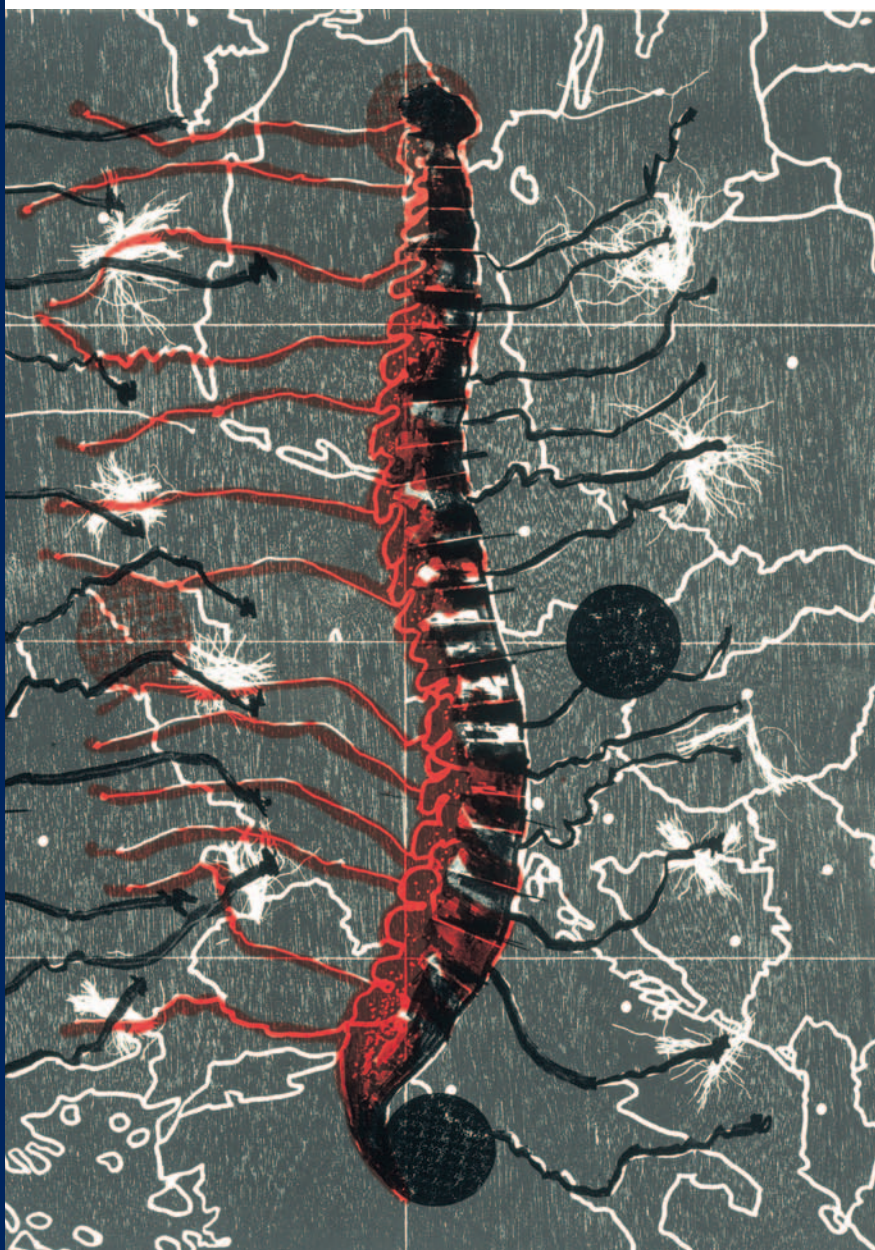
CSABA ATTILA BOER

Distributed Simulation in Industry



Distributed Simulation in Industry

Gedistribueerde simulatie in de industrie

Thesis

to obtain the degree of Doctor from the
Erasmus University Rotterdam
by command of the
rector magnificus

Prof.dr. S.W.J. Lamberts

and according to the decision of the Doctorate Board.

The public defence shall be held on
Friday 21 October 2005 at 13:30 hrs.

by
Csaba Attila Boer
born at Satu Mare, Romania

Erasmus Research Institute of Management (ERIM)
RSM Erasmus University / Erasmus School of Economics
Erasmus University Rotterdam
Internet: http://www.erim.eur.nl
ERIM Electronic Series Portal: http://hdl.handle.net/1765/1
ERIM Ph.D. Series Research in Management 65

The research reported in this thesis has been carried out in cooperation with SIKS (The Dutch Research School for Information and Knowledge Systems) and TRAIL (The Netherlands Research School for TRAnsport, Infrastructure and Logistics).

SIKS Dissertation Series No. 2005-12
TRAIL Dissertation Series No: T2005/12

ISBN 90–5892–093–3

Note on cover illustration

The backbone can be considered as a unifying element of a “body”. By unification, distributed elements are able to collaborate with each other on issues which otherwise could not be addressed by autonomous elements. The central question of my research is to create a backbone that supports the coupling of distributed simulation models. I found a similarity in Evelyn´s artwork in which he reflects the ongoing process of unification of the countries in an expanding European community by using a backbone structure as a coordinator.

Doctoral Committee:

Promotors:	Prof.dr. A. de Bruin
	Prof.dr.ir. A. Verbraeck
Other members:	Prof.dr.ir. R. Dekker
	Prof.dr.ir. R. Fujimoto
	Dr.ir. H. de Swaan Arons
	Dr. S.J.E. Taylor
	Prof.dr. A.P.M. Wagelmans

To
my wife Katica and my son Danika

ACKNOWLEDGEMENTS

This book is the outcome of a long journey that started more than 1800 km away from Rotterdam. Several people joined me on this journey and influenced the content of this book by drawing my attention to various interesting aspects of it.

The ones who actively influenced the content of this book are my promotors. I would therefore like to express my gratitude to Arie de Bruin and Alexander Verbraeck for their support. The brainstorming sessions the three of us had together improved the quality of this book immensely. This journey would probably have lasted much longer if I had not had the opportunity to work with Alexander, who oriented me in the direction of this specific topic and involved me in relevant projects. I am greatly indebted to Alexander for teaching me to think not only as a system engineer but also as a scientist. I am grateful to Arie for reading and commenting on my thesis endlessly. I thank him for visiting me several times to discuss the content of the thesis after I had started my new job. Without his intensive support I would not have been able to finish the thesis in time. Furthermore, I would like to thank Henk de Swaan Arons for supporting me at the beginning of my research and for directing me to the field of simulation.

Rommert Dekker is the person who initiated my journey, as he was the one who invited and accepted me for a position as a researcher. Although our ways parted in the first months as I was affiliated with the computer science department, I would like to thank him for giving me the chance to start my research activities at Erasmus University.

During my Ph.D. research I was affiliated with several research schools and institutes, such as ERIM, TRAIL, SIKS and BETADE. I am grateful to all of them for their scientific and managerial support. My special thanks go to the members of the BETADE group for accepting me in their team, and involving me in several interesting and useful meetings and workshops.

An important part of my research was the data collection using questionnaire and interview surveys. I would like to thank the Commercial-Off-The-Shelf simulation package vendors for participating in the questionnaire survey and the simulation experts for the useful discussions during the interview survey. Special thanks go to Richard Fujimoto with whom I had an important face-to-face conversation concerning the final evaluation of the surveys. In addition to the surveys, I had the pleasure to be involved in two FAMAS projects, which are used for my case studies. I would therefore like to thank all the participants and sponsors of these projects for their collaboration.

Research activity is not limited by any borders. Connecting people and ideas from various countries is also illustrated by the cover of this book. I am grateful to Angelo Evelyn for providing me with the opportunity to use his artwork for this reason. In travelling to international conferences and being a member of international research groups, I had the chance to collaborate with several researchers from other countries. I would like to thank Simon Taylor for his interest in my work and for inviting me to Brunel University, where I had the opportunity to meet other researchers from distributed simulation area. I especially found fruitful the discussions with Mike Ryde. During one of my stays in London I had the pleasure of being Mike's guest. I would like to thank Mike and his family for their

hospitality and for the conversations about simulation, education, ponies and many other topics.

The friendly atmosphere within the computer science department made my work even more enjoyable. I would like to thank my colleagues, in particular my room mates Gerard and Vladimir for the interesting conversations. My special thanks go to Cia for her advice and for helping me to find my way in the Netherlands.

During a long journey it is important to have a rest from time to time in order to "recharge", in order to be able to continue on our way. Thanks to my best friends, with whom I spent a great time in the Netherlands. Thank you Lehel, Emőke, Bart, Gabriella, Bert, Ildikó, Marc, Szabi, Robi and Tímea.

I would not have been able to start and complete this journey without the support and love of my family. I would like to thank my parents-in-law for their visits and for helping us with the housework during the time I was writing the thesis. I am grateful to Manyó for the advice she has given me throughout my life, for always being open to discussions about any topic and especially for being such an amazing sister. Special thanks go to my dear parents for their love and for their everlasting support. *Drága édesanyám és édesapám szívből köszönöm nektek mindazt, amit értem tettetek.*

Finally, the person to whom I am the most grateful is my wife Katica, who, next to being an excellent wife, is my best friend and a helpful colleague. I thank you very much for your love, patience and the support that you gave me from the beginning to the end of this journey. Furthermore, I thank you very much for giving birth to the most wonderful present I could ever have: our son, Danika.

Csaba Attila Boer

Capelle aan den IJssel, July 2005

Table of Contents

List of Figures

List of Tables

1 INTRODUCTION

Modelling and computer simulation are accepted problem solving methodologies for the solution of many real-world problems. Computer simulation refers to "methods for studying a wide variety of models of real-world systems by numerical evaluation using software designed to imitate the system's operations or characteristics, often over time" (Kelton*, et al.* 2003, pg. 7). During the last decades several simulation modelling styles have appeared, such as discrete and continuous simulation, that modellers apply to build computer simulation models (Zeigler*, et al.* 2000), (Vangheluwe and De Lara 2002). The simulation modelling style we consider in this thesis is discrete simulation and, henceforth, when we refer to computer simulation, or simply simulation we mean discrete-event computer simulation. Computer simulation is becoming increasingly more popular and powerful, due to the continuously advancing and increasing popularity of computer and software technologies.

In the late 1950s and 1960s computer simulation was a very expensive and specialized tool which was generally used only by large corporations (e.g., steel and aerospace organisations) that required large capital investment. In 1970s and early 1980s computers were becoming faster and cheaper and the value of simulation was being discovered by other industries as well. Simulation really began to mature during the early 1990s when many smaller firms started to use commercially available simulation tools (Kelton*, et al.* 2003), (Nance 1993). These tools provide a natural framework for simulation modelling, including extra features that are needed to build a simulation model (e.g., predefined simulation libraries) and high-quality animation (Law and Kelton 2000). They helped the simulation to establish its roots in the business and more and more organisations started to use them during the early design phase of their systems, before any production. As Kelton argues "the rate of change in simulation has accelerated in recent years, and there is every reason to believe that it will continue its rapid growth" (Kelton*, et al.* 2003, pg. 15).

With the rapid advances being made in computer and software several new branches appeared in the computer simulation domain. One of these branches is *distributed simulation.* Distributed simulation refers to technologies "that enable a simulation program to execute on a computing system containing multiple processors, such as personal computers, interconnected by a communication network" (Fujimoto 2000, pg. 4). The appearance of distributed simulation mainly relates to military domain (Singhal and Zyda 1999). The needs of the military establishment to have more effective and economical means to train personnel has driven a large body of work in developing virtual environments where distributed participants can interact with each other as if they were in actual combat situation (Fujimoto 1998a). Although the military establishment has had a major role in developing distributed simulation technology for virtual environments, recently there is a growing interest to apply distributed simulation in industry as well (Taylor*, et al.* 2002), (Straßburger*, et al.* 2003).

At present, however, the application of distributed simulation in industry is still in its infancy. The simulation community in industry is still looking for an acceptable solution to couple distributed simulation models. We illustrate the need for coupling through three typical examples from different industrial domains: manufacturing, supply chain management and container logistics. These examples serve as a primary motivation for our

research that is aimed to introduce an appropriate approach for coupling distributed simulation models.

1.1 Distributed Simulation for Car-Building Industry

The vision of DaimlerChrysler and other major auto makers is that "by 2005, every production factory will be planned, built, launched and operated first using "full simulation", before going to bricks and mortar" (DaimlerChrysler 2003). These car makers expect that in the future, the entire factory process will be digitally represented and simulated before going to brick and mortar by means of consistent data management, simultaneous development of product and production, and early consideration of production requirements in development. The problem is, however, that simulations serving as digital representations, like "most common applications, are narrowly focused on isolated questions and do not provide a complete and universal reflection of the complex process chain" (Straßburger*, et al.* 2003, pg. 112). Some processes within a real factory might have been modelled independently and analyzed separately for local purposes (e.g., the paint shop model, or assembly model). When pursuing more global purposes, we might find out however that the isolated processes are in causal relationship with other processes (e.g., after the assembly process the car needs to be painted). The objective of such a digitally represented and simulated factory, or Digital Factory as DaimlerChrysler calls it, is to have a detailed digital representation of the whole real factory, which covers all relevant causal relationships. A natural way to achieve this objective is to couple several independently designed and developed simulation models each representing a part of the factory (e.g., a chain in the production process) and to provide an IT framework that allows for the *coupling* of simulation models (Waller and Ladbrook 2002), (Straßburger*, et al.* 2003).

1.2 Distributed Simulation for Supply Chain Logistics

A supply chain refers to the flow of materials, information, and services, from the raw material suppliers through factories and warehouses to the end customers. A supply chain includes the organizations and processes that create and deliver these products, information and services to the end customer (Hugos 2003). The most important reasons for the increasing attention and practice of supply chain management are the possibilities that technology enhancement have generated (Boyson*, et al.* 2003). In (Boyson*, et al.* 2003) a supply chain project is discussed, the aim of which is to illustrate the future to supply chain planners across Department of Defense (DoD) and to create a prototype of an infrastructure for real-time supply chains. In this project, simulation is applied, again, as a powerful tool for solving supply chain logistics problems. However, in contrast to modelling processes related to only one organization, supply chain modelling entails additional difficulties since it involves different organizations. A key barrier to full supply chain management has been the cost of communication with, and coordination among, the many independent suppliers in each supply chain (Fredenhall and Hill 2001). According to Gan, "building a detailed model of the supply chain does not pose a problem when the chain involves only a single organization. In contrast, not many participating organizations are willing to share detailed model information when the chain crosses the organization

boundaries" (Gan*, et al.* 2000, pg. 1245). In order to cater for this, each organisation can design and develop its own simulation model, thereby encapsulating the internal information. The task is then to couple the independently designed and developed simulation models in order to analyze the simulated supply chain as a whole. Therefore simulation *coupling* plays an important role in supply chain projects involving different organizations and this leads to the requirement for a well designed standard IT framework for coupling the constituent simulation models (Banks*, et al.* 2003).

1.3 Distributed Simulation for Container Logistics

The port of Rotterdam, one of the major ports in Europe, is confronted with continuously increasing container traffic. Therefore a long term project was initiated in order to design new container terminals for a new Port of Rotterdam. The new container terminals, compared with the existing ones, will provide additional facilities, such as handling Jumbo Container Vessels, that are ocean giant vessels with a capacity of twelve thousand TEU (Twenty Feet Equivalent), providing AGVs (Automated Guided Vehicles) and supporting optimal connections to all forms of onward transportation (truck, train, barge, vessels, etc.) (Connekt 2001). In container logistics, as with the previous examples, simulation is a powerful tool that plays an important role in the planning and designing process (Bluemel and Novitsky 2000). Several organisations are involved in designing the new container terminals and they use simulation for performing their assignment. Although these different organisations concentrate on designing simulation models for different parts of the terminal (e.g., the AGVs, truck handling, barge handling, etc.), they have a common goal, namely the accurate functioning of the container terminal as a whole. In order to investigate the global behaviour of the simulated container terminal, *coupling* of these models is needed. Coupling helps to analyze interactions and side effects, and can give feedback regarding the performance and achievements of the integrated system. In order to provide an effective solution for integration there is a need for an IT framework that supports the coupling in a uniform way (Veeke*, et al.* 2002), (Boer*, et al.* 2002b).

1.4 Research Objective

The organisations and interorganisations in the three cases illustrated above all intend to understand and improve the behaviour of their systems by applying simulation. We would like to emphasize three characteristics of the projects presented:

Complexity. The systems that need to be represented in the three cases are all complex and consist of various independent subsystems. The software engineering community recognized long time ago the "divide and conquer" approach for solving complex problems. The idea is to break down the complex problem into several subproblems, then to solve the subproblems separately and finally to combine the solutions of the subproblems into a solution for the original complex problem (Cormen*, et al.* 2001). Although in the above examples there are interrelationships among the subsystems, some of them, such as the assembly process in the digital factory, or the AGVs that are applied in the container terminals, can be analyzed individually. In order to analyze the global behaviour of the whole system, however, besides separately studying all concerned subsystems, the interrelationships between them must be taken into account as well.

Collaboration. Since simulation models are designed and developed by multiple teams, collaboration plays an important role in the above projects. Multiple teams can be formed within the same single organisation (e.g., in the case of the digital factory), or every team can be the representative of distinct organisations that collaborate (e.g., in the case of a supply chain or the container terminal). Given the fact that multiple teams are involved it is natural to divide the models in submodels and let the teams work on submodels in parallel. Collaboration in a case like this is needed, for example, to combine the submodels. Collaboration might be made easier by using an appropriate standardized framework for coupling distributed simulation models.

Information hiding. Especially in the second example, the supply chain logistics, information hiding was crucial, as different collaborating organisations wanted to hide confidential information concerning the business process of their own subsystem.

