Germany, 2005.

[Ste09] A. J. Steenstra. *Mapping to manage: the design of a process to fully exploit the benefits of process mapping within APMT's trajectory towards container terminal automation*. Master's thesis, Delft University of Technology, the Netherlands, 2009.

[Str05] S. Strassburger, T. Schulze, M. Lemessi, and G. D. Rehn. Temporally parallel coupling of discrete simulation systems with virtual reality systems. In M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, editors, *Proceedings of the 2005 Winter Simulation Conference*, pp. 1949–1957. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2005.

[Tek10] C. Tekinay, M. Seck, M. Fumarola, and A. Verbraeck. A context-based multi-perspective modeling and simulation framework. In B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E.Yücesan, editors, *Proceedings of the 2010 Winter Simulation Conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2010.

[Tew02] T. Tewoldeberhan, A. Verbraeck, E. Valentin, and G. Bordonnet. An evaluation and selection methodology for discrete-event simulation software. In E. Yücesan, C. H. Chen, J. L. Snowdon, , and J. M. Charnes, editors, *Proceedings of the 2002 Winter Simulation Conference*, pp. 793–800. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2002.

[The09] The Boeing Company. EASY5. `http://www.boeing.com/assocproducts/easy5/`, April 3 2009.

[tra10] transportweekly. Rotterdam first half box traffic up 18pc. `http://www.transportweekly.com/pages/en/news/articles/74874/`, September 6 2010.

[Ull88] D. G. Ullman, T. G. Dietterich, and L. A. Stauffer. A model of the mechanical design process based on empirical data. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing*, vol. 2(1):pp. 33–52, 1988.

[Vah05] R. Vahidov. Design researcher's IS artifact: a representational framework. In *Proceedings of the 2006 Design Science Research in Information Systems and Technology*, pp. 24–25. 2005.

[Val05] E. Valentin and A. Verbraeck. Requirement for domain specific discrete event simulation environments. In M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, editors, *Proceedings of the 2005 Winter Simulation Conference*, pp. 654–663. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 2005.

[Van00] H. Vangheluwe. DEVS as a common denominator for multi-formalism hybrid systems modelling. In *IEEE International Symposium on Computer-Aided Control System Design*, pp. 129–134. IEEE Computer Society Press, 2000.

[Ven99] J. Venable and J. Travis. Using a group support system for the distributed application of soft systems methodology. In *Proceedings of the 10th Australasian conference on Information systems*. 1999.

[Ven06] J. Venable. A framework for design science research activities. In *Proceedings of the 2006 Information Resource management association conference*. 2006.

[Ver04a] A. Verbraeck. Component-based distributed simulations: the way forward? In *Proceedings of the 18th Workshop on Parallel and Distributed Computer Simulation*, pp. 141–148. 2004.

[Ver04b] C. Versteegt. *Holonic control for large scale automated logistics systems*. Ph.D. thesis, Delft University of Technology, the Netherlands, 2004.

[Ver09] A. Verbraeck, M. Seck, M. Fumarola, and C. Versteegt. *Validation simulation models Maasvlakte 2: methodology, findings, and recommendations*. Delft University of Technology, 2009. Internal document.

[Ver10] A. Verbraeck, M. Fumarola, M. Seck, and C. Versteegt. *Validation of the TTS Kollmorgen cassette AGV System for MV2*. Delft University of Technology, 2010. Internal document.

[Vis03] I. F. A. Vis and R. de Koster. Transshipment of containers at a container terminal: an overview. *European Journal of Operational Research*, vol. 147:pp. 1–16, 2003.

[Vis06] W. Visser. *The cognitive artifacts of designing*. Lawrence Erlbaum Associates, NJ, USA, 2006.

[Waa08] A. de Waal. *APMT MV2 Terminal: algorithms & rules simulated TOS*. TBA B.V., 2008. Internal document.

[Wad04] M. Wade and J. Hulland. Review: The resource-based view and information systems research: review, extension, and suggestions for future research. *MIS Quarterly*, vol. 28(1):pp. 107–142, 2004.

[Wai09] G. Wainer and Q. Liu. Tools for graphical specification and visualization of DEVS models. *Simulation*, vol. 85(3):pp. 131–158, 2009.