The three large scale organisational and interorganisational simulation applications from different industrial areas described above, point out the need for a framework that supports the *coupling of simulation models.* The separated, that is distributed design of several simulation submodels followed by their integration would help to easier deal with the complexity of the system, would encourage collaborative design and development and would support information hiding.

While the software engineering community already recognized the potentials of coupling distributed software components (Szyperski 1998), (Heineman and Councill 2001), in the simulation domain we rarely observe this phenomenon (Taylor*, et al.* 2002), (Taylor*, et al.* 2003a), (Straßburger 2001). Although the military domain has done some work in this direction by interfacing distributed training simulation models using distributed simulation architectures (Weatherly*, et al.* 1991), (Kuhl*, et al.* 1999), (DMSO 1998a), (DMSO 1998b), (DMSO 1998c), (Singhal and Zyda 1999); as we will explain in Chapter 3 and 4, in the industrial domain an effective distributed simulation architecture for coupling simulation models is still lacking (Taylor*, et al.* 2002), (Taylor*, et al.* 2003a). We expect that, by providing such an architecture, large scale simulation model-based problem solving activities can be improved. In this research we pursue the following research objective:

Research Objective

Provide an architecture for coupling simulation models and test its appropriateness in industry.

In order to accomplish the research objective, the following three research questions should be answered.

Research Questions

RQ 1. *What are the requirements for an appropriate architecture for coupling simulation models in industry?*

RQ 2. *Can an architecture be found, adapted or designed that satisfies these requirements?*

RQ 3. *Is it possible to successfully apply this architecture in practice?*

In order to achieve the research objective and to answer the research questions we will follow a specific research strategy, which is presented in the next subsection.

1.5 Research Strategy

Researchers approach building and testing of theories from two directions (Creswell 2003). The *deductive approach* begins with abstract, logical relationship among concepts, and moves towards concrete empirical evidence. The *inductive approach* begins with detailed observations of the world and moves towards more abstract generalisations and ideas. Taking into account these two basic research directions, Churchmann defines five types of research strategies, or as he calls them *inquiring systems*, which can be distinguished based on the way the scientific knowledge is gathered (Churchmann 1971), (De Vreede 1995):

- Leibnitzean - expanding scientific knowledge by formal deduction from elementary forms of knowledge
- Lockean - expanding scientific knowledge by induction from sensing experiences, endowing them with properties, and combining them with previous experiences
- Kantian - expanding scientific knowledge by formal deduction as well as by informal experiencing through a set of priori sciences; blend of Leibnitzian and Lockean
- Hegelian - expanding scientific knowledge objectively by identifying conflicting interpretations of observations, and going beyond this conflict through synthesis
- Singerian - expanding scientific knowledge by adapting it endlessly, inductively, and multidisciplinary based on new observations

In order to achieve our research objective, namely, *provide an architecture for coupling simulation models and test its appropriateness in industry,* first of all we need to gather some primary scientific knowledge regarding simulation, modelling, integration or coupling, component based development, distributed systems, etc. This knowledge can be obtained from different existing sources, such as simulation theory (Zeigler*, et al.* 2000), distributed system theory (Tanenbaum and Van Steen 2002), component-based development theory (Szyperski 1998), (Heineman and Councill 2001), distributed simulation theory (Fujimoto 2000) and theories on simulation model integration in the military domain (Singhal and Zyda 1999). Although these theories are useful for our purposes, we need to expand the primary knowledge gathered from these theories in order to design an approach for appropriately coupling simulation models. This will be carried out in an inductive way. Through observations we try to identify all factors that are essential in order to appropriately couple simulation models and we adopt these factors within our approach. Taking into account the above considerations, apparently we have found the Singerian research strategy the most suitable approach.

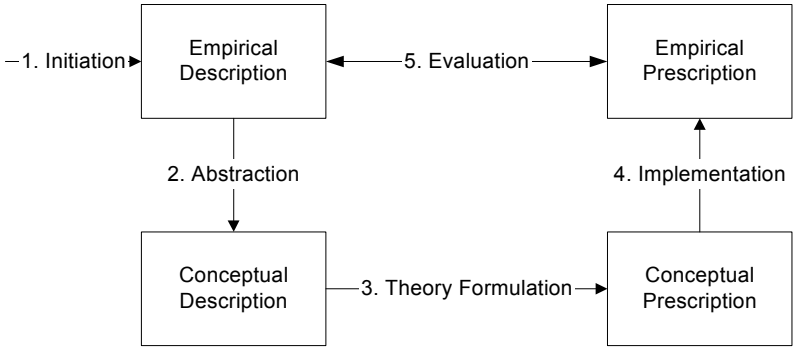


Figure 1.1 Inductive Hypothetical Research Cycle

Based on Churchmann's inquiring system an inductive-hypothetical research cycle strategy has been proposed in (Sol 1982). Our research follows this strategy (Figure 1.1).

Since in the military domain effort has been put into coupling simulation models, during the *initiation* step we aim to describe solutions from this domain. In this phase especially literature review and personal observations play an important role, leading to an initial, empirical description of the current state of affairs in distributed simulation. Further, in order to validate our empirical description and to obtain an understanding of the current situation we conduct a questionnaire and interview survey. Then, we analyze the data collected and we identify concepts and causal relations between these concepts through an *abstraction*, which results in a conceptual description. This conceptual description provides the basis for the required features for an appropriate distributed simulation architecture for industry. In a sense this can be called *theory formulation*. The concept of *theory* in the context of the inductive hypothetical research strategy can be applied in a broad sense, for example to provide modelling support (Verbraeck 1991), a set of problem handling guidelines and modelling concepts (Wierda 1991) or an inquiring system (Sol 1982, De Vreede 1995). *In this research theory refers to an appropriate design approach for a distributed simulation architecture for industry expressed in form of requirements.*

The requirements form in fact the conceptual prescription of an architecture that should be designed and developed in order to achieve the research objective. During the *implementation* phase we intend to build and present such an architecture. The implemented distributed simulation architecture, that is the empirical prescription, has to be finally *evaluated* in order to test to what degree it accomplishes the research objective. In this phase case studies play an important role (Yin 1994).

1.6 Outline of the Thesis

The outline of this thesis is depicted in Figure 1.2. The thesis is structured in seven chapters, the first chapter being this introduction part. In Chapter 2 we present the state of the art in simulation model integration. In the first half of this chapter we elaborate on the role of simulation model integration as part of, what we call, simulation model-based problem solving activities. After that we discuss distributed simulation which is an approach that provides theory for simulation model integration. First, the theory behind distributed simulation is presented followed by a discussion on the most advanced available tools and standards that provide support for distributed simulation. Finally, the High Level Architecture (HLA), being the most advanced tool and standard for distributed simulation, is discussed in more detail.

The fact that HLA is the most advanced tool for distributed simulation suggests us to select this solution, instead of designing and developing another one, in order to achieve our research objective. However, since we perceive that HLA standard is hardly applied in industry, we are not completely confident with this choice. Therefore, in Chapter 3 we investigate in more detail whether and to what degree is our observation accurate. For that reason, we conduct a survey confronting a group of experts with this observation, stated as a hypothesis. We request the experts to motivate their answers and indicate alternative solutions to achieve our research objective. In Chapter 3 we present how we selected the experts, we discuss the methodology that we applied for data collection and analysis, and finally we present the collected data in a structured form.

We analyze the collected data in Chapter 4. The analysis firstly aims to identify the advantages and drawbacks of applying distributed simulation in industry, secondly, to discuss the appropriateness of existing approaches for industry, and finally to propose a design approach for distributed simulation architecture for industry. The proposed approach is expressed in form of a list of requirements that needs to be satisfied when one intends to design and develop distributed simulation for the industrial domain. The requirements are validated by distributed simulation experts. The resulting set of requirements provides the answer to our first research question presented in Section 1.4.

In Chapter 5 we present a tool for coupling simulation models, which we have designed and developed for the industrial domain. The tool, called the FAMAS Simulation Backbone Architecture, was specifically built to couple port related simulation models. In this chapter we aim to provide the answer to the second research question presented in Section 1.4.

In Chapter 6 we evaluate the FAMAS Simulation Backbone Architecture through two case studies. These case studies aim to show the industrial applicability of the backbone, through which we try to answer our third research question. Finally, in Chapter 7 we evaluate the FAMAS Simulation Backbone Architecture in view of the requirements proposed in Chapter 4. In second part of Chapter 7 we reflect on our research by discussing our research findings and by discussing opportunities for further research.

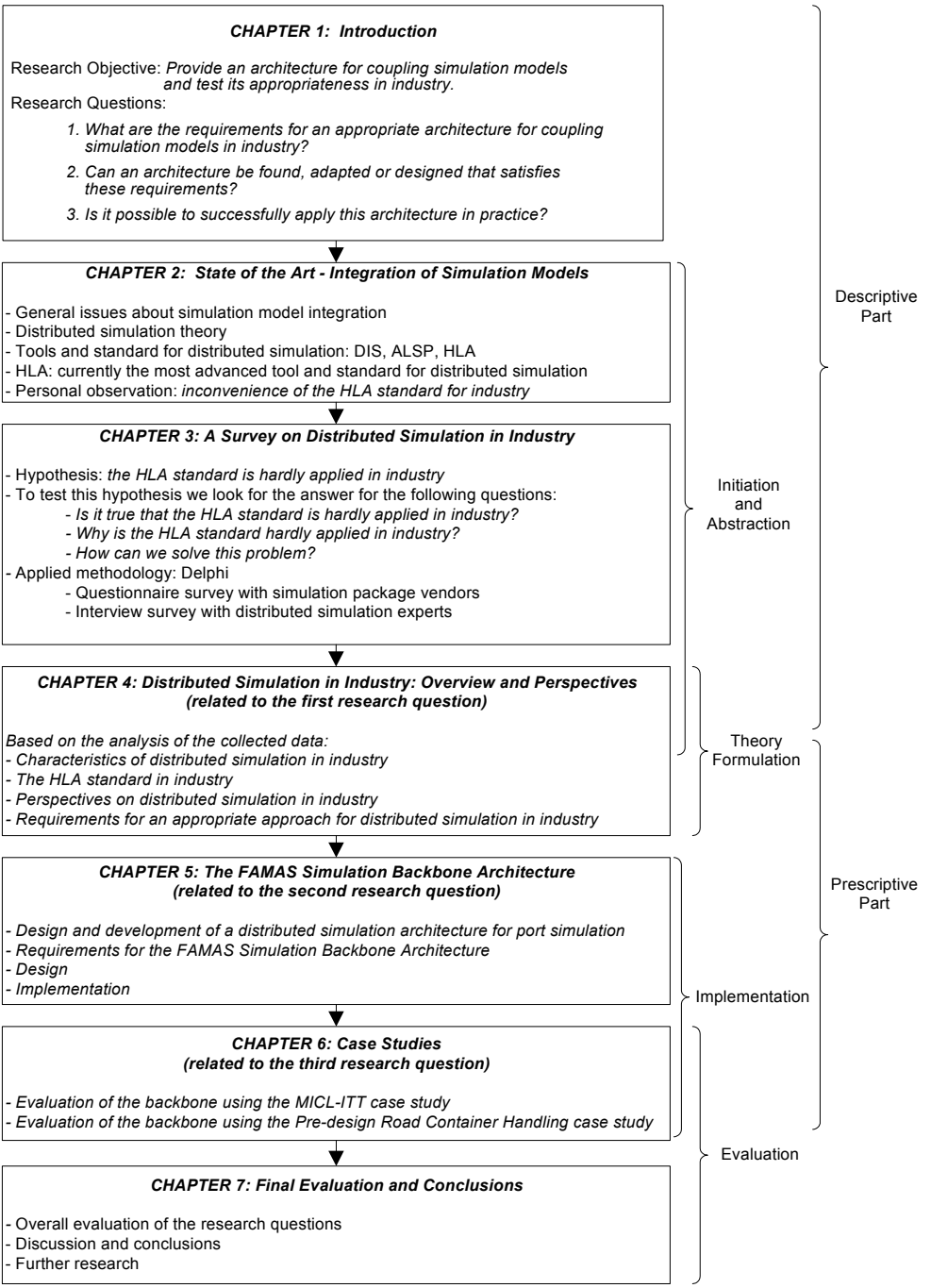


Figure 1.2 Research Outline

2 STATE OF THE ART – INTEGRATION OF SIMULATION MODELS

2.1 Introduction

In this chapter we aim to present the state of the art in distributed simulation. In Chapter 1 we gave some examples of distributed simulation and we illustrated the need to integrate simulation models by sketching three typical projects from the industrial domain. In this chapter we focus first, in Section 2.2, on modelling approaches from a general point of view and present a general model based problem solving strategy. Then, in Section 2.3, we zoom in on distributed modelling and model integration, thus introducing the notion distributed model based problem solving strategy. After that, in Section 2.4, we focus on the role of simulation within this distributed model based problem solving strategy. Finally, in Section 2.5 we elaborate on the distributed simulation approach and simulation model integration, and present the tools and standards for distributed simulation that are currently available.

2.2 Model Based Problem Solving Strategy

We can interpret reality in order to understand and control it as consisting of various systems (Bunge 1979). A system is defined as "a set of two or more interrelated elements of any kind" (Ackoff 1974, pg. 13). A *model* is a representation of such a system. A model is an abstraction of a system in the sense that it is less complicated than reality and hence it can be easier used for manipulation. Several definitions of the notion model exist. Ackoff defines a model as "a representation of reality" (Ackoff and Sasieni 1968, pg. 9). In this thesis we use the notion of model as refined by Pidd. According to him a model is "an external and explicit representation of part of reality as seen by the people who wish to use that model to understand, to change, to manage and to control that part of reality" (Pidd 2003, pg.12).

We discuss problem solving as conducted by organizations. An *organization* can be defined as "a purposeful system" (Ackoff 1974, pg. 18). It is considered a system because it consists of a set of interrelated elements that perform certain organizational functions. Furthermore it is purposeful because an organization functions with the aim to fulfil the demands of its environment (Ackoff and Sasieni 1968), (Simon 1969).

If an organization tries to achieve goals, *decisions* must be made taking into account various courses of actions. Managing organizations involves decision making and decision making entails problem solving. In this view managers are *problem owners*, and they tend to solve these problems through a decision making process.

In order to solve a problem a *problem solving strategy* (Ackoff 1974) needs to be worked out. In (Simon 1965), (Mitroff 1974), (Ackoff 1974), and (Sol 1982) a problem solving strategy is described that explicitly uses models. We will base our research on this general model based problem solving strategy regarding organizational and interorganisational problems (see Figure 2.1).

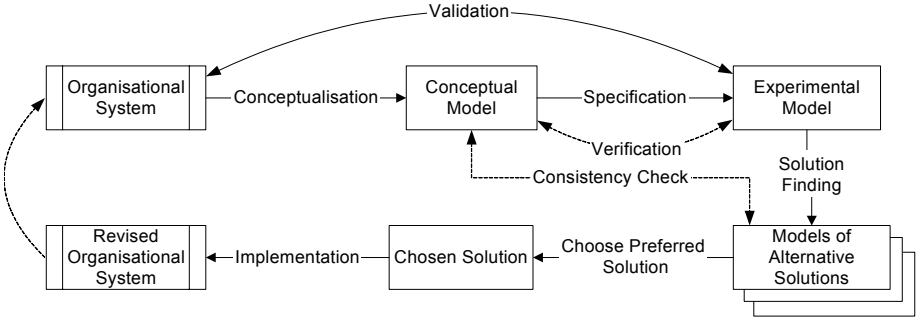


Figure 2.1 Model Based Problem Solving Strategy (derived from (De Vreede 1995))

The problem solving strategy depicted in Figure 2.1 involves five main steps, conducted in the following order:

1. Conceptualization
2. Specification
3. Solution Finding
4. Choosing the Preferred Solution
5. Implementation

Before the conceptualization process is started it is important to set objectives and to establish an overall project plan. Objectives indicate the questions that need to be answered by the problem solving process, while the overall project plan should include the various scenarios that will be investigated. During the *conceptualization* process the real problematic system is abstracted into a conceptual model. A *conceptual model* is "a model formulated in the mind of the modeller and specified in a variety of communicative forms intended for different users such as managers, analysts and developers" (Balci 2003, pg. 152). Communicative forms can include text, animation, audio, chart, drawings, equations, graphs, images, video, etc. Although these conceptual models are highly abstract, creating a conceptual model helps in structuring perception, representation and reasoning on the problem at hand (Sol 1982), and further, it conveys the decision maker's perception of the system to other people.

In the next step in the problem solving strategy, the conceptual model has to be translated into an *experimental model*. This translation is carried out through a so called *specification* process. An experimental model is more specific than a conceptual model in the sense that it allows the users to carry out experiments in order to analyze the problem. Different types of experimental models exist. Management science, while focusing on model building distinguishes three types: optimization, simulation and heuristic models (Keen and Scott-Morton 1978). In order to analyze the accuracy of transforming a conceptual model into a specific experimental model a *verification* process is required. Model verification analyzes whether the model is transformed from one form into another with sufficient accuracy. Next to verification a *validation* process of the experimental model against the existing system is needed in order to substantiate whether the experimental model behaves with satisfactory accuracy and is consistent with the stated objectives. "While model

verification deals with building the model *right*, model validation deals with building the *right* model (Balci 1998, pg. 336).

The third step of the problem solving strategy, the Solution Finding process, also referred to as the experimentation process, leads to a set of alternative solutions for the problem. The alternative solutions generated might go beyond the scope of the original problem; therefore a *consistency check* is required with the conceptual model. If a solution is not expressible in terms of the conceptual model, then probably the boundaries of the problem have been overstepped. In this case reconceptualization and respecification is required.

After evaluating the alternative solutions, the most appropriate solution is chosen in the next step, *Choose Preferred Solution*. The problem owner or the modeller compares the different alternatives and makes a decision choosing one solution. If the decision is made by the modellers, the chosen solution is documented and presented to the problem owners. The problem owners decide whether to implement the chosen solution resulted by the experiments.

The last activity that needs to be carried out is the *implementation* of the preferred solution in the organization in order to actually solve the problem. The problem owner should keep in mind the fact that the implementation of the solution in the organization may bring up new problems. In such a case we can start the whole process from the beginning by formulating another problem, which illustrates its cyclical nature.

2.3 Problem Solving Strategy Based On Distributed Models

Next, we would like to elaborate on the second step of the problem solving strategy, more precisely on the way experimental models can be designed and developed. Experimental models aim to provide alternative solutions for the problem stated through experimentations. Modellers can choose to design one single, compact model referred to as a *monolithic model* that incorporates all the functionality needed for the experimentation. However, they could also decide to design and develop a collection of submodels that focus on subproblems that are interrelated and by integrating them they form the whole experimental model which is intended to solve the main problem. We refer to a collection of interrelated submodels as a *modular or distributed model*.

In this research we focus on the second case where the experimental model is a distributed one. For this reason we extend the general strategy discussed above, introducing a *problem solving strategy based on distributed models*. Not only the experimentation model can be distributed, but the conceptual model as well, by constructing it as a set of several conceptual submodels. However, in this thesis we focus only on the experimental models. Now that several submodels are designed, in order to get the final experimental model, the submodels have to be integrated. Therefore, this strategy additionally comprises an *integration* phase which, as Figure 2.2 illustrates, extends the specification step of the problem solving strategy.

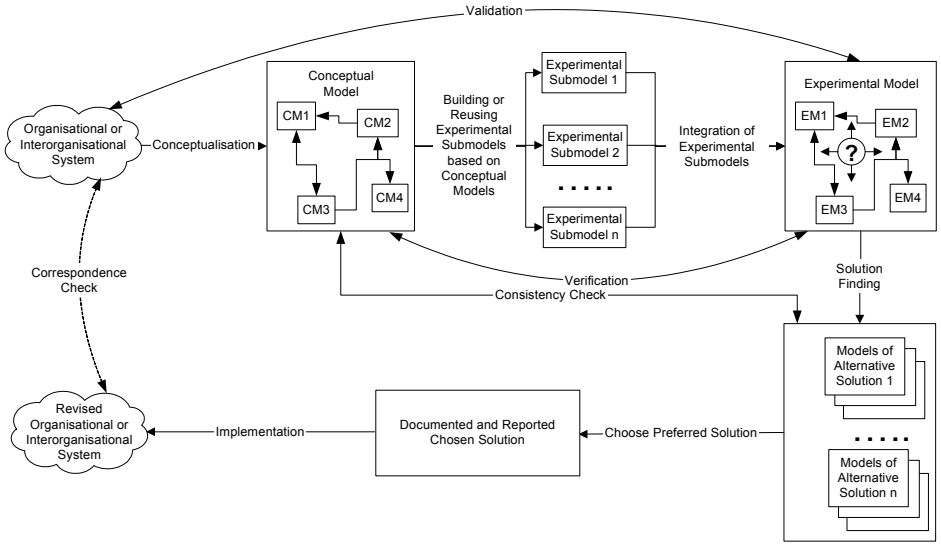


Figure 2.2 Problem Solving Strategy Based on Distributed Models

Integration of individually developed models is not just a *plug and play* issue, although that would be desirable. Children, who are building a castle or a car using the pieces of a Lego game, are in a favourable position because the Lego pieces are tailored as to support an easy assembling process. Lego blocks are built according to certain criteria that enable one to combine the different pieces. The collection of these criteria is specified as a *standard*. Accordingly, in order to be able to integrate distributed models as easy as it is possible with the Lego pieces, the interfaces of the models should be standardized.

Schmidt and Werle define a standard as an "abstract specification of the necessary features of a component that make it compatible with the rest of a system - they ensure its fit" (Schmidt and Werle 1998, pg. 3). Keen considers standards as the "*key to model integration*" (Keen 1991, pg. 186).

The concept of model integration is not a new phenomenon. An excellent example of model integration is the telephone network, one in which billions of machines (telephones) exchange data more or less seamlessly every day due to the fact that there is a well defined model and an interface behind these systems. Although during decades different communication technologies and standards continuously evolved, such as advanced networks, new devices, analogue vs. dialog data, etc., we still integrate our devices to a network in a similar seamless way as we did decades ago. Model integration plays a very important role in the software engineering area as well. An example for this phenomenon is the recent introduction of the concept of *software component based development* (Szyperski 1998).

Distributed model integration is not a straightforward task, and providing an accepted standard would certainly increase the applicability of distributed model based problem solving strategy.

2.3 Problem Solving Strategy Based On Distributed Models

The application of distributed model based problem solving strategy generates several advantages. These benefits are:

- support for reusability of already existing submodels;
- support for combination of heterogeneous technologies;
- support for a collaborative model design and development process;
- support for information hiding.

Next we elaborate these four issues in more detail.

2.3.1 Reusability

During the problem solving strategy we might reuse existing parts of other models that suit our purpose. Reusing might lead to a more rapid development of the experimental model.

The economical benefit of reusability was recognized years ago by the software engineering community. Reese and Wyatt define *software reuse* as "isolation, selection, maintenance and utilization of existing software artefacts in the development of new systems" (Reese and Wyatt 1987). The potential of software reuse is formulated in McIlroy's law: "software reuse reduces cycle time and increases productivity and quality" (McIlroy 1969), (Endres and Rombach 2003, pg.77).

There are many approaches to reusability and levels on which reusability can be applied. Pidd identifies a spectrum of different types of software reuse (Pidd 2002, pg.772), which is depicted in Figure 2.3. The frequency arrow indicates the frequency of reuse in the different approaches. According to this arrow reuse is more frequently applied at the right end of the spectrum. The complexity arrow illustrates the grade of difficulty in the various approaches. Reuse can be more easily applied in approaches that are at the right hand side of the arrow. So we observe that when reusability can be easier applied it is more popular, like in code scavenging, while complex, full models are rarely reused.

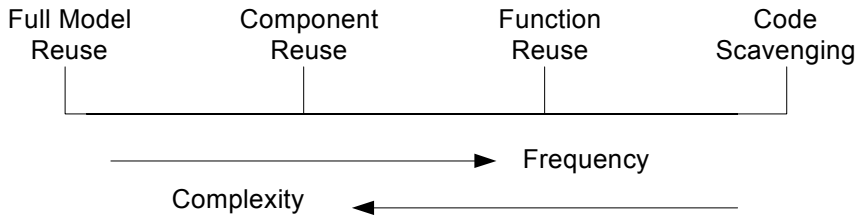


Figure 2.3 Reuse Spectrum (Pidd 2002)

According to Pidd's reuse spectrum (Figure 2.3) Code Scavenging is the way of reusing existing parts that is most often applied. This is probably what most of us used in learning how to model or how to program. We take something that appears to work and we use it to solve a new problem. Usually Code Scavenging is applied by the person who created the code in the first place. Function Reuse allows us to reuse not just a piece of code, but a service provided by an existing module. Function Reuse mostly refers to the use of previously defined modules or built-in functionality, like a generator module, for example, that generates entities based on a given probability distribution. Similar to this approach,

Component Reuse provides the possibility to reuse built-in components. Component Reuse refers to applying domain specific libraries, also called building blocks (Verbraeck, *et al.* 2002), (Oses, *et al.* 2003). The most complex and less frequent opportunity for reuse, at the leftmost side of the spectrum, is Full Model reuse, which refers to the reuse of a full model in its original form. This approach would be the most beneficial one economically. However, reusing a full model without any modification is feasible only if we intend to solve *exactly* the same problem or subproblem that the model was intended for. In most of the cases, however, we neither model the same system nor do we intend to solve the very same problem. Reusing a model for a purpose other than for which it was originally constructed requires modifications on the model itself (Fishwick 1995). In such a case an evaluation regarding the validity and credibility of the reused simulation model is in order (Balci 1997).

2.3.2 Heterogeneity

When building a model of a system certain system specification formalism is needed. The system specification formalism refers, as Zeigler defines, to "the types of modelling styles, such as continuous or discrete, that modellers can use to build system models" (Zeigler, *et al.* 2000, pg. 3).

Formalisms are proposed and accepted because they provide convenient means to express models for particular classes of systems and problems. Formalisms such as DEVS (Discrete Event System Specifications), Petri nets, State charts on discrete side and DESS (Differential Equation System Specifications), System Dynamics, Bond graphs on the continuous side are widely employed (Zeigler, *et al.* 2000), (Vangheluwe and De Lara 2002). However, many real-world phenomena cannot be fit into a single formalism at time. This is especially the case with complex systems, because such systems often have several components and aspects, the structure and behaviour of which cannot be described by a single comprehensive formalism. For example, Zeigler et al. point to an automated highway traffic control system, which cannot be modelled by a pure discrete or continuous paradigm (Zeigler, *et al.* 2000).

Consequently, when dealing with this issue, we either need to provide a tool which supports multi-formalism modelling or each system component may be modelled using the most appropriate formalism and tool (Vangheluwe and De Lara 2002). In this sense besides heterogeneous formalisms, we talk about heterogeneous tools in which the models are designed and developed. During distributed model-based problem solving strategy one can integrate submodels which are representations of system components and which follow heterogeneous formalisms and are built in heterogeneous tools.

2.3.3 Collaborative Design and Development

Recently, due to the growing interdependencies between organizations, there has been a rapid increase in interest in interorganisational networks (Daft 1998), (Mullins 2002) defined as "a collection of interacting organizations, groups, and persons" (Van de Ven*, et al.* 1980, pg. 22). Because of these interdependencies cooperation is needed. The concept of *teamwork* plays an important role when cooperating units or member organizations intend to accomplish a certain goal. Keen states that "teamwork is relational; the quality of the performance rests on the quality of interactions, communication, and coordination among team members" (Keen 1991, pg. 109). The basis for effective teamwork is *collaboration* or, as Schrage coins, shared creation (Schrage 1995). Keen defines collaboration as "a joint commitment to a target output, with team members sharing authority and responsibility as needed, at different stages and for different tasks" (Keen 1991, pg. 110). During collaboration "two or more individuals with complementary skills interact to create a shared understanding that none had previously possessed or could have come to on their own" (Schrage 1995, pg. 33).

Collaboration is a "purposive relationship" that facilitates solving problems in the context of constraints. These constraints are related to (Schrage 1995, pg. 29):

- *expertise* - one unit or member organization alone does not know enough to deal with the situation so others are needed;
- *time* - one unit or member organization cannot solve the problem in a certain time period and involvement of other units might reduce the time needed;
- *resources*[1] -one unit or member organization can use the resources of other units without investing in it;
- *competition* – in business organizations can feel the need to collaborate against other organizations.

Schrage's collaboration constraints do not necessarily require distributed modelling. Modellers can also cooperate on building monolithic models. Collaboration can be established within a small team consisting of just a couple of persons or by multiple teams representing different organizations. In most of the cases, as we perceived, small teams collaborate on designing and developing a monolithic model. However, in large scale projects that involve more organizations, each of them having its own team responsible for a certain part of the model, distributed modelling might be promising. In this thesis we focus on situations in which the models are designed and developed in a distributed way, by multiple, collaborating teams. The separately designed and developed models by different teams must be coupled into one big model. Distributed modelling has the advantage that it allows the modellers to work in parallel with other participants, to focus on the design of the certain parts of the model, and to design their submodels using their preferred framework. In this way, the choice for distributed models might speed up the specification step in the problem solving strategy, however, collaboration during carrying

[1] Schrage refers to it as "money"

out the projects is always needed because modellers have to agree on the details that will be exchanged between the coupled models. The existence of a standard framework certainly would speed up this process, however, certain level of collaboration will be always needed.

2.3.4 Information Hiding

During problem solving usually teams are formed that are responsible for solving a certain aspect of a problem, and for designing certain parts of the end model. Team members are often connected with and responsible for the individual operating goal of a specific department or organization of which they are a part. In the case of an interorganisational network that aims to solve a single problem, the final aim is accurate functioning of the interorganisational network itself. The team members are often fully involved in the sensitive business logic of their own organization and they are willing to share only information that is indispensable for the interorganisational relationship. The business logic of an organization comprises strategic information such as a product road map, information about business contacts such as suppliers, customers, etc. (Olson*, et al.* 2002)

Sharing business logic is a sensitive issue in collaborative interorganisational problem solving activities. In general there are many teams representing different organizations that need to collaborate whilst hiding relevant commercial sensitive information. Information hiding might be an issue within one organization as well, when, for example, the financial department is not willing to provide sensitive information to other departments.

Collaborating organizations, on the one hand, cannot leave out their classified information, such as confidential data or algorithms, because it might negatively influence the overall quality of the experimental model. On the other hand, they do not want to disclose this confidential information to the collaborative partners. Therefore, it is advantageous for an organization to design and develop its part of the experimental model separately in house, and integrate it with the models of the other organizations only at the end. The distributed modelling approach enables one to share only the required information and to keep confidential issues invisible for other parties.

In this section we have presented the benefits of designing and developing separate submodels applying a problem solving strategy based on distributed models. Furthermore, we introduced the concept of model integration as an essential notion within this strategy. In the next section we focus on simulation as a specific approach for designing and developing experimental models.

2.4 The Role of Simulation in Model Based Problem Solving Strategy

The model based problem solving strategy discussed in Section 2.2 can be based on several modelling approaches. While the analytical modelling approaches tend to focus on powerful abstract models and methodologies that managers find unrealistic and hard to follow, the simulation approach generally draws on more accessible methods (Simon 1965), (Keen and Scott-Morton 1978), (Law and Kelton 2000). In this thesis we focus on applying simulation modelling.

Several definitions of simulation exist. We follow the definition by Shannon: "the process of designing a model of a concrete system and conducting experiments with this model in order to understand the behaviour of the concrete system and/or to evaluate various strategies for the operation of the system" (Shannon 1975, pg. 2). As a tool to solve organizational problems Sol values simulation as a "powerful decision making aid" (Sol 1982). He further points to the fact that applying simulation models fits very well in the general model based problem solving strategy discussed here (Sol 1982).

Keen and Scott Morton also recognize the high applicability of simulation in decision making: "simulation is the most widely used managerial computer-based technique in both government and industrial decision making" (Keen and Scott-Morton 1978, pg. 45). Pidd identifies several advantages of the simulation approach instead of experimenting with the real system itself: simulation is *cheaper, safer, quicker and more secure* (Pidd 2003, pg. 226).

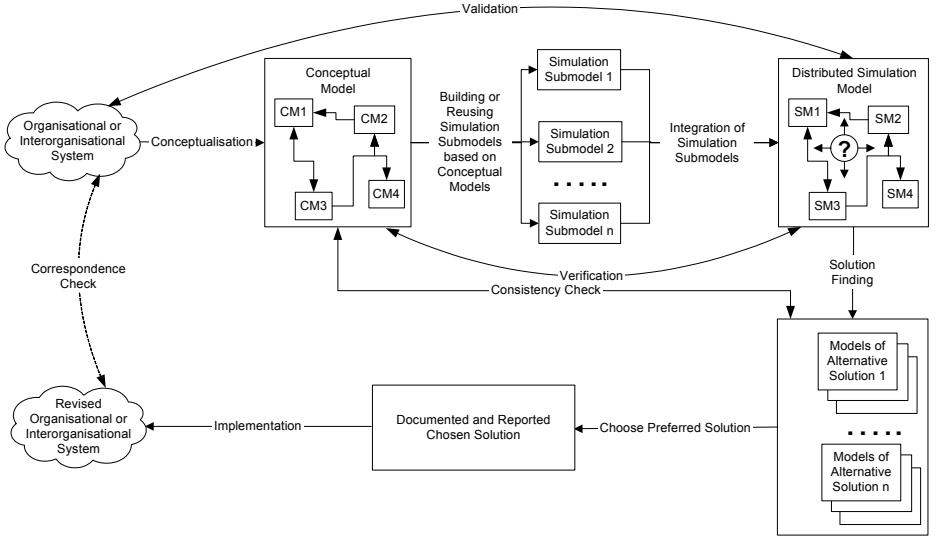


Figure 2.4 The Role of Simulation in Problem Solving Strategy Based on Distributed Models

The way simulation modelling fits into the framework of the organizational and inter-organizational problem solving strategy is depicted in Figure 2.4. This strategy extends the problem solving strategy based on distributed models presented in Section 2.3 and

illustrated by Figure 2.2 to the problem solving strategy based on distributed simulation models.

When using the simulation based problem solving strategy the problem owners provide a problem description to the simulation practitioners which can represent different organizations. The simulation practitioners might also formulate the problem themselves, but in that case the problem owners must understand and completely agree with the formulation of the problem. After obtaining conceptual models through the conceptualization process the modellers start creating their computer simulation models. During the specification process the simulation practitioners develop experimental models of the real problem situation by translating the conceptual models into computer simulation models. This process is called *simulation model development*. During the specification process the simulation practitioners might decide to reuse already existing computer simulation models, if they are suited for the required purpose.

Most computer simulation models in industry are created using simulation packages (Nikoukaran*, et al.* 1999). The most used and advanced simulation packages that are commercially available are referred to as commercial-off-the-shelf (COTS) simulation packages (Nikoukaran*, et al.* 1999), (Tewoldeberhan*, et al.* 2002). Law and Kelton identify several advantages of using COTS simulation packages rather than general purpose programming languages. They affirm that the increasing popularity of simulation modelling in the recent years is highly influenced by several improved and more widely used COTS simulation packages (Law and Kelton 2000). The industrial domain largely applies COTS simulation packages. In this research we will focus on integrating simulation models created in these packages.

Within the problem solving strategy based on distributed simulation models the simulation model consists of more integrated simulation submodels. The reasons why one might decide to design and develop a distributed simulation model are discussed in Section 2.3.