[Wal92] J. Walls, G. R. Widmeyer, and O. A. El Sawy. Building an information system design theory for vigilant EIS. *Information Systems Research*, vol. 3(1):pp. 36–59, 1992.

[Wal95a] G. Walsham. The emergence of interpretivism in IS research. *Information Systems Research*, vol. 6(4):pp. 376–394, 1995.

[Wal95b] G. Walsham. Interpretive case studies in IS research: nature and method. *European Journal of Information Systems*, vol. 4:pp. 74–81, 1995.

[Wal01] A. Wallace. Applicaiton of AI to AGV control - agent control of AGVs. *International Journal of Production Research*, vol. 39:pp. 709–726, 2001.

[Web04] R. Weber. The rhetoric of positivism versus interpretivism: a personal view. *MIS Quarterly*, vol. 28(1):pp. 3–7, 2004.

[Whi99] K. P. White. Systems design engineering. *Systems Engineering*, vol. 1:pp. 285–302, 1999.

[Why00] J. Whyte, N. Bouchlaghem, A. Thorpe, and R. McCaffer. From CAD to virtual reality: modelling approaches, data exchange and interactive 3D building design tools. *Automation in Construction*, vol. 10(1):pp. 43–55, 2000.

[Win09] R. Winter, A. Gericke, and T. Bucher. Method versus model - two sides of the same coin? *Lecture Notes in Business Information Processing*, vol. 34(1):pp. 1–15, 2009.

[Wys04] B. Wyssusek. Ontology and ontologies in information systems analysis and design: a critique. In *Proceedings of the Tenth Americas Conference on Information Systems 2004*, pp. 6–8. Institute of Electrical and Electronics Engineers, Inc., New York, United States, 2004.

[Xia05] W. Xia and G. Lee. Complexity of information systems development projects: conceptualization and measurement development. *Journal of Management Information Systems*, vol. 22(1):pp. 45–83, 2005.

[Yin02] R. K. Yin. *Case study research: design and methods*. SAGE Publications, CA, USA, third edn., 2002.

[Yun99] W. Y. Yun and Y. S. Choi. A simulation model for container-terminal operation analysis using an object-oriented approach. *International Journal of Production Economics*, vol. 59:pp. 221–230, 1999.

[Zei00] B. P. Zeigler and H. Praehofer. *Theory of Modeling and Simulation*. Academic Press, MA, USA, 2000.

[Zei05] B. P. Zeigler and S. Mittal. Enhancing DoDAF with a DEVS-based system lifecycle development process. In *Proceedings of the 2005 IEEE International Conference on Systems, Man and Cybernetics*, pp. 3244–3251. IEEE Computer Society Press, 2005.

[Zei07] B. P. Zeigler and P. E. Hammonds. *Modeling and simulation-based data engineering: introducing pragmatics into ontologies for net-centric information exchange*. Academic Press, MA, USA, 2007.

[Zia07] M. Zia, S. Mustafiz, H. Vangheluwe, and J. Kienzle. A modelling and simulation based process for dependable systems design. *Software and Systems Modeling*, vol. 6(4):pp. 437–451, 2007.

[Zij95] E. J. A. Zijderveld. *A structured terminal design method with a focus on rail container terminals*. Ph.D. thesis, Delft University of Technology, the Netherlands, 1995.

Summary

Multiple Worlds: a multi-actor simulation-based design method for logistics systems

Logistics systems have become essential to our current global economy. A substantial part of the total freight is transported using containers. Container terminals form the main hubs in which containers are transloaded from and to trains, ships, and trucks. New technologies are constantly being developed to increase the efficiency of container terminals. Novel advances include automating several parts of container terminals using computer-guided equipment. The design process of this type of container terminals is complex from an actor as well as system perspective. Container terminals are often multi-million projects that require the involvement of multiple actors: both internal in the company designing the terminal as external such as of policy makers, consultants, and customers. The internal actors, such as the one in charge of business development and the operational design, are highly involved in the process of designing a complex system that includes many interrelating subsystems, such as the storage area of a terminal, the sea side of the terminal and the terminal operating system that controls the automated equipment.