Before starting the experimentation using the distributed simulation models, experimental design techniques can be worked out that may reduce the required set of experiments for the problem diagnosis. An *experimental design* is the process of formulating a plan under which the simulation model is executed to produce the required information at minimal cost in a suitable form to enable the analyst to draw valid inferences (Shannon 1975). An *experiment* can be defined as "a set of replications with the same experimental conditions" (Sol 1982, pg. 47). A replication is defined as "one run out of a set of runs under the same treatment except for initialization conditions that provide statistical independence" (Sol 1982, pg. 47). Experimentations lead to several alternative simulation results that need to be evaluated.

In our research we focus on industrial simulation. Here one of the problems is that the above presented COTS simulation packages have been built to support monolithic simulation only. Although some of the packages provide partial solutions for integration, according to the literature and our observation none of the COTS packages is designed to support distributed development of the simulation models. In order to find ways to remedy this situation, we will discuss the theory on *distributed simulation modelling* in the remainder of this chapter. Distributed simulation is an application of distributed systems technology that enables models to be coupled together over computer networks so that they interoperate during a simulation run.

2.5 Distributed Simulation

In this section we aim to elaborate on the existing solutions for distributed modelling and the theory behind it in the specific area of simulation. In the first part of this section we present the distributed simulation theory for simulation model integration. Then, in the second part of the section we give an overview of the currently available tools and standards for distributed simulation.

2.5.1 Distributed Simulation Theory

The idea of integrating separately designed and developed models and/or software applications originates from distributed system theory (Tanenbaum and Van Steen 2002). This advanced technology has led to the fact that nowadays it is not only feasible, but easy, to put together more computers connected by a network. The literature calls these connected systems as distributed systems (Tanenbaum 1995).

Based on the results of the distributed system theory, the simulation community has formulated a *theory on distributed simulation models*. This theory provides solutions for data exchange, representation and time synchronization, that needs to be taken care of when the simulation practitioner intends to integrate simulation models (Fujimoto 2000), (Singhal and Zyda 1999).

Initial research on distributed simulation has been conducted in the military domain (Singhal and Zyda 1999). Large investments, long time periods and a big amount of money have been made available for this reason since distributed simulation seemed to be a promising approach allowing for the *interoperability* between several models and the *reusability* of them (Fujimoto 2000). Interoperability is a key concept within distributed simulation, defined as "the ability of a simulation model to provide services to and accept services from other simulation models, and to use the services so exchanged to enable them to operate effectively together"[2]. Besides interoperability and reusability Fujimoto discusses other benefits of distributed simulation, such as (Fujimoto 2000, pg. 5):

- *reduced execution time* - subdividing a large simulation computation into many subcomputations, and executing these subcomputations concurrently can reduce execution time;
- *geographical distribution* - executing the simulation program on a set of geographically distributed computers enables one to create virtual worlds with multiple participants that are physically located at different sites;
- *integrating simulation models from different vendors*, - instead of porting heterogeneous programs from different vendors to a single computer, it may be more cost effective to hook them together, each one executing its task on a different computer;

[2] DoD Dictionary of Military Terms: http://www.dtic.mil/doctrine/jel/doddict/

- *fault tolerance* - if one computer goes down, it may be possible for other computers to pick up the work of the failing machine, allowing the simulation computation to proceed despite the failure.

Distributed simulation entails simulation model integration. Next we identify the basic issues needed to integrate simulation models. We take a small and simplified example and try to identify all basic issues for simulation model integration.

To start with, assume that we have two simulation models that we would like to couple. Coupling is needed in order to answer a question that separately none of these models could answer, only together. As the models are aimed to solve together a problem, we assume that interdependencies exist between them (e.g., production flow, cash flow or information flow). The interdependencies between the models can be either unidirectional or bidirectional. Suppose for example, that the first simulation model provides data (e.g., product, cash, information) to the second model and the second model does not provide any data to the first model. In this case unidirectional interdependency exists between the two models. Since the second model does not provide any data to the first one there is no feedback between these two models. Due to the absence of feedback the two models can be executed independently and be coupled through an intermediary system. What we need to provide is temporary storage for the data produced by the first model and needed by the second one. These two simulation models follow a sequential process, in the sense that we first execute the first simulation model and we store its output results in intermediary storage (e.g., text file, spreadsheet, database), then we execute the second model that takes the output of the first model as part of its input. An example of such a kind of simulation model integration has been carried out by Babeliowsky, who designed and developed simulation models for improving the transfer baggage handling process at the Amsterdam Airport Schiphol. He applied this simulation integration technique for achieving interoperation between independently designed and developed individual airport terminal simulation models (e.g. check-in allocation model, reclaim allocation model, passenger flow model, etc.) (Babeliowsky 1997, pg. 261).

Unidirectional interdependency and no feedback is a very special situation that seldom occurs. In most of the cases bidirectional interdependencies exist between the simulation models, requiring feedback from one model to another at any simulation time. In such a case the simulation models cannot be executed sequentially, because future events of each of them might depend on events generated by another simulation model. The situation can be solved in similar way to the previous approach, applying intermediary storage, if the models produce data to other models in a well-defined order (e.g., the first model produces data, the second model takes it, then the second model produces new data for the first one, the first one takes the produced data and produces new data for the second one, and so on). This, however, is again a case that rarely occurs. Things get more complicated when models can send output and take input at *any time* and can trigger several other events in any of the other simulation models.

The dependency between the simulation models can be solved in this case by an event passing mechanism. As events might occur at any time, it is expedient that each simulation model processes all of its events, both those generated for and by itself (locally) and those generated by other models (globally) in the order of their occurrence (time stamp order). Failure to do this could cause an event to affect another event in the past, clearly an

unacceptable situation. Errors resulting from out of order event processing are referred to as *causality errors*, and the general problem of ensuring that events are processed in the right order is referred to as the *synchronization problem* (Fujimoto 2000).

As illustrated above, simulation literature identifies two essential basic functional issues for solving simulation model integration (Fujimoto 2000):

- Data Exchange; and
- Time Synchronization.

As far as data exchange is concerned there is the additional issue of data representation. Two models that are interdependent need to understand each other. They need to exchange some data and have a correct interpretation of the data they send and receive. This requires that the participants agree on a common interpretation of data which is produced and exchanged between them. Additionally, it is necessary to analyze the data types used internally by the simulation models (Straßburger 2001). Distributed simulation theory covers different data distribution mechanisms that are essential for data representation and exchange. Most of these mechanisms (e.g. Publish/Subscribe) are derived from distributed system theory (Tanenbaum and Van Steen 2002).

The simulation clock that controls simulation time during execution of a simulation model resides within each simulation model itself. When integrating simulation models there is a need for synchronization of time between the different models. Time synchronization is required for simulation models that use different time advancing mechanisms as well. It might be the case that someone must integrate discrete-event simulation models, continuous simulation models and even real-time simulation models. Distributed simulation theory describes various time synchronization mechanisms, such as conservative, optimistic, time-stepped, hybrid and so on, and provides several approaches (e.g. algorithms) for implementing these time synchronization mechanisms (Fujimoto 1998b), (Fujimoto 2000).

In order to integrate different simulation models there is a need for a common framework, which provides distribution services. Such a framework can be based on distributed simulation theory (Singhal and Zyda 1999). In order to create such a common framework, middleware from general software domain, such as CORBA, DCOM or RMI (Buss and Jackson 1998), (Verbraeck*, et al.* 2000) might be considered. These approaches are general enough for any data exchange between software applications; however, they do not include all services needed (e.g. time synchronization).

The U.S. Department of Defence (DoD) made large investments in the distributed simulation area and is active in developing distributed simulation standards that are designed to facilitate interoperability of simulations that are distributed over a computer network. The aim of these standards is to provide a common framework that involves the required distributed simulation services. In the next section we present three common frameworks that are designed and developed by the military domain for this purpose.

2.5.2 Simulation Integration: Tools and Standards

In order to carry out distributed simulation, different simulation standards were initiated in the military domain, such as SIMulator NETworking (SIMNET) (Pope 1989), Distributed Interactive Simulation (DIS) (DIS Steering Committee 1994) and Aggregate Level Simulation Protocol (ALSP) (Weatherly*, et al.* 1991). Most of these standards provided specific services for interoperability in a small niche of the simulation market, for example DIS for human-in-the-loop simulators or ALSP for war games (Straßburger 2001). Finally, in 1996 the Defence Modelling and Simulation Office (DMSO) presented the *High Level Architecture (HLA)*, which is the most recent and most advanced approach for integrating simulation models (DMSO 1998a), (DMSO 1998b), (DMSO 1998c), (Kuhl*, et al.* 1999). In this section we present the continuous effort made by the military domain for creating a standard for integrating simulation models.

2.5.2.1 The Distributed Interactive Simulation Standard (DIS)

The Distributed Interactive Simulation (DIS) is a networking protocol standard, the primary mission of which is to define an infrastructure for linking simulation models of various types at multiple locations to create realistic, complex, virtual "worlds" for the simulation of highly interactive activities (DIS Steering Committee 1994).

The development of DIS started in 1989 and was based on the SIMNET (Miller and Thorpe 1995), (Pope 1989) program. DIS was under development for about 10 years, which led to final standardization, as IEEE Standard 1278 (DIS Steering Committee 1994). DIS has the same three basic components as SIMNET: an object event architecture, the notion of autonomous distributed simulation models, and an embedded set of predictive modelling algorithms for dead reckoning (DIS Steering Committee 1994), (Pope 1989). The DIS object event architecture has been designed to cover a broader spectrum of military simulation requirements than SIMNET (Singhal and Zyda 1999).

The core of the DIS network software architecture is the Protocol Data Unit (PDU). The PDUs deal with data transfer among different simulation models. Each time the state of an entity changes within the simulation model a PDU is sent because the changes might affect the other simulation models.

The DIS considers and uses the following design principles (Fujimoto 2000), (DIS Steering Committee 1994):

- *No central computer controls the entire simulation exercise.* DIS uses a distributed simulation approach in which the responsibility for simulating the state of each entity (tank, submarine, carrier, aircraft, missile, etc.) is attached to separate simulation applications residing in computers communicating via a network.
- *Autonomous simulation models.* This principle emphasizes that each integrated simulation model is considered as an autonomous part in the whole distributed system. Simulation models are autonomous in the sense that each simulation model advances its simulation time according to a real-time clock. They do not need to determine which other model will request a PDU. Instead they use a

broadcasting mechanism and the receivers are responsible to capture the relevant messages. This autonomy principle enables simulation models to leave or join a distributed simulation execution which is in progress, without disrupting the rest of the simulation models.

- *Transmission of "ground truth" information.* Each simulation application communicates the absolute truth about the state of the entity it controls (location, orientation, velocity, articulated parts position, etc.) to other simulation models on the network. The receiving simulation model is responsible for taking this ground truth data into consideration.
- *Transmission of state change information only.* In order to reduce network traffic, the simulation models only transmit changes in behaviour. For example if a vehicle moves in a straight line with constant velocity the rate at which state updates are transmitted is reduced. In such a case the simulation model transmits only "keep alive" messages.
- *Use of "dead reckoning" algorithms to extrapolate entity state information.* Each node maintains information concerning other entities, such as those that are visible to it on the battlefield. This information is updated whenever the entities send new status information via PDUs. In between state updates, all simulators use common algorithms to extrapolate the current state (position) of other entities between state updates, based on previously reported information.

Achieving these design principles "DIS was an enormous success for real-time networked virtual environments" (Singhal and Zyda 1999, pg. 26), although for a comprehensive applicability the simulation community identified several disadvantages. Straßburger specifies the following disadvantages of the DIS concept (Straßburger 2001, pg. 19):

- With the definition of application specific data packets, a fusion between the architecture and the application is performed. This is a major disadvantage, because it limits the applicability and flexibility of the standard.
- The broadcast mechanism places a heavy burden on the underlying network. The common DIS technology of applying dead reckoning to overcome this problem requires the usage of certain algorithms inside the simulator. Thus, the architecture places constraints regarding the internal implementation onto the participating simulators.
- DIS synchronization is limited to real-time simulations. Simulation models based on other time advance mechanisms cannot participate in a DIS exercise.

Although DIS has several disadvantages, the simulation community considers it as the first one of the major standards achieved in the field of distributed simulation. It is considered as a predecessor of the High Level Architecture, one of the most advanced distributed simulation approach at this time. The High Level Architecture presented later focuses on eliminating the deficiencies of the DIS standard.

2.5.2.2 The Aggregate Level Simulation Protocol (ALSP)

The second major development that originates from SIMNET is the Aggregate Level Simulation Protocol (ALSP). ALSP is a project designed to permit multiple, pre-existing war game simulation models to interact with each other over local or wide area networks. In concept it was patterned after SIMNET (at that time DIS) where each simulation model controls its own objects and shares information about them with other simulation models (Weatherly, *et al.* 1991), (Fujimoto 2000). The ALSP concept was initiated in January 1990 when DARPA (Defence Advanced Research Projects Agency) sponsored MITRE (Massachusetts Institute of Technology Research Establishment) to investigate the design of a general interface between large, existing, aggregate level combat simulation models to be used for military training simulations (Weatherly, *et al.* 1991). In an aggregate level simulation model the primary objects are collections of doctrinally identifiable military assets, e.g. tank battalions (Page and Smith 1998). After an analysis of the DIS principles, the designers of ALSP concluded that aggregate level simulation models had unique requirements beyond those of DIS. These requirements concern (Seidel 1993):

- *Time management.* Aggregate level simulation models evolve time in a manner that is not directly tied to the wall clock time. These simulation times must be coordinated so that the times for all simulation models would appear the same to users and so that event causality would be maintained, that is, events will occur in the same sequence in all simulation models.
- *Data management.* Each simulation model uses its own representation of data. A method must be devised to permit simulation models to share information in a commonly understood manner.

The goal of MITRE was to design a message-based protocol derived from the experience of prototyping an interface between several real world training simulations. DARPA sponsored the design of a protocol in order to provide a vehicle by which various training organizations could cooperate, increasing the functionality of their individual simulations (Weatherly, *et al.* 1991). The protocol aims to guarantee causality by applying a conservative time synchronization mechanism based on the basic Chandy-Misra algorithm (Chandy and Misra 1979).

The ALSP approach introduced two new terms in the field of distributed simulation: confederate and confederation. The different simulation models are referred to as *confederates* and they form a common distributed simulation called *confederation* (Wilson and Weatherly 1994). ALSP provides time management services to coordinate simulation times and preserve event causality across simulations. The ALSP Common Module (ACM) provides data and time management services to each confederation member. The ACM coordinates joining and departing from the confederation, coordinates simulation local time with confederation time, filters incoming messages so that only those of interest are received by the simulation and coordinates and enforces the ownership of object attributes (Weatherly, *et al.* 1991).

The ALSP approach made a great step towards creating a standard for distributed simulation by supporting interoperability among heterogeneous systems and introducing the time management concept. In order to improve and extend the solutions (e.g. time management, data distribution management, etc.) provided by this approach the military community in 1996 initiated the High Level Architecture project. As it stands now, ALSP can in fact be considered as a subset of HLA (Straßburger 2001).

2.5.2.3 The High Level Architecture (HLA)

As we have seen in the previous sections, DIS was developed to support the interoperability of entity level combat simulation models (singular military objects, e.g. a tank or a soldier) and ALSP was designed to support the interoperability of aggregate level combat simulation models (e.g. a tank battalion). The United States Department of Defence (DoD) believed that it would be more efficient to develop a more general solution to distributed simulation. Design and development of this general solution was initiated by the Defence Modelling and Simulation Office (DMSO) where a group called the Architecture Management Group was formed, which started the project in 1993 and finished it in 1996. The end result of this project was renamed as *High Level Architecture* (DMSO 1998a), (DMSO 1998b), (DMSO 1998c). The roots of the High Level Architecture (HLA) stem from DIS, aimed primarily at training simulations, and ALSP which applied the concept of simulation interoperability to war gaming simulations (Fujimoto 2000). In September 1996 the Under Secretary of Defence, Acquisition and Technology designated HLA as the standard technical architecture for all DoD simulations. This entailed that, from then on all simulations, developed by, or for the US DoD had to be HLA compliant.

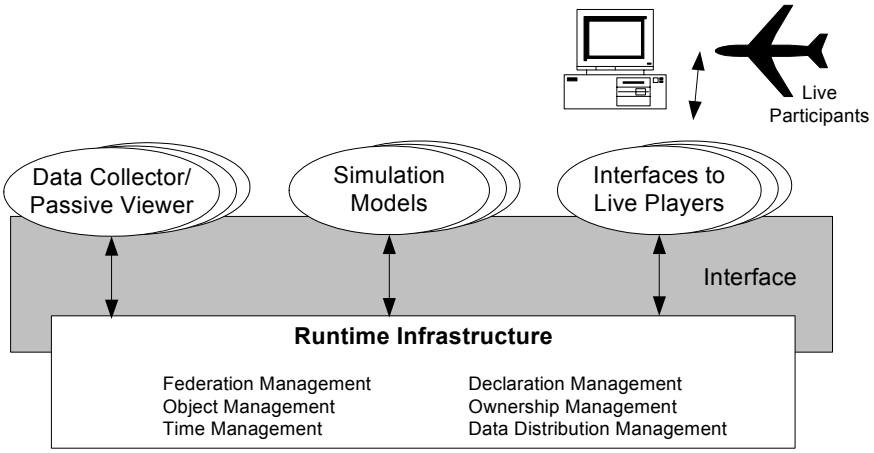


Figure 2.5 Functional View of an HLA Federation (Dahmann, *et al.* 1998)

Figure 2.5 illustrates an HLA based distributed simulation system which can be partitioned in three different major components (Dahmann, *et al.* 1998):

- One of the major notions in the HLA standard are the simulation models or, as HLA calls them, the federates. The HLA standard defines the notion of *federate* and *federation*, corresponding more or less to confederates and confederations from ALSP. A federate can be a computer simulation, a manned simulator, a supporting utility, such as a viewer or data collector or even an interface to a live player or instrumented facility. A federation is the alliance of one or more federates acting together in a distributed simulation to achieve a certain objective.
- The *Run Time Infrastructure (RTI)* functional component behaves like a distributed operating system for a federation. It provides a set of general purpose

services that support federate to federate interactions and federation management (DMSO 1998b). The interactions that take place between two federates are performed through the RTI. RTI includes six services: Federation Management, Object Management, Time Management, Declaration Management, Ownership Management and Data Distribution Management (DMSO 1998b).

- Another main component within the HLA standard is the *interface* to the RTI. The HLA run time interface specification provides a standard way for federates to interact with the RTI, to invoke the above mentioned RTI services, to support run time interactions among federates and to respond to requests from the RTI.

The High Level Architecture was adapted as an IEEE 1516 standard on September 26, 2000. The HLA standard is formally defined by three components:

- *The Object Model Template* (DMSO 1998c), (IEEE 2000c);
- *The Interface Specification* (DMSO 1998b), (IEEE 2000b);
- *The HLA Rules* (DMSO 1998a), (IEEE 2000a).

The HLA Object Model

An HLA object model provides a formal definition of the data that is transferred between federates (DMSO 1998c). Furthermore, HLA defines an Object Model Template (OMT), as the interface language between the federates. Although HLA applies an object-oriented world view for representing the data that is transferred between federates, this view differs from the well-know object-oriented view defined in the software engineering domain in the sense that HLA object models do not specify methods for the objects. It should be noted that this object-oriented world view does only define how federates have to represent themselves to other federates (Kuhl*, et al.* 1999), (Straßburger 2001). The OMT defines two classes for this reason: object classes and interaction classes. While the object classes describe the simulated entities with their attributes, the interaction classes describe the interactions between different objects (DMSO 1998c).

The simulated entities or objects within the object classes are characterized by the following issues (Kuhl*, et al.* 1999):

- they define a set of named data called *attributes;*
- they are of interest to more than one federate and thus handled by the Run Time Infrastructure;
- they persist or endure for some interval of simulated time.

An interaction within the interaction class is a collection of data sent at one time through the RTI to other federates. An interaction has the following features (Kuhl*, et al.* 1999):

- it carries a set of named data called *parameters;*
- it may represent an occurrence or event in the simulation model of interest to more than one federate;
- it has no continued existence after it has been received.

During simulation run the distributed models interoperate with each other by means of exchanging simulated entities or interactions. Therefore, each distributed model should have a representation of the objects (entities and interactions) it provides and respectively expects. For this reason, each individual federate provides a so called simulation object model (SOM), which has to be specified in terms of the OMT (DMSO 1998c), (IEEE 2000c). The SOM describes the simulation (federate) in terms of the types of objects (attributes) and interactions (parameters) that it can provide or will expect from other federates. The SOM is distinct from internal design information; rather it provides information on the capabilities of a simulation to exchange information as part of a federation.

Furthermore, each federation defines a so called federation object model (FOM). The FOM is a superset of the information from the individual SOMs of the federates, which contains all classes defined by the individual participants of the federation and gives a description of all shared information (DMSO 1998c), (Straßburger 2001). Similarly to a SOM, a FOM is specified in terms of the standard OMT.

There are tools available for developing SOM and FOM object models. One of these tools is called Object Model Development Tool (OMDT). Using this tool one can create the SOMs and FOMs and then automatically generate a Federation Execution Data file, which is required by the Run Time Infrastructure for data exchange.

The HLA Interface Specification

The HLA Interface Specification describes the services provided to the federates by the Run Time Infrastructure, and by the federates to the Run Time Infrastructure (DMSO 1998b), (IEEE 2000b). This bidirectional service providing (RTI to federate and federate to RTI) is carried out applying the ambassador paradigm (Figure 2.6). While an RTI communicates with a federate through its federate ambassador, the federates communicate with the RTI through the RTI ambassador. The federate and RTI ambassadors behave like objects and communication among the participants is performed by calling "methods" of these objects. The HLA interface specification defines these methods both for RTI and federate ambassadors (DMSO 1998b).

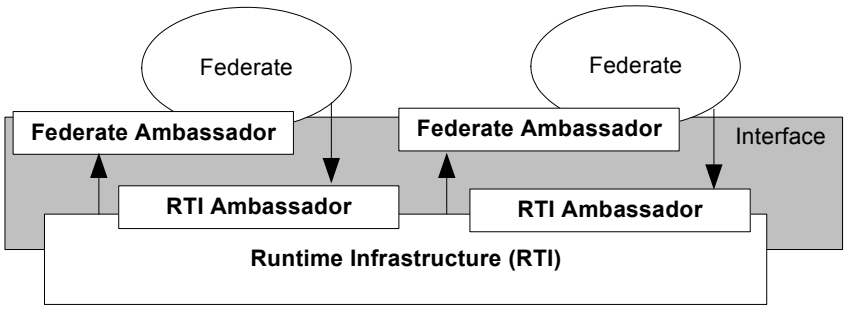


Figure 2.6 The Ambassador Paradigm (Kuhl, *et al.* 1999)

The Run Time Infrastructure behaves like an operating system in which the different federates are embedded. It is a piece of software that provides all services that are needed for distributed execution. These services can be triggered by the RTI and the results of the services are transferred to the federates through the Federate ambassadors. The RTI provides six services: Federation management, Declaration Management, Object Management, Ownership Management, Time Management, and Data Distribution Management.

The *Federation Management* services are essential for the coordination of the federation activities during federation execution. These services offer basic functions for initiating, joining, resigning and managing a federation execution.

The *Declaration Management* services control the way federates declare their intent to produce (publish) or consume (subscribe) data. The data must be in accordance to the SOMs and the FOMs.

The *Object Management* services are used for data exchange. A federate uses these services to register new instances of an object class, to update its attributes, and to send and receive interactions.

The *Ownership Management* services are used by federates and the RTI to transfer ownership of objects among federates.

The *Time Management* services support time synchronization. Time advances are coordinated by the RTI with Object Management services so that data is delivered to federates in a causally correct and ordered fashion. The Time Management services perform two main tasks:

- they allow each federate to advance its simulation time in coordination with other federates;
- they control the delivery of time stamped events in such a way that a federate will never receive events from other federates lying in its past, meaning events with simulation time less than its own simulation time.

The Time Management services offer both conservative and optimistic time synchronization (Fujimoto 1998b).

The *Data Distribution Management* services provide mechanisms for efficient routing of data among federates during the course of federation execution. This service intends to improve the mechanisms applied in the previous technologies (e.g. DIS) where data distribution was based on broadcast principles only.

Currently the Run Time Infrastructure implements these HLA Interface Specification services in CORBA IDL, C++, ADA and Java (DMSO 1998b).

The HLA Rules

The third component of the HLA standard consists of the HLA rules. The HLA rules summarize the key principles behind the HLA. The rules are divided into two groups: there are five rules for federations and five rules for federates. Rules related to federations state, for example, that federations shall have an HLA Federation Object Model (FOM), specified in terms of HLA Object Model Template (OMT), or, that during a federation execution, FOM data among federates shall be exchanged via the RTI. Similarly, a rule for federates, specifies, for instance, that federates shall have an HLA Simulation Object Model (SOM), documented in accordance with HLA Object Model Template (OMT). The complete list of rules can be found in (DMSO 1998a) and (IEEE 2000a).

Besides designing and developing HLA another goal was to establish a strategy by which HLA federations can be built. This strategy is worked out in the Federation Development and Execution Process (FEDEP). The Federation Development and Execution Process is set up as an "iterative waterfall software engineering process" to support development of HLA federations (Lutz*, et al.* 1998). The FEDEP process follows the basic idea behind the simulation model based problem solving strategy based on distributed models discussed in Section 2.4 and depicted in Figure 2.4. According to DMSO: "currently FEDEP represents the best practice available to the HLA community for creating HLA compliant federates and federations" (DMSO 1999, pg. 23).

2.6 Concluding Remarks

This chapter aimed to present the state of the art of simulation model integration as a form of problem solving strategy. Before focusing on simulation model integration, we have introduced some general concepts, such as systems, models, organizations, problem solving, model-based problem solving strategies, model integration, and simulation modelling.

For solving organizational and interorganisational problems we have described a model based problem solving strategy. Then we extended this strategy to the case where the experimental model consisted of integrated submodels. We also investigated the possible benefits of model integration within this extended strategy. Next we introduced the concept of simulation modelling and we embedded it in the extended problem solving strategy. This established the concept of simulation model integration.

After that we presented the notion of distributed simulation as an approach to solve integration of simulation models. We described the evolution of distributed simulation tools and standards and we presented three advanced approaches for distributed simulation: DIS, ALSP and HLA. The three approaches presented above are the most advanced solutions for distributed simulation. Interestingly, they all originate from the military domain. From these three approaches HLA is by far the most mature approach. Straßburger compared the HLA standard approach with other distributed approaches, such as CORBA, RMI, DIS and ALSP, and he found that "The comparison of HLA with similar techniques reveals, that although HLA might not be perfect, it is the most advanced technology for interoperability among simulations currently available" (Straßburger 2001,

pg. 24). HLA has gathered ground in defence and finally became "the standard technical architecture for all Department of Defence simulations" (DMSO 1998b, pg. 1-2). This led to the fact that various departments started to develop their own version of RTI that aimed to be more efficient than the DMSO-RTI. Lately, other more efficient commercial available RTI's appeared, such as pitch RTI[3] or MÄK RTI[4].

Given the fact that distributed simulation is taking off in industry and that HLA is a standard and the most advanced approach for distributed simulation, some researchers have recognized the potential for migrating the HLA concept into the industrial domain (Straßburger 2001), (Rabe*, et al.* 2001), (McLean and Riddick 2000), (Revetria*, et al.* 2003).

Although this led to some new insights regarding the applicability of HLA in industry, it seems that it is still rarely applied as it stands now. This is our personal observation and it certainly needs to be validated. In the thesis we deal with this observation, and accordingly we aim to test whether our observation holds. The possibility that a standard tool for simulation integration exists that is hardly applied in industrial cases triggers several questions that need to be answered. Thus, if we indeed find support for the proposition that HLA is hardly applied in industry, then we need to study why this is the case, what the reasons are behind not applying such a mature solution. Is the tool not appropriate for industry, or is it the case that the industrial community does not have enough knowledge about it, or cannot apply it? Finally, if our assumption can be validated, and we can reveal some of the reasons behind it, we might be able to suggest an approach that can avoid and solve the problem. In order to validate our observation and to obtain a global view on the application and applicability of HLA in industry we carry out a survey research which is presented in the next chapter. The survey focuses not only on the specific case of HLA, but also on the views on possibilities for distributed simulation in general for industry.

[3] http://www.pitch.se/

[4] http://www.mak.com/

3 A SURVEY ON DISTRIBUTED SIMULATION IN INDUSTRY

3.1 Introduction

In the previous chapter we introduced some theory on simulation integration and presented several tools and standards, based on distributed simulation concepts, which are useful for the practitioner who wishes to accomplish appropriate simulation model integration. As we have concluded, currently the most advanced approach for integrating distributed simulation models is the High Level Architecture. However, we also observed that this approach is almost completely absent from the industrial domain. This chapter and the next one aim to validate this observation and if it is indeed true try to find the reasons behind it.

We start by stating the observation as a hypothesis. In order to test the hypothesis and to find the reasons behind the phenomenon, if turns out to be true, we generate three questions. In order to answer these questions we conduct a survey research which has an iterative character. In the first place a quantitative investigation is carried out, the primary aim of which is to validate the hypothesis. If this is successful, in the next iteration, a qualitative investigation is conducted which takes as a basis the results of the quantitative research and aims to discover the reasons why the hypothesis is true.

This chapter describes the survey consisting of both the quantitative and qualitative investigations presenting the method used for selecting the participants, for data collection, and for data analysis. Furthermore, this chapter provides a structured representation of the data that we have collected during the survey. The final answer for the three questions will be provided in the next chapter, which contains mainly the evaluation of the collected data.

3.2 Motivation for the Survey

The previous chapter covered the theory on distributed simulation and presented the most advanced approaches for distributed simulation. We concluded that the most appropriate approach, which is accepted as the de facto standard for integrating simulation models is the High Level Architecture (HLA).

Originally the High Level Architecture was designed and developed by the US defence community, namely Defence Modelling and Simulation Office (DMSO), for integrating all kinds of existing and prospective defence oriented simulation models (Kuhl*, et al.* 1999). The HLA was designated to be the standard for all simulations of the Department of Defence (DoD) in 1996 (DMSO 1998a) (DMSO 1998b), (DMSO 1998c) and in 2000 it was accepted as the IEEE 1516 standard for distributed simulation (IEEE 2000a), (IEEE 2000b), (IEEE 2000c).

The evolution of distributed computing, the continuously increasing complexity of large-scale systems that need to be simulated, and, last but not least, the HLA standard initiated a new stream of activity in simulation, namely the *interoperation of simulation models*. Large organisations are interested and involved in this new stream, such as the US

Department of Defence (DoD), the Defence Modelling and Simulation Office (DMSO), the North Atlantic Treaty Organization (NATO) [5], the Simulation Interoperability Standards Organizations (SISO), IEEE.

Simulation interoperability is facilitated by SISO[6] across a wide spectrum by providing forums to educate the modelling and simulation community. SISO also supports the development of standards. Furthermore, it organizes large conferences and workshops on simulation interoperability, such as the semi-annual Simulation Interoperability Workshop (SIW) and the annual European Simulation Interoperability Workshop (ESIW). The first one "typically has a total attendance of 450 - 500 people each fall and spring"[7], which is a large number compared to the one of the largest annual simulation conferences, Winter Simulation Conference (WSC)[8], where the total attendance in the last three years varied between 500 and 550[9]. The difference is even more telling, if one realizes that WSC covers the whole range of simulation topics, while SIW focuses on simulation interoperability only. Furthermore, recently various Product Development Groups and Study Groups have been started under the umbrella of SISO that aim to investigate simulation interoperability issues from different perspectives.

While generally within the defence domain we can see a big drive in interoperability of simulation models and common application of HLA, we are not aware of much effort in industry. The reason behind this phenomenon is not clear, given the various benefits, such as the possibility to reuse existing components, support for information hiding, and support for integrating heterogeneous models. Although the initial step in designing and developing the HLA standard was carried out by the defence community, this large effort was intended to support the industrial community as well. The research community also observed that HLA is rarely applied in the industrial domain. In order to find out the reasons behind this phenomenon, in the last two consecutive years separate panel discussions were organized at the Winter Simulation Conference (Taylor*, et al.* 2002), (Taylor*, et al.* 2003a). Furthermore, a forum, called HLA-CSPIF[10] was initiated that aims to study the applicability of distributed simulation in industry.

We assume that most of the industrial simulation projects are carried out in COTS simulation packages. This assumption is based on a literature review and on discussions with simulation practitioners at various conferences and workshops, such as the Winter Simulation Conference, European Simulation Interoperability Workshop and Europeans Simulation Symposium. Certainly this assumption should be validated.

Our observation is that COTS simulation package vendors hardly support HLA, or even distributed simulation, in their product. This might be a reason why distributed simulation

[5] HLA recently become a NATO standard – STANAG 4603

[6] http://www.sisostds.org/

[7] Information provided by Duncan Miller, SISO Executive Director

[8] http://www.wintersim.org/

[9] Information collected from http://www.wintersim.org/wschistory.htm

[10] The HLA Commercial-Off-The-Shelf Simulation Package Interoperability Forum - http://www.cspif.com

or HLA cannot be found in the industrial market. Assuming that most of the simulation practitioners apply COTS simulation package for their daily use, they do not have a chance to apply distributed simulation.

On the other hand the simulation practitioners do not seem to request facilities, like transparent HLA interfaces, for distributed simulation. This might be caused by the fact that simulation practitioners cannot see the benefits because they do not have the proper tool for it. So basically this is a chicken egg scenario. However, this is again an assumption that needs to be validated.

All these statements and assumptions relate to our observation that we posed at the concluding remark section of the previous chapter. We would like to state this observation as a hypothesis which needs to be validated:

Hypothesis:

The HLA standard is hardly applied in industry.

In order to validate this hypothesis and the assumptions that led to this hypothesis we intend to conduct a survey. The validation process aims to address the following survey question:

Survey Question 1: *Is it true that the HLA standard is hardly applied in industry?*

The answer to this question can be either affirmative or negative. Based on our observations we expect a positive answer which then leads to two additional questions:

Survey Question 2: ***Why*** *is the HLA standard hardly applied in industry?*

Survey Question 3: ***How*** *can we solve this problem?*

In order to have a proper evaluation of the applicability of distributed simulation, respectively HLA, in the industrial domain we try to collect information not only from the industrial but also from the defence community, which has designed and developed HLA and uses it in its projects. Accordingly, the participants of this survey are selected both from the industrial and the defence simulation community.

In the next part of this chapter we present the survey in more detail.

3.3 Overview of the Applied Methodology

The method of the survey follows the Delphi approach, the purpose of which is to elicit information and judgments from participants to facilitate problem solving, planning, and decision making (Linstone and Turoff 1975), (Linstone 1978). Delphi is a method that proves to be particularly useful when the individuals who need to interact cannot be brought together in a face-to-face exchange because of time or cost constraints (Kenis 1995). Using this method information is exchanged via mail, FAX, or e-mail. In our survey we exchange information via e-mail and through phone interviews.

The Delphi method is iterative. The results of an initial survey are summarized and then form the basis of a second follow on questionnaire. Results from the second questionnaire are the basis of a third questionnaire and so on (Linstone and Turoff 1975). Accordingly, after getting the preliminary results, we maintain a continuous relation with the involved experts in follow up sections in order to gather sufficient feedback for the validation of the end result (Figure 3.1).