Simulation models are often been used to provide insight into a system's behavior. Simulation studies are usually carried out at the end of a design process without taking into consideration the multi-actor environment in which a system is being designed. However, the typical view of the use of simulation models as part of the hard systems approach is changing and current literature presents applications of simulation models from a soft systems perspective. The latter presents a learning process in which each actor becomes conscious of the other actors' perspectives. Combining simulation models in a soft systems approach presents a number of issues that we chose to address in our design method, which we call the "Multiple worlds" design method.

We formulated the objective of our research as follows: "To provide a method that uses simulation models to support multi-actor design processes of logistics systems". To satisfy this objective, we identified three research questions:

1. What are the current methods used for designing logistics systems in practice?

2. What are the issues in current multi-actor design methods that use simulation models?

3. How can a multi-actor design method that uses simulation models address the issues identified in RQ2?

With the first question, we aim at identifying the current way of working in business environments where simulation models are used to design logistics systems. The second question helps us finding existing theories on how to use simulation models to support multi-actor design processes. Finally, the third questions combines the finding of the first and second question to construct the "Multiple Worlds" design method.

To carry out this research, we took a critical realist ontological and interpretivist epistemological perspective. The former looks at our knowledge of reality as resulting from social conditioning, which means that it cannot be understood independently of the social actors involved in the knowledge derivation process. The latter states that facts and values are intertwined, thus both are involved in scientific knowledge. We argue in this dissertation that this stance helps us create a rich picture that encompasses the artifact and the actors that are influenced by the artifact. This brings us to our research methodology: in information systems research, a hegemonization is taking place in terms of research methodology. Design science research in information systems presents a three-cycled view on research, concentrating on relevance and rigor of a research project which results in the design of an artifact. Looking at our research questions, the first question is used to go through the relevance cycle (Chapter 2), the second to go through the rigor cycle (Chapter 3), and, finally, the third one to design the artifact (Chapter 4, Chapter 5, and Chapter 6).

To cover the first part of relevant cycle of our research, we carried out a case study at APM Terminals (Chapter 2). During this descriptive case study, we analyzed how design processes are carried out at APM Terminals, particularly focused on design processes of automated container terminals. On this regard, we have analyzed the cases of the APMT's Virginia (USA) terminal and their future terminal on Maasvlakte 2 in Rotterdam. During the case study, we focused on the system (the terminal as a physical entity, with its equipment and control system) and on the actor environment (the actors involved in the design process, which need to be supported during the process). The case study showed the complexity present in the design process of automated container terminals. The many decisions that need to be made incur multiple interactions in the technical system and between the actors involved in the process. Simulation models can capture the dynamicity of the technical system and by doing so, support the decision making process of the designers. We concluded that repositioning the use of simulation, by having it during the early stages of the design process, could make it possible to support the actors in rigorously assessing multiple design alternatives during the design process.

Throughout the rigor cycle, we took a close look at SE as conceived in the early fifties of the previous century and how it came to a split into hard and soft systems thinking. We studied how these different approaches are used to design systems and what issues are present in combining them into a multimethodological approach. We posed that we can address these issues and propose a design method that uses simulation models in a multi-actor environment. We sought appropriate methods and models that tackle each issue individually. Based on these methods and models, we identified four conceptual constructs that we could use to design our method: component based modeling, different levels of specification, structuring of alternatives, and participatory design. The resulting method, which we call the "Multiple Worlds" method, is presented as a formally defined domain independent method (Chapter 4) and its instantiation using our case study, automated container terminals (Chapter 5).

The "Multiple Worlds" design method allows for an iterative design process that is structured using a tree. By using several component libraries, various simulation models can be constructed to provide answers to questions relevant at each specific stage during the design process. The tree allows for branching to construct alternative designs: as each design can be presented as a simulation model, the results of the models can be used to compare different designs on key performance indicators. This allows for actors to assess the design decisions they consider prevalent during the process and analyze the results to choose among different alternatives. We instantiated this method to support the design process of automated container terminals. The design environment lets users input a design using pre-defined components that are translated to simulation models to output key performance indicators and animations that provide insight into the behavior of the system. The users can input their design either using a customized interface that requires design parameters (for designs specified at a high level of abstraction) or using AutoCAD (for designs specified at a low level of abstraction). These designs are translated using libraries of simulation components defined in DEVS using D-SOL, a Java based simulation framework. The simulation experiment outputs are animated in a 3D visualization and the statistical calculations are carried out to visualize the key performance indicators. We presented the design process that guides actors and lets them design alternative designs in different stages using the software environment.