In order to obtain the answer to the first survey question, namely whether it is true that HLA standard is rarely applied in the industry, we tried to identify a community which has continuous and intensive contact with the industrial simulation area. The most appropriate group for this purpose would be the totality of software companies that carry out industrial simulation projects. However, it is impossible to identify all companies that develop these types of projects. Therefore, even if we identify some companies, they cannot be used to properly test the hypothesis because we cannot generalize their answers if we do not know how many other projects are conducted out there. As a consequence, the community we have recognized as appropriate for testing our hypothesis consists of the COTS simulation package vendors who provide tools for their customers in the industry. COTS vendors adapt their packages to the market needs they perceive. They have a continuous and intensive connection with industrial simulation practitioners who request new features in the package based on the projects they intend to carry out. In this way the COTS vendors have an opportunity to gain information about the projects that their customers carry out. Consequently, these vendors are in the position to have an overview regarding the application of the HLA standard within industry. Therefore, our primary aim is to mobilize this group for providing an answer to our first survey question. Furthermore, COTS vendors can provide information regarding the support for HLA within their tool.

Although we select COTS vendors to focus primarily on the first survey question only, they can give some indications regarding the expected answers to the second and third survey questions as well. The final answers to the second and third survey questions are expected from a selected group of people that we consider experts in distributed simulation and who have already had experience with distributed simulation either in industry or defence.

After selecting the participants the researcher must identify the collection procedures that he intends to apply. The literature mentions four basic ways of data collection: observation, questionnaire or interviews, documents, and media (audio and visual) material (Creswell 2003), (Seale*, et al.* 2004).

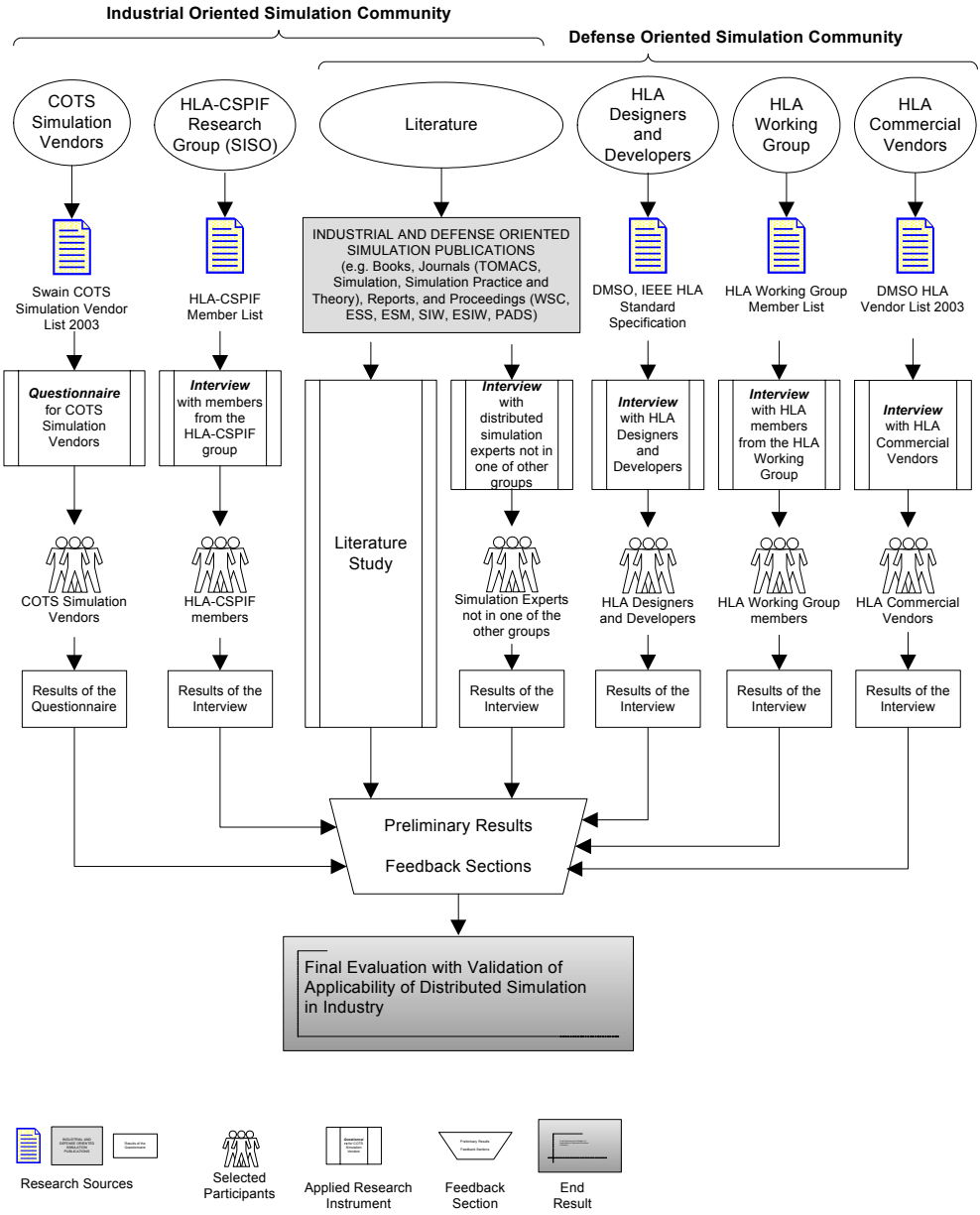


Figure 3.1 Survey on Distributed Simulation in Industry

Due to the fact that there is a large number of COTS simulation package vendors active in the market and that the answer to the first survey question can be shortly and easily formulated, the type of the survey that we have chosen for our initial investigation is a structured questionnaire. In contrast to the first question, the second and third survey questions assumingly generate more extensive answers and for this reason we have chosen for an open ended interview in the second round. To some extent the interview is designed based on the results of the questionnaire survey.

We apply thus two types of data collection: firstly a structured questionnaire for COTS simulation package vendors, and secondly an open ended interview based on our observations and the results of the questionnaire.

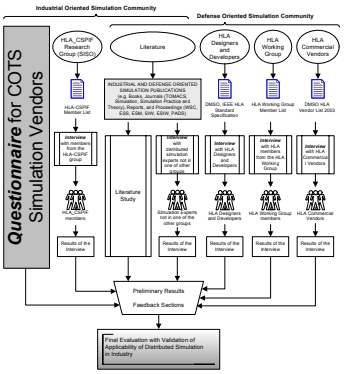
The primary aim of the data collected from the vendors from the questionnaire is to validate our hypothesis. This data is analyzed quantitatively (Creswell 2003). The way the vendors are selected and the methodology used for data collection are further elaborated in Section 3.4. In the same section we present the methodology for data analysis and we present the analyzed data. Based on the interpretation and evaluation of the collected data we draw then the first conclusions regarding the validation of our hypothesis. We expect that the data collected from the questionnaire survey will provide enough evidence to validate our hypothesis.

The interview survey aims to go one step further and answer the second and the third survey questions. As we have stated before, the interview survey is based to some extent on the questionnaire survey and accordingly, it can be considered as a second iteration in this research. For this survey we employ purposeful selection in order to determine the participants. The data in this case is collected through an open ended phone interview with experts on distributed simulation both from industry and defence. In order to analyze the data collected from the interviews we apply qualitative content analysis (Creswell 2003), (Seale*, et al.* 2004). The way of selection as well as the methodology applied for data collection and analysis is discussed in more detail in Section 3.5. In the same section we present the collected data that will be evaluated in the next chapter.

A graphical overview of the survey is given in Figure 3.1. It will be further elaborated in the next sections.

3.4 The Questionnaire for the COTS Simulation Vendors

In this section we aim to describe the questionnaire for the COTS simulation package vendors. The questionnaire is the first step of the methodology as highlighted next to this paragraph by the grey box at the left side of the figure that is in fact a snapshot of Figure 3.1. In the first part of this section we present the way the survey is designed. Then we discuss how the participants are selected. Further, the data collection mechanism is covered in the third part of the section, while the method applied for the analysis of the data is described in the fourth part. Finally, we summarize and evaluate the collected data.



3.4.1 Survey Design

The design of the questionnaire survey should be in consonance with the aim of the questionnaire. As we have specified before the primary aim of the questionnaire is to test and strengthen our hypothesis by providing an answer to the first survey question. Additionally, the questionnaire should provide a preliminary indication on some of the possible answers to the second and third survey questions as well.

The questionnaire is designed in accordance with the three survey questions stated above, and, accordingly, it consists of three parts. The first part of the questionnaire deals with the first survey question, namely: Is it true that the HLA standard is hardly applied in industry? Before asking specifically about the application of the HLA standard we intend to find out whether simulation packages support integration with external applications in general and whether the vendors are aware of any distributed simulation projects in the industry in which their COTS packages were applied. If, as we expect, they indeed claim that their packages support distributed simulation and that they are aware of distributed simulation projects, we turn to the application of HLA in general and ask them whether they support the HLA standard and whether they have had the chance to apply it. This first part of the questionnaire, besides providing answers to the survey question, aims to give an overall picture about the integration approaches that are supported and applied by the COTS vendors and their customers. Accordingly, the first part of the questionnaire covers the following five issues:

1. *The application and applicability of distributed simulation in industry. The existence of distributed simulation projects carried out successfully either by vendors or by their customers.*
2. *The existence of successful simulation projects when the interoperating simulation models are created with the same, respectively different, COTS simulation packages.*
3. *Openness of the simulation package, support for integration with external applications.*

4. *Support for and willingness to support HLA compliant models by the vendors.*
5. *The existence of successful projects on integration through HLA.*

The second part of the questionnaire intends to find arguments to the second survey question, namely, *why* the HLA standard is hardly applied in industry. Therefore, this part is relevant for vendors who were involved in distributed simulation projects in which the simulation models were integrated through the HLA standard. In order to elicit these arguments we asked about the problems and difficulties that the vendors or their customers were confronted with when they applied HLA. This part of the questionnaire thus covers the following issue:

6. *Problems and difficulties with HLA.*

The third part of the questionnaire tries to disclose the vendors' vision regarding the future of distributed simulation and the HLA standard. The vendors' vision might give indications and support for answering the third survey question that intends to find alternatives for implementing distributed simulation solutions in industry. The issue that this part covers can be formulated as:

7. *Future visions regarding the support of HLA or other distributed simulation standards.*

Based on the above enumerated issues that need to be investigated, and including as an additional part collecting information regarding the participants, the questionnaire consists of the following eight sections:

- Section 1 collects information about the representatives of the organisation who filled out the questionnaire;
- Section 2 and 3 gather information about the support for interoperability with simulation models in the same, respectively, different simulation packages as evidenced by successfully carried out distributed simulation projects;
- Section 4 and 5 collect information regarding the support for interoperability with external applications. These sections are aimed to indicate the supported protocols/middleware for interoperability;
- Section 6 and 7 assemble information regarding the support for HLA as evidenced by successfully carried out HLA projects. Additionally, they deal with the difficulties and benefits of using HLA;
- Section 8 focuses on future plans regarding interoperability and the HLA standard.

The complete questionnaire is presented in Appendix A.

3.4.2 Selecting Participants

The participants of the questionnaires are COTS simulation package vendors. In Section 3.3 we have already specified the reason why we have identified the COTS simulation package vendors as an appropriate community that can provide enough evidence to strengthen our hypothesis. This section covers the way we have discovered and chosen these vendors.

In order to get an adequate assessment of the opinion of the vendors, we tried to contact as many vendors as possible. In order to obtain a large number of vendors, we have taken as a basis the survey conducted by James Swain (Swain 2001), referred to as the Swain list in the rest of the paper. The reason why we have chosen this list for the questionnaire survey is its openness, comprehensiveness and scientific character.

- *Openness.* The Swain list is an open scientific survey. It does not have any commercial intention and COTS vendors can add the description of their simulation products to the list at any time.
- *Comprehensiveness.* It is a long-term, continuous survey, which started in 1995 and is still running. The results have been published three times consecutively by Operation Research/Management Science Today (OR/MS Today) (Swain 1995), (Swain 1997), (Swain 2001) and since 1997 it is available on the web[11].
- *Scientific.* It is produced by INFORMS (Institute for Operations Research and the Management Sciences)

Of course there are COTS simulation packages which are not represented on the list. This can be caused by different reasons, for example, because vendors have no information about this survey or they do not want to participate in the survey. In spite of this possibility, we expect that this phenomenon does not influence the outcome of the survey, be it only because the Swain list contains the main and most well-known simulation package vendors.

The list contains 39 organisations[12], from which we have invited 35 organisations to participate. We have ignored 4 organisations, because the description of their activity does not fit in our target group, for example they do not provide simulation packages but only optimisation modules – see Appendix B. Furthermore, there are two additional COTS simulation package vendors, Wolverine Software and Lanner Group, which are not enrolled on the list but their products have been presented during the last years at the exhibition of the Winter Simulation Conferences, and therefore we consider them as well.

3.4.3 Methodology for Data Collection

For the 37 COTS simulation package vendors that we have contacted we made two options available for filling out the questionnaire: either through a web site or by filling out a template file and submitting it by FAX. We had the chance to collect data both through the

[11] It can be accessed via: http://www.lionhrtpub.com/orms/surveys/Simulation/Simulationmain.html

[12] This was the case on 4 October 2004

web site and fax; however the web site was more popular. The attractiveness of this form of data collection seems to increase (Best and Krueger 2004). The process for collecting the data took approximately two months.

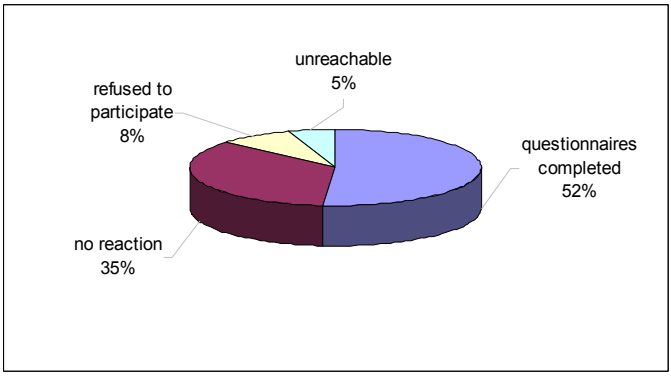


Figure 3.2 The Reaction of Contacted Organisations[13]

From the 37 organisations that we have contacted (Figure 3.2):

- 19 (52%) filled out and sent back the questionnaire;
- 13 (35%) did not react to the request;
- 3 (8%) refused to participate on the survey;
- 2 (5%) were unreachable.

The majority (52%) of the COTS simulation package vendors filled out the questionnaires. We collected data on these participating organisations in Table 3.1. This table summarizes the answers obtained from Section 1 of the questionnaire. In view of the evaluation of the survey it is relevant to observe that the main organisations are among these 52%.

3.4.4 Methodology for Data Analysis

The results of the questionnaire survey are quantitatively analyzed in the next section. For the quantitative analysis we have followed the series of steps presented in (Creswell 2003, pp. 159-161):

1. Report information about the number of participating COTS vendors who did and who did not return the survey. A table with numbers and percentages describing respondents and non-respondents is a useful tool to present this information (see Figure 3.2, Table 3.1 and Appendix B).
2. Discuss the method by which response bias will be determined. Bias means that if non respondents had responded, their responses would have substantially changed the overall results of the survey. In order to check for response bias, we did not apply any specific analysis (e.g., wave analysis, or respondent/non-respondent

[13] For detailed information see Appendix B.

analysis) because we have succeeded to obtain responses from the major COTS simulation vendors (Table 3.1 and Annex B).

3. Identify the tools for statistical analysis and a statistical computer program for testing the major questions from the questionnaire. For this reason we did not use very special tools.

Table 3.1 Participating COTS Simulation Vendors

Simulation Package	Name of Company	Representative	Function
AnyLogic	XJ Technologies	Alexei Filippov	CTO
Arena	Rockwell Software	David Sturrock	Product Manager
AutoMod	Brooks Automation	Ian McGregor	Simulation Manager
Crystal Ball	Decisioneering	Eric Wainwright	CTO
eM-Plant	Tecnomatix	Matthias Heinicke	Marketing
Enterprise Dynamics	Incontrol Enterprise Dynamics	Fred Jansma	Head of Development
Extend	Imagine That, Inc.	David Krahl	Vice President
FlexSim	FlexSim Software Products, Inc.	Bill Nordgren	President/CEO
GoldSim	GoldSim Technology Group, LLC	Ian Miller	President
GPSS World	Minuteman Software	Springer Cox	Not Specified
HighMAST (SAGE)	Highpoint Software System, LLC	Peter Bosch	Not Specified
Micro Saint Sharp	Micro Analysis & Design	Daniel W. Schunk	Industrial Engineer
ProModel	ProModel	Charles Harrell	CEO
ShowFlow	Webb Systems Limited	Steve Webb	Not Specified
Simul8	Simul8 Corporation	Mark Elder	CEO
SLX	Wolverine Software	Jim Henriksen	President
VisSim	Visual Solutions	Pete Darnell	President
WebGPSS/microGPSS	Flux Software Engineering	Ingolf Stahl	Not Specified
WITNESS	Lanner Group	Tony Waller	Product Manager

3.4.5 Presentation and Interpretation of the Collected Data

The collected data is presented on a question by question basis[14]. First, for each question we present a table which contains the question and the cumulative results. Then additional comments provided by the vendors and short evaluation comments by ourselves, if applicable, are given. If some questions are closely related to each other then we combine them: first we present for each question the cumulated answers and then we interpret and evaluate the data together.

Table 3.2 and Table 3.3 summarize the answers given regarding the part of the questionnaire dealing with interoperability in general, that is the questions formulated in Sections 2 and 3.