To evaluate the design method, we organized a workshop wherein participants were required to design a fictive automated container terminal. After the workshop, we carried out a group interview to get an in-depth view on how the participants experienced the workshop. The interview focused on the use simulation models and the collaborative aspect of the design method. We presented the results of the workshop to an expert in container terminal design to analyze the process and the designs. The results from the interview with the participant and the interview with the expert gave us an understanding on the usability of the design method and the quality of the process and designs.

To close the rigor cycle, we reflected on our contribution by presenting the construct definitions, the model descriptions, and the methodical choices used in building the artifacts. To close the relevance cycle, we carried out multiple interviews with experts experienced in container terminal design and simulation models. We assessed the possible use of the method in business environments and identified the limitations it has for using it in business environments. Overall, the experts agreed on the potential of the method in support the use of simulation models throughout the design process while taking into account the interests of several actors involved in the process. Future work on the method includes assessing the method on other logistics systems, such as rail networks, and supporting multiple perspectives by having specific models for each perspective.

Samenvatting

Mulitple Worlds: een multi-actor simulatiegebaseerde ontwerpmethod voor logistieke systemen

Logistieke systemen zijn essentieel geworden voor onze huidige globale economie. Een substantieel deel van de totale vracht op wereldgebied wordt vervoerd via container terminals. Container terminals zijn de hoofdzakelijke hubs waarin containers worden overgebracht van en naar treinen, schepen en vrachtwagens. Technologische ontwikkelingen worden continue aangewend om de efficintie van container terminals te verbeteren. Zo kunnen we met de huidige vooruitgang delen van een container terminal automatiseren met computergestuurde apparatuur. Het ontwerpproces van dergelijke container terminals is complex vanuit een actor- en systeemperspectief. Container terminals zijn vaak het resultaat van projecten van meerdere miljoenen waarbij meerdere actoren betrokken zijn. Deze actoren kunnen zowel intern, als deel van een organisatie, als extern zijn zoals beleidsmakers, adviseurs en klanten. De interne actoren zoals deze die verantwoordelijk zijn voor business development en het operationeel ontwerp, zijn nauw betrokken bij het proces om een systeem te ontwerpen dat bestaat uit intergerelateerde subsystemen, zoals het opslaggedeelte van een terminal, de kade van de terminal en het terminalbesturingssysteem dat de geautomatiseerde voertuigen bestuurt.

Simulatiemodellen worden gebruikt om inzicht te verschaffen in het gedrag van het systeem. Simulatiestudies worden doorgaans uitgevoerd op het einde van een ontwerpproces zonder rekening te houden met de verschillende actoren die betrokken zijn bij het ontwerpproces. Hierdoor moet het gebruik van simulatiemodellen volgens de hard systems aanpak veranderen. Recente literatuur stelt voor om simulatiemodellen te gebruiken volgens de soft systems aanpak. Deze laatste presenteert een leerproces waarin iedere actor bewust is van de perspectieven van de overige actoren. De combinatie van het gebruik van simulatiemodellen en de soft systems aanpak resulteert in een verzameling kwesties die worden aangewend in onze methode: de Multiple Worlds ontwerpmethode.

We hebben het doel van ons onderzoek als volgt geformuleerd: "Een methode voorzien om multi-actor ontwerpprocessen voor logistieke systemen te ondersteunen gebruikmakend van simulatiemodellen". Om dit doel te bereiken, hebben we de volgende drie onderzoeksvragen gedentificeerd:

1. Wat zijn de huidige methodes om logistieke systemen te ontwerpen in de praktijk?

2. Wat zijn de problemen in huidige multi-actor ontwerpmethodes waarin gebruik wordt gemaakt van simulatiemodellen?

3. Hoe kan een multi-actor ontwerpmethode dat gebruik maakt van simulatiemodellen, de problemen aanpakken die resulteren uit de tweede onderzoeksvraag?