Table 3.2 Support for Interoperability within the Simulation Package

Question 1	**YES**	**NO**
Have, to your knowledge, any projects been carried out successfully that link two or more separate simulation models created in your package?	**13**	**6**

Table 3.3 Support for Interoperability with other Simulation Packages

Question 2	**YES**	**NO**
Have, to your knowledge, any projects been carried out successfully that link two or more separate simulation models created in your package with models created in other simulation packages?	**10**	**9**

The answers given reflect that COTS simulation vendors recognize the success of their package in different industrial oriented distributed simulation projects both when the same, homogeneous, simulation packages (Table 3.2) and when different, heterogeneous, simulation packages (Table 3.3) are integrated. Although, we claim in our hypothesis that distributed simulation, specifically HLA, is hardly applied in industry, in both cases more than 50% of the COTS vendors or their customers claim success with distributed simulation projects. Consequently, although we still need to analyze the support for HLA, it seems that the results contradict our hypothesis. The approaches that the vendors or their customers apply for distributed solutions differ, but generally, as some of them specified, they employ more or less basic homespun solutions especially based on WinSock.

Sections 4 and 5 of the questionnaire are aimed to further elaborate on support for external applications and the applied solutions.

[14] For the whole questionnaire see Appendix A

Table 3.4 Support for Interoperability with External Applications

Question 3	**YES**	**NO**
Does your simulation package support interoperability with external applications (e.g., data bases, spreadsheets, optimisation software, etc.)?	**18**	**1**

Based on Table 3.4 we can conclude that almost all vendors opened their packages for external applications. The COTS simulation packages support interoperability with external applications, such as:

- Databases (SQL/ODBC links, Access, Oracle, SQL Server);
- Spreadsheets (Excel);
- Text Editors (Word);
- Optimisation tools (OptQuest, OPTIMIZ, ISSOP, GRG2, WITNESS Optimizer);
- Mathematical and Statistical Software (ExpertFIT, Stat::Fit, MATLAB, Mathcad);
- External devices;
- Modelling tools (Visio, AutoCad, ProEngineer);
- ERP Systems (SAP).

Table 3.5 Protocols/Middleware Used for Interoperability

Question 4	
If your package supports interoperability, which protocol(s)/middleware do you use to realize it?	**N = 18**
WinSock	**8**
CORBA	**3**
COM	**13**
ALSP Interface	**0**
DIS Interface	**1**
HLA Interface	**7**
Other Specified Interfaces	**9**

In order to achieve interoperability with external applications, the COTS packages use different techniques/interfaces. In the questionnaire we specified a list of protocols/middleware which the vendors might have used for interoperability purposes. Table 3.5 specifies the answers of the vendors. Besides our proposed solutions, participants pointed at other interfaces that their packages support. Those are:

- RS232;
- DLL interface;
- C interface;
- VBA interface;
- ActiveX;

- Dynamic Data Exchange (DDE);
- OPC;
- XML;
- .NET.

Summarizing, we can state that WinSock, COM and, surprisingly again, the HLA interface are the most supported protocols from the six ones that we mentioned explicitly. As we can see, seven vendors out of nineteen claims that their package supports HLA, a result that again seems to contradict our hypothesis that HLA is hardly applied in industry.

Next we explicitly asked the participants whether they made use of HLA. The results of Section 6 that focuses on this issue are presented in Table 3.6.

Table 3.6 Support for HLA

Question 5	**YES**	**NO**
Have, to your knowledge, any projects been carried out successfully in which simulation models created in your package are integrated using the HLA standard?	**7**	**12**

Some participants, seven out of nineteen, claimed that they carried out successful projects by applying HLA. This result again seems to contradict our hypothesis.

Section 7 of the questionnaire is related to Section 6 and asks vendors who applied HLA about their experience and difficulties with applying this package.

> ***Question6.*** *How much effort did the developer, to the best of your knowledge, take to make the simulation model HLA compliant (specify as percentage of the overall time of the simulation project)?*

The answer to this question varied from 2% till 75%, which warns us that the question is badly stated in the sense that it does not cover the initial effort made on creating the simulation model apart from HLA compliancy. The transformation of a very expensive monolithic simulation model into an HLA compliant one might cost 2% of the original amount, whereas transforming a simple, relative cheap model into an HLA compliant one might amount to 75% of the original cost. However, the 2% of the costly model might be far more expensive than the 75% of the low-priced model.

The COTS vendors who specified that significant effort was needed to make a simulation model HLA compliant complain about:

- The alignment of shared data (the specification of the FOM);
- Translation of the COTS concepts into HLA terms;
- The exchange of heterogeneous data;
- The verification and debugging of the distributed models.

3.4 The Questionnaire for the COTS Simulation Vendors

We now present the results of Section 7 dealing with reuse.

Table 3.7 Reuse of HLA Compliant Simulation Models

Question 7	**YES**	**NO**
Has your organisation ever reused HLA compliant simulation models created in your package in another project?	**1**	**18**

Although reusability is one of the most important economical arguments for applying HLA, or distributed simulation at all, apart from one, none of the participants reports on reuse of HLA compliant simulation models (Table 3.7). This is a result that triggers some thinking. First of all, reusability is an important issue for doing distributed simulation. Secondly, there are seven vendors that support HLA in their package and all of them had successful HLA-oriented simulation projects. Considering the fact that they provide support for HLA, how is it possible that only one of them reused existing HLA compliant models? Why did not the others benefit from the reusability feature? Can it be that reusability is not an important issue at all, or is it the case that it cannot easily be solved by their HLA support?

The last section of the questionnaire focuses on the future intentions of the vendors regarding the use of HLA in particular, and distributed simulation in general. Table 3.8 and Table 3.9 give an overview of the participants' plans in this respect.

Table 3.8 Future Plans for Supporting the HLA Standard

Question 8	**YES**	**NO**	**IT SUPPORTS**
Does your company make efforts to support HLA as an additional feature in your package?	**6**	**8**	**5**

Although many COTS vendors provided a negative answer to the question on efforts made to support HLA by their simulation packages, as Table 3.8 depicts, taking into account the affirmative answers and the number of companies who already claim to support HLA, there seems to be a tendency towards supporting HLA.

Table 3.9 Future Plans for Supporting Other Standards than HLA

Question 9	**YES**	**NO**
Does your company make efforts to support other standards than HLA for distributed simulation?	**13**	**6**

Most of the COTS vendors explicitly want to support distributed solutions other than HLA in the future (Table 3.9). When asked to name specific protocols they intended to support, most of them indicated to stay with low level technical protocols, middleware, or interfaces that they intend to provide in their future versions, such as WinSock, COM, DCOM, .NET. Six out of eight vendors, who do not want to support HLA, do not want to support other standards as well. These vendors are probably not oriented at all towards distributed simulation.

Evaluating the questionnaire we were surprised to see a picture featuring quite HLA-minded COTS simulation package vendors. From our preliminary observation that the HLA standard is rarely applied in industry, we would expect that almost none of the COTS packages supports HLA, whereas analyzing the results of the questionnaire it turns out that there are quite some successful HLA related projects and furthermore it seems that there is a significant support for the HLA standard.

On the other hand, analyzing the results in more detail there seems to be some discrepancy in the answers. Let us take a look at the different variations of answers to the questions 4, 5 and 8.

Table 3.10 The Variations of Collected Answers for Questions 4, 5 and 8

Question 4	NO	YES	YES	NO	YES	NO	YES	NO
Question 5	NO	YES	YES	NO	YES	YES	NO	NO
Question 8	NO	YES	It Supports	YES	NO	NO	It Supports	It Supports

Table 3.10 depicts all combinations of the vendor's answers regarding question 4, 5 and 8 that we obtained. We repeat that, Question 4 and 8 mainly refer to support for the HLA standard, whereas Question 5 refers to successful projects applying HLA. As we can see we collected peculiar combinations that one could imagine. On the one hand, there are vendors who claim that they do not provide an HLA interface and they do not want to support it in a future version, whereas they or their customers have carried out HLA related successful projects. On the other hand there are vendors who state that they provide an HLA interface and they support it in their package as additional feature, however they never carried out successful HLA related simulation projects. Furthermore, there are vendors who claim that they provide an HLA interface and they also applied it in successful HLA related projects, however they do not want to support it as an additional feature in their packages. Further, there are vendors who affirmed that they do not provide an HLA interface and they never applied in any successful project, however they support it as an additional feature in their package.

Based upon these hard to interpret combinations of answers, the only conclusion that we can draw is that the vendors interpret the notion "supporting HLA" in different ways. Studying HLA, for example, or attending presentations regarding HLA, or providing low level protocols, such as WinSock, might be considered by them as an effort to support HLA as an additional feature, which to a certain extent might be true. However, this is far

from what we would call de facto support and service for an efficient HLA interface for simulation practitioners.

In order to find out the way vendors support or intend to support HLA in their packages we have decided, fully in line with the Delphi approach, to conduct a second iteration of questionnaires in which we have focused more sharply on the effort spent on supporting HLA. We have confronted the five vendors who answered "It supports" to question 8 with a list of options and we presented another set of options to the six vendors who answered "yes" to the same question. The set of options for those vendors who answered "It supports" to question 8 is given in Table 3.11.

Table 3.11 Confirmation Regarding the Support of HLA Standard

***Question**: It turned out that there is one issue that I cannot evaluate properly. The problem is related to the question "Does your company make efforts to support HLA as an additional feature in your package?" for which you have answered "It supports already". I would like to ask you to specify the type of the effort you have made. Could you please choose one of the possible options from the list below:*	
Options	N=5
1. We do not provide specific HLA middleware, but we support HLA in the sense that we provide protocols/middleware/interfaces (e.g., WinSock, COM, CORBA, C++ interface, etc.) that enables one to connect to external software applications, that in their turn could connect to the HLA RTI.	2
2. We have already designed and developed specific HLA middleware (adaptor). The simulation models created in our package can connect to the RTI through this middleware that is not part of our package, but a separate piece of software code. It connects the models developed in our package to the HLA RTI. In order to create this middleware we did NOT modify the structure (e.g., engine) of our simulation package at all.	1
3. We provide HLA middleware as an external application in the same way as described in 2, but in order to solve it we have SLIGTHLY modified our simulation package.	-
4. The whole HLA concept is INCLUDED within our package. Simulation practitioners design and develop HLA compliant simulation models within our package without being involved in low level HLA details (e.g., using internal HLA components/libraries).	-
5. Other. Please state.	-

According to Table 3.11 two COTS vendors out of five have chosen option 1, one vendor has chosen option 2, and the rest two vendors did not make a choice. Actually, they did not answer to the second round of questions. Summarizing, from the choice of the vendors who responded to this question, we perceive that HLA support is mainly provided by low level protocols through which one can connect to the HLA-RTI.

The set of options for the six vendors who answered "yes" to Question 8 are presented in Table 3.12.

Table 3.12 Confirmation Regarding the Intention to Support of HLA Standard

***Question**: It turned out that there is one issue that I cannot evaluate properly. The problem is related to the question "Does your company make efforts to support HLA as an additional feature in your package?" for which you have answered "Yes". I would like to ask you to specify the type of the effort you intend to make in the near future. Could you please choose one of the possible options from the list below:*	
Options	N=6
1. Our organisation makes efforts to support HLA in the sense that we are studying books, articles about HLA or we attended some HLA tutorials in order to discover what the benefits are of using this standard.	1
2. Besides studying the HLA we are working on HLA middleware (adaptor), which is aimed to support the interoperability of simulation models created in our package with the HLA RTI. We are NOT going to modify our simulation package; we just intend to create HLA middleware as a separate application that supports the connection of our model through RTI.	-
3. We are working on HLA middleware as a separate application, and our organisation is open to make some SLIGHT modifications to the structure of our simulation package (e.g., simulation engine) if it is necessary.	2
4. We are working on a version that will INCLUDE the whole HLA concept in our simulation package even if we have to modify the whole structure of the package. We aim to provide to our customers a simulation package that supports the design and development of HLA compliant models. The low level HLA details will be hidden from the developers in this way.	-
5. Other efforts. Please state.	1

From the responses we have got, one COTS vendor considers option 1, two vendors consider option 3 and one vendor considers option 5 with the comment: "We are implementing HLA support that is INCORPORATED in our product. It means SLIGHT modifications to our product". Two vendors did not choose anything; they did not react to our request.

As we can see, the efforts the COTS simulation package vendors intend to make for supporting HLA differs on a wide scale. There is a vendor that just intends to study this topic, while another one would like to take the chance and "incorporate" it in their package.

Concluding, the results of our additional questionnaire indicate that, although there seems to be active interest in HLA by about half of the COTS vendors, this interest is rather tentative and aiming at low level solutions. This conclusion is strengthened by two more

results from the main questionnaire, namely complaints on HLA and the type of projects mentioned.

Those organisations that refused to consider HLA as an additional feature in their future package complain that:

- The cost is too high to incorporate it as a supported feature considering the benefits they would gain from it;
- It is very military specific and so is weighed down by support for features not required in many cases;
- The implementation of the HLA standard has not the expected performance when applied to accomplish interoperation between simulation models;
- The system management of running the model is complex and not easy to support in a general architecture;
- There are representation problems of the very different attributes to be exchanged.

Miller from GoldSim explicitly states that "I recall reviewing the HLA approach, and being impressed with its scope and ambition, but sceptical about its practicability."

Besides the support for the HLA standard within the COTS simulation packages we analyzed in detail the specified distributed simulation projects with the purpose to extract those which applied HLA for interoperability. Table 3.13 presents the distributed simulation projects referred by the COTS vendors in the questionnaire. These projects were mentioned in the answer to Question 1, 2 and 5. Although seven vendors have chosen "yes" in Question 5, one of them did not explicitly specify or name the project. Since we could not identify whether it was an industrial or defence oriented project, we did not include it in the table below.

Table 3.13 Distributed Simulation Projects Specified

Distributed Simulation Projects Specified	**N=23**
1. **Industrial** oriented distributed simulation projects that integrate **homogeneous** COTS simulation packages **through homespun architectures**	**12**
2. **Industrial** oriented distributed simulation projects that integrate **heterogeneous** COTS simulation packages **through homespun architectures**	**5**
3. **Industrial** oriented distributed simulation projects that integrate **homogeneous** or **heterogeneous** COTS simulation packages **through HLA**	**2**
4. **Defence** oriented distributed simulation projects that integrate **homogeneous** COTS simulation packages **through HLA**	**3**
5. **Defence** oriented distributed simulation projects that integrate **heterogeneous** COTS simulation packages **through HLA**	**1**

Before concluding anything from these numbers we have to state that we do not know to what extent the projects mentioned give full insight, because we did not ask the vendors to

mention all projects they are aware of. Therefore, this information can be considered only as an indication.

Analyzing the provided information we consider three dimensions:

- Industrial or Defence oriented projects;
- Integration of Homogeneous or Heterogeneous Simulation Packages;
- Integration through HLA or homespun architectures.

Table 3.13 suggests the following trends:

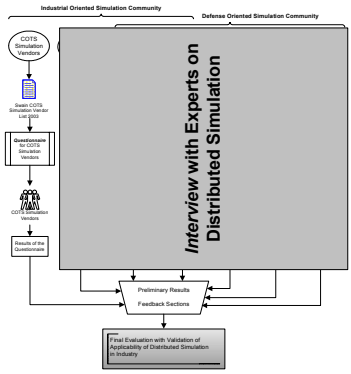
1. HLA is rarely applied in industry when integrating COTS simulation packages.
2. When HLA is applied to integrate COTS simulation models, this occurs mostly in defence oriented projects.
3. COTS simulation packages are rarely applied for distributed defence oriented simulation models.
4. The COTS vendors and their customers prefer simple homespun solutions over the HLA standard.

The questionnaire aimed to validate our hypothesis that the HLA standard is rarely applied in industry. Looking superficially to the results it seemed that our hypothesis was contradicted. Deeper analysis unveiled several inconsistencies which seemed to favour our hypothesis again. Our conclusion is that, although the questionnaire could not support us to completely validate the hypothesis, the analysis we have carried out and a second round of questions strengthens our hypothesis to some extent. The hypothesis as it is, still stands and we intend to further validate it by means of the interview survey.

Concerning the second survey question, there are several arguments we collected from the questionnaire, why the HLA standard is hardly applied in industry. The heavy structure of the HLA RTI, the performance, the alignment of shared data, the complex runtime management, the verification and debugging, and translation of COTS concepts into HLA terms led to an unfair cost benefit ratio.

Regarding the third survey question we conclude that some of the COTS simulation package vendors are looking into the future to support HLA in their packages; however, currently there does not seem to be a big drive in this direction. Furthermore, we have observed that while a couple of organisations intends to accommodate distributed simulation by creating “homespun” architectures based on low level technical solutions, other organisations do not have any intention at all to support connection with other packages.

3.5 The Interview with Experts on Distributed Simulation



In accordance with the Delphi approach we use the results from the questionnaire as a starting point for the next round, the interview survey with experts. This section focuses on the description of the interview. Like the section on the questionnaire survey, it includes five parts. In the first part we present the design of the interview survey that is organized around nine topics. Then, in the second part, we describe the way we selected the participants and introduce the experts, who are classified into five groups as depicted in Figure 3.1. Next, we describe the method that we have used for collecting and analyzing the data. Finally, the data collected from the interview is presented based on the survey design. For each group of people we give a separate report and each report is organized along the nine topics. The results of interview are interpreted and evaluated in the next chapter.