Met de eerste vraag, richten we ons op het identificeren van de huidige manier van werken in bedrijfsomgevingen waar simulatiemodellen worden gebruikt om logistieke systemen te ontwerpen. De tweede vraag helpt ons in het vinden van bestaande theorin die aangeven hoe simulatiemodellen kunnen worden gebruikt in multi-actor ontwerpprocessen. Uiteindelijk voegt de derde onderzoeksvraag de bevindingen van de eerste en tweede onderzoeksvragen samen om de "Multiple Worlds" methode te ontwerpen.

Om dit onderzoek uit te voeren, volgen we een kritisch-realistische ontologie en een interpretivistische epistemologie. De ontologie schrijft voor dat we onze kennis van de realiteit zien als het resultaat van sociale conditionering, wat betekent dat het niet begrepen kan worden onafhankelijk van de sociale actoren die betrokken zijn bij het kennisderivatieproces. De epistemologie geeft aan dat feiten en waarden met elkaar zijn verweven en dat ze dus beide deel zijn van onze wetenschappelijke kennis. Dit standpunt helpt ons in het creren van een rijk beeld dat het artefact en de actoren die benvloed worden door het artefact, bevat. Om dit onderzoek uit te voeren, volgen we de design science methodologie dat drie cycli presenteert die een balans verzorgen tussen relevantie en strengheid van het onderzoek dat als doel heeft het ontwerpen van een artefact. Onze onderzoeksvragen richten zich tot de drie afzonderlijke cycli: de eerste onderzoeksvraag wordt gebruikt voor de relevantie van het onderzoek, de tweede onderzoeksvraag leidt ons door de strengheidcyclus en de derde onderzoeksvraag helpt ons in het ontwerpen van onze methode.

We hebben een case study uitgevoerd bij APM Terminals als deel van de relevantiecyclus van ons onderzoek. In deze beschrijvende case study hebben we de ontwerpprocessen geanalyseerd die worden gebruikt bij APM Terminals, voornamelijk bij het ontwerpen van geautomatiseerde container terminals. Hiervoor hebben we onze aandacht besteed aan APMT's container terminals in Virginia (VS) en op de Tweede Maasvlakte in Rotterdam. In deze case study, hebben we ons geconcentreerd op het systeem (de container terminal als fysieke entiteit met apparatuur en controlesoftware) en op het actornetwerk (de actoren die betrokken zijn bij het ontwerpproces die moeten worden ondersteund). De case study toont de aanwezige complexiteit in het ontwerpproces van geautomatiseerde container terminals. De vele beslissingen die moeten worden genomen veroorzaken interacties in het technisch systeem en tussen de actoren betrokken bij het ontwerpproces. Simulatiemodellen kunnen het dynamisch karakter van technische systemen weergeven en hierdoor het besluitvormingsproces ondersteunen. We zijn tot de conclusie gekomen dat een herpositionering van het gebruiken van simulatiemodellen ervoor zou kunnen zorgen dat actoren vanaf het begin van het ontwerpproces ondersteund kunnen worden in het evalueren van alternatieve ontwerpbeslissingen.

In de strengheidcyclus, hebben we nader gekeken naar systeemkunde zoals deze bedoeld werd vanaf de conceptie en hoe het werd opgesplitst in hard systems en soft systems aanpakken. We hebben bestudeerd hoe deze twee aanpakken kunnen worden aangewend om systemen te ontwikkelen en wat de problemen zijn in het combineren van deze aanpakken in een multimethodologische aanpak. In ons onderzoek, stellen we een nieuwe methode voor waarin deze problemen worden aangepakt, waar dus simulatiemodellen worden gebruikt in een multi-actor omgeving. We zijn op zoek gegaan naar modellen en methodes waarmee we ieder probleem afzonderlijk kunnen aanpakken. Startende van deze modellen en methodes, hebben we constructies gedentificeerd die kunnen worden gebruikt in onze methode: componentgebaseerde ontwikkeling, verschillende niveaus van specificatie, het structureren van alternatieven, en participatief ontwerp. De resulterende methode is gepresenteerd als een formele domeinonafhankelijke methode die we hebben genstantieerd voor het ontwerpen van container terminals.

De "Multiple Worlds" methode stelt een iteratief ontwerpproces voor dat een boomstructuur volgt. Door verschillende componentbibliotheken te gebruiken, kunnen verscheidene simulatiemodellen worden ontwikkeld die vragen beantwoorden specifiek voor iedere fase van het ontwerpproces. Tijdens het proces kan een opsplitsing in de boomstructuur leiden tot alternatieve ontwerpen: ieder ontwerp is gekoppeld aan een simulatiemodel waardoor de resultaten kunnen worden gebruikt om alternatieven te vergelijken. Dit stelt actoren in staat om ontwerpbeslissingen te evalueren die ze belangrijk vinden en de resultaten te analyseren van alternatieve ontwerpen. We hebben de methode genstantieerd om het ontwerpproces te ondersteunen van geautomatiseerde container terminals. De ontwerpomgeving laat gebruikers toe om ontwerpen in te geven gebruikmakend van voorgedefinieerde componenten die worden vertaald naar simulatiemodellen om key performance indicators en visualizaties weer te geven die inzicht verschaffen in het gedrag van het systeem. De gebruikers kunnen hun ontwerpen ingeven met behulp van hoog-niveau parameters of met behulp van AutoCAD tekeningen om meer gedetailleerde ontwerpen te bepalen. Deze ontwerpen worden vertaald met behulp van DEVS simulatiecomponenten ontwikkeld met behulp van DSOL, een Java-gebaseerde simulatieomgeving. De resultaten van de simulatie-experimenten worden weergegeven in een 3D visualisatie en statistische berekeningen worden uitgevoerd om key performance indicators te bepalen. We hebben tevens het ontwerpproces gepresenteerd dat actoren begeleidt om alternatieve ontwerpen te ontwikkelen gebruikmakend van deze softwareomgeving.

Om de ontwerpmethode te evalueren, hebben we een workshop georganiseerd waarin deelnemers werd gevraagd om een fictief geautomatiseerde container terminal te ontwerpen. Na de workshop, hebben we een groepsinterview uitgevoerd om te begrijpen hoe de deelnemers het gebruik van de methode tijdens de workshop hebben ervaren. De focus van het interview zat op het gebruik van simulatiemodellen en op de samenwerking tussen verschillende actoren. We hebben de resultaten van de workshop voorgelegd aan een expert om het proces en de ontwerpen te analyseren. De resultaten van het interview met de deelnemers en de interview met de expert hebben ons een inzicht gegeven in de bruikbaarheid van de ontwerpmethode en de kwaliteit van het proces en de ontwerpen.

Om de strengheidcyclus te sluiten, hebben we gereflecteerd op onze contributie door het presenteren van constructiedefinities, de modelbeschrijvingen en de methodologische keuzes die we hebben gebruikt in het bouwen van het artefact. Om de relevantiecyclus te sluiten, hebben we meerdere interviews uitgevoerd met experts in het gebruik van simulatiemodellen en het ontwerpen van logistieke systemen. Ze hebben geoordeeld over het mogelijke gebruik van de methode in bestaande bedrijfsomgevingen en hebben de beperkingen gedentificeerd van de methode. De experts concludeerden over het algemeen dat de methode potentieel gebruikt kan worden om meerdere actoren te ondersteunen in ontwerpprocessen waarin simulatiemodellen worden gebruikt. Toekomstig werk hierop kan zich focussen op het toepassen van de methode in andere logistieke systemen dan container terminals en het ondersteunen van verschillende perspectieven door het aanbieden van perspectiefgebonden modellen.

Curriculum Vitae

Michele Fumarola was born in Maasmechelen (Belgium) on March 11, 1983. In 2001, he graduated from the Sint-Pauluscollege in Houthalen (Belgium). He obtained his B.Sc. (2005) and M.Sc. (2006) in computer science at Hasselt University in Belgium. After his graduation, he worked as a researcher at Expertise Centre for Digital Media, a research institute of Hasselt University specialized in computer graphics, human-computer interaction and networked virtual environments. As a member of the Human-Computer Interaction group, he was first involved in projects regarding pervasive computing and later regarding user-centered design. The latter aimed at applying user-centered design principles in a number of industrial projects. In June 2007, he started his Ph.D. trajectory under the supervision of prof. dr. ir. Alexander Verbraeck in the Systems Engineering Section of the Technology, Policy and Management Faculty of Delft University of Technology.

Michele presented his research at several international conferences in Australia, Hong Kong, Singapore, the Netherlands, Belgium, Spain, and the USA. He also published his work in several academic journals and books.