3.5.1 Survey Design

As concluded from the previous section, the questionnaire survey did not provide sufficient data to validate our hypothesis. However, it still gives several hints that can help us to design an interview survey that, besides testing the hypotheses, aims to provide answers to our second and third survey questions. Additionally, the interview survey should also clarify or explain the contradictory results found during the analysis of the questionnaire. In order to be able to answer the three survey questions of this section, and to provide a valid interpretation and evaluation of the collected data, several additional pieces of information should be collected, and further the survey questions should be refined, so that the participants understand and answer the right questions. In accordance to this reasoning, we design the interview and collect data along nine main topics:

1. *The experience and knowledge of the interviewed experts* in the field of distributed simulation, the HLA standard, COTS simulation packages, and general programming languages. This information is especially important in order to determine and validate the weight of the participants' opinion.
2. *Differences between industrial and defence simulation.* The questions around this topic are introduced by stating the observation, that "In my opinion there is a difference between the industrial and defence-oriented simulation communities. Defence simulation communities mainly use general purpose languages, such as Java, C++, FORTRAN, and so on, for creating simulation models, while the industrial simulation communities use COTS simulation packages". We have asked the experts about their opinion and observations regarding this statement. In this way we try to validate the assumption that industry mostly applies COTS simulation packages. Furthermore, we try to uncover other differences between these two communities that

they might have observed. This topic primarily aims to find reasons why distributed simulation gathered ground in defence and hardly appears in industry.

3. *Reusability of existing simulation models.* As in our view reusability of already existing models seems to be the one of the most beneficial motivations for applying distributed simulation, we enquire the participants of the interview about their experience concerning reusability. We especially focus on reusability in the context of HLA, namely on reusability of HLA compliant simulation models. Difficulties and efforts invested for reusing models compared with reimplementing the whole model are investigated here. The aim of this topic is to help us to gather motivation for applying distributed simulation, and further to suggest solutions that can filter out the encountered difficulties.
4. *Information hiding within the simulation model.* Besides reusability information hiding seems to be another motivation for distributed simulation. We try to elicit information on whether this is indeed the case.
5. *Difficulties and benefits of applying the HLA standard.* From the questionnaire and also from literature opinions can be heard about the difficulties of using the HLA standard. We try to discover whether this is a significant reason why HLA in particular and distributed simulation in general does not gain ground in industry.
6. *Integrating simulation models developed in COTS simulation packages through the HLA standard.* This topic is introduced by stating our hypothesis that "The HLA standard is rarely applied for integrating simulation models designed and developed in COTS simulation packages". We invited the experts to give their opinion on this hypothesis and furthermore provide some arguments concerning their opinion. Similarly to the previous section but focusing more on COTS simulation packages, this section intends to investigate whether this is another significant reason why the HLA standard is not considered in the industrial domain.
7. *Benefits and disadvantages of integrating simulation models designed and developed in COTS simulation packages.* This issue is only discussed with the COTS simulation package vendors, who provide tools to the simulation practitioners. As we aim to provide a solution for the integration of different COTS models we are curious about their vision regarding the integration of different COTS simulation packages.
8. *Support of commercial HLA tools for the industrial domain.* This topic is discussed only with the HLA vendors, who provide commercially available distributed simulation architectures for integrating simulation models mostly for the defence community. We intended to investigate their role in industry and their support for industrial domain.
9. *Future visions on 'idealized'[15] distributed simulation architectures for the industry.* This issue tries to investigate the vision of experts on distributed simulation from different groups. We pose the question: "How would you imagine an 'idealized' distributed simulation architecture for the industrial domain where the COTS

[15] With 'idealized' we refer to an architecture that experts would wish or imagine even if it is impossible to realize this today

simulation packages are frequently applied?". From the answers we aim to compose a list of requirements for future solutions for distributed simulation architectures.

Each issue presented above, except for the first one, can be associated to one of the survey questions discussed in Section 3.2. The sixth issue together with our statement in the second one aims to validate our hypothesis. The line of reasoning is that industry uses mainly COTS packages, and that COTS and HLA rarely go together. These two statements would then imply our hypothesis. The answer for the second survey question is gathered by dealing with the 7 issues numbered from 2 till 8. The 9th issue relates to the third survey question by explicitly asking the experts what kind of solution they imagine for stimulating distributed simulation in industry. Additionally, given that this is an open-ended interview, the answers given to some of the questions can be relevant for more than one survey questions. A comprehensive description of the sections and the questions posed in the interview is provided in Appendix C.

3.5.2 Selecting Participants

As we mentioned before through the interviews we aim to collect data from experts. The interview we conducted involves unstructured and generally open-ended questions intending to elicit views and opinions of the participants (see Appendix C).

"The idea behind qualitative research is to purposefully select participants or sites (or documents or visual material) that will best help the researcher understand the problem and the research question. This does not necessarily suggest random sampling or selection of a large number of participants and sites, as typically found in quantitative research" (Creswell 2003, pg. 185). Consequently, in contrast to the questionnaires, where we applied quantitative research and we intended to address a large number of vendors, for the interview we try to select *purposefully* those experts who, we believe, can support our research with their experience, view and opinions. This leads into questions around how to select experts, and finding criteria on who can be considered to be an expert.

An expert is a person whose knowledge in a specific domain, in our case distributed simulation, is obtained gradually through a period of learning and experience (Turban 1995). A person's experience can be theoretical, this deals with experience obtained from scientific research on distributed simulation, or can be practical, this deals with experience obtained by integrating simulation models, or can be a combination of both (Schreiber*, et al.* 2000). An important consideration during the selection of experts is whether to use a heterogeneous group of experts (e.g., both scientists and practitioners, or industrial-oriented simulation practitioners and defence-oriented practitioners) or a homogenous group (e.g., only scientists). We choose to select a heterogeneous group of experts, the advantage of which over a homogenous group being that different groups can sense issues differently, they might consider the problem from a different angle (Schreiber*, et al.* 2000).

Although is it impossible to provide a definite list of criteria for expert selection, based on the above observations we lay out the main decisive factors according to which we selected the experts:

- The person's knowledge and experience with distributed simulation;

- The circumstances in which the person gained his/her experience: e.g., theoretical or practical circumstances;
- The selected group should be heterogeneous including people both from the industry and the defence simulation community (especially because HLA was designed and developed by defence).

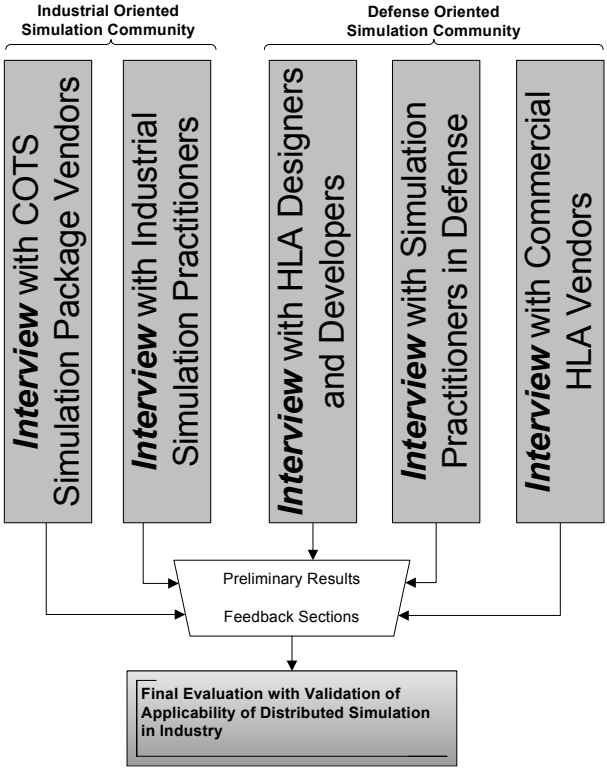


Figure 3.3 The Five Groups of Experts

As depicted in Figure 3.1, we consider experts both from industry and from the defence community. Initially we consider a subset of the COTS vendors, we choose them based on the result of the questionnaire, and industrial oriented simulation practitioners both from the scientific area and from practice. As a starting point for selecting these experts we considered the HLA-CSPIF [16] group, which aims to stimulate the applicability of distributed simulation in the industrial domain. We also make use of some pointers to interesting people as provided in the questionnaire by the COTS vendors. For the defence community, we identify three groups: experts who designed and developed HLA, simulation practitioners who apply HLA in defence oriented simulation projects (for these

[16] The HLA Commercial-Off-The-Shelf Simulation Package Interoperability Forum - http://www.cspif.com

two groups we considered DMSO[17], MITRE[18] TNO-FEL[19] and SISO[20] as starting points), and HLA commercial vendors which are chosen from the DMSO vendor list published on the DMSO website[21]. In this way we try to avoid bias against HLA, given that we mainly address practitioners who have applied HLA.

The five target groups identified above that we intended to interview and the methodological steps applied for the interview survey are illustrated by Figure 3.3 that is in fact a snapshot related to Figure 3.1 where the methodology for the whole survey is summarized. Table 3.14 provides a list of the experts who have been selected and interviewed. The table furthermore presents the groups from which the experts were selected and their expertise/activity indicating why they were selected.

3.5.3 Methodology for Data Collection

For conducting the research and for answering the three questions posed in Section 3.2 the main collection procedure that we choose is *an unstructured open ended interview with experts, which will be recorded and then transcribed*. Besides, as support, we employ our observations and the literature study.

In order to interview experts there are different options, such as a face-to-face interview, a phone interview or a group interview in which researcher interviews participants in a group (Creswell 2003). In our case, we conducted a phone interview with all participants, except for Richard Fujimoto who validated the first version of the findings by means of a face to face interview.

The average time for an interview was 58 minutes. Regarding the evaluation of the interview, as a last question, we asked the experts' opinion about it. They were all quite positive both concerning the topic and the content.

3.5.4 Methodology for Data Analysis

Data analysis in qualitative research involves preparing the data for analysis, conducting different analyses, moving deeper and deeper into understanding the data, representing the data, and making a global interpretation of the data (Creswell 2003), (Seale*, et al.* 2004). Several generic processes are described in literature to convey the overall activities of qualitative data analysis, in this research we follow the generic steps presented in (Creswell 2003, pg. 191-195):

- *Organize and prepare the data for analysis.* In our case this involves transcribing interviews and sorting, arranging them.

---

[17] Defence Modelling and Simulation Office - https://www.dmso.mil/

[18] Research and Development Center for US Department of Defence - http://www.mitre.org/

[19] TNO Physics and Electronics Laboratory, Command & Control and Simulation Division department - http://www.tno.nl/instit/fel/

[20] Simulation Interoperability Standards Organizations - http://www.sisostds.org/

[21] https://www.dmso.mil/public/transition/hla/vendorlist

- *Read through all data.* A first step is to obtain a *general idea* of the information and to reflect on its overall meaning. What general ideas are the experts expressing? What is the tone of the ideas? What is the general impression of the overall depth, credibility, and use of the information?
- *Begin detailed analysis with a coding process.* Coding is the process of organizing the material into different topics, subtopics, and so on. For this coding process there are different approaches, one of them being discussed in (Creswell 2003). First, the researcher gets an idea of the overall picture by reading all transcriptions carefully. Then he/she picks one of the documents, maybe the most interesting one, reads it carefully and makes a list of major topics. If there are similar topics then he/she clusters them. After that he/she takes this list and goes back to the whole data, trying to apply it to the data which might lead to new topics. The coding procedure will be dealt with in Section 3.6 and Chapter 4.
- *Use the coding process to generate a detailed description of topics and subtopics.* Again will be treated in Section 3.6 and Chapter 4.
- *The final step in data analysis involves making an interpretation or meaning of the data.* Based on the results presented in the previous point, we interpret the data and we construct an evaluation. The evaluation aims to explicitly answer the initial three questions presented in Section 3.2, based on the interview results, questionnaire results, own observations and literature review. Finally, the whole methodology and the evaluation are validated by experts in the field. The validation involves two steps. The first step is a preliminary evaluation which is conducted through a face to face presentation and interview with Richard Fujimoto. Then the second evaluation phase involves all experts who were involved in this survey. A report is sent to these experts and their feedback is considered to evaluate the results of this survey.

Table 3.14 Participating Experts

Nr	Experts	Group[22]		Expertise/Activity	Current Organisation
1.	Matthias Heinicke	I	CV	Marketing Manager at Tecnomatix	Tecnomatix, Germany
2.	Ian McGregor	I	CV	Simulation Manager at Brooks Automation. Involved in various distributed simulation projects within Brooks Automation.	Brooks Automation, USA
3.	Stijn-Pieter van Houten	I	SP	Involved in a Distributed Supply Chain Simulation project integrating Arena simulation models through HLA	Delft University, The Netherlands
4.	Peter Lendermann,	I	SP	Involved in parallel and distributed simulation and advanced methods for supply chain planning and production scheduling	SIMTech, Singapore
5.	Low Malcolm	I	SP	Involved in integrating semiconductor supply chain simulations through HLA	SIMTech, Singapore
6.	Markus Rabe	I	SP	Project Manager of the IMS Mission Project Modelling and Simulation Environments for Designing Globally Distributed Enterprises	Fraunhofer Institute, Germany
7.	Steffen Straßburger	I	SP	Involved in distributed simulation projects at DaimlerChrysler aimed at designing digital factories	Fraunhofer Institute, Germany
8.	Simon Taylor	I	SP	Coordinator of the HLA-CSPIF forum	Brunel University, UK
9.	Mike Ryde	I	SP	Researching the possibilities to integrate COTS simulation packages	Brunel University, UK
10.	Edwin Valentin	I	SP	Researches possible ways of applying building blocks with COTS simulation packages	System Navigators, The Netherlands
11.	Katherine L. Morse	D	HD	Involved in the technical support team for HLA in which she was responsible for the design of the HLA data distribution management services.	SAIC, USA
12.	Richard Weatherly	D	HD	Led the HLA infrastructure development and verification team.	MITRE, USA
13.	Len Granowetter	D	HV	Director of MÄK product development	MÄK Technologies, USA
14.	Björn Möller	D	HV	Responsible for HLA products development at Pitch AB, Sweden. Co-founder and vice president of Pitch AB	PITCH AB, Sweden
15.	Wim Huiskamp	D	SP	Involved in defence oriented distributed simulation projects	TNO-FEL, The Netherlands
16.	Rick Severinghaus	D	SP	Involved in defence oriented distributed simulation projects. Member of SISO Executive Committee	Dynamic Animation Systems, Inc, USA
17.	Richard Fujimoto	D/I	SP	Expert on Distributed Simulation Theory.	Georgia Institute of Technology, USA

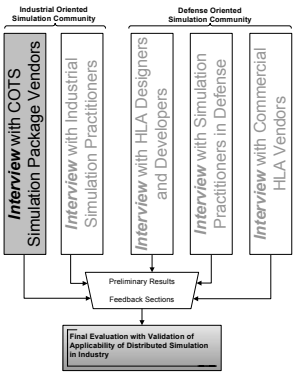
I=Industry; D=Defence; CV=Cots Vendor; SP=Simulation Practitioner; HD=HLA Developer; HV=HLA Vendor

[22] The role in which the experts participated in the survey. Experts might be either consultants, or developers, or researchers, regardless of the group in which they are classified. The first column (I and D) refers to the nature of the projects they were involved at the time we interviewed them.

3.5.5 Presentation of the Collected Data

The results are presented for each selected group separately (see Figure 3.3) ordered by the issues presented in Section 3.5.1.

3.5.5.1 COTS Simulation Package Vendors



We have contacted two COTS vendors for the scope of this interview. One of the vendors is Brooks Automation from the US that provides the simulation package AutoMod and the other one is the European company Tecnomatix that sells the eM-Plant simulation package. Brooks Automation is represented by Ian McGregor and Tecnomatix by Matthias Heinicke. Their functions in the company can be found in Table 3.14.

Experience in distributed simulation projects

Both representatives have been directly or indirectly involved in industrial distributed simulation projects.

They have applied distributed simulation in different ways. Brooks Automation has two different views here. According to McGregor "The first view would be AutoMod models talking to each other through sockets and those models would be mostly large models, like a model of postal operations. This is what we used when we had several large contracts with the US postal service. We produced pretty much template models for quite a few different distribution centers, and because those models were so big it became preferable to build them separately and then join different parts of them using a socket based technology that we put into an optional part of AutoMod called MCM (Model Communications Module). The second kind of integration that we have done, which is a quite rapidly growing part of our business is communication with control systems via OPC[23]. What we are doing is using the 3D model of a for instance material handling system to replace the real thing in testing the control system". As we can see in the first context they used distributed simulation because of the benefit of collaborative design and development, while the second case is more related to emulation, connecting to real equipment.

Tecnomatix considers three main reasons for using distributed simulation, the first one being integrating already existing simulation models, using socket communication or HLA. Secondly, they have their own integrated distributed simulation architecture that "distributes a simulation from one computer over more computers, running the same simulation on other computers to use their CPU time to get faster result". This is more or less parallel simulation. Finally, they connect eM-Plant models to real equipment and controllers through control logic software (PLCs).

[23] OLE for Process Control – A set of seven open standards for connectivity and interoperability of industrial automation and enterprise systems (http://computing-dictionary.thefreedictionary.com).

Benefits from integrating different COTS simulation models

Regarding the benefits vendors could gain from integrating simulation models in their own package with models in other packages, the experts showed a similar attitude. McGregor does not see any benefits because "the amount of asset that we would have to go to, to map our objects to their objects would probably outweigh any financial benefit that we would get. (…) Why would we encourage our users to integrate our products with our competitor's products? That is the basic question you have to ask. What is the economic reason for working with a competitor? So simple as that". He agrees that "If we could find a way that it would help increase our sales as well as theirs, perhaps, then you might have a business case for it, but otherwise there is no financial motivation to do that".

Heinicke sees some benefits of integration in the case when "you have specialized tools, with specialized concepts, and you absolutely do not have a chance to include this in eM-Plant." He illustrates the benefits by referring to a project, where a chemical reaction is simulated by another tool and linked to an eM-Plant factory simulation model. Heinicke considers distributed simulation beneficial especially for those companies that are using two or more simulation packages. McGregor mentions supply chain modelling as an area where distributed simulation might really benefit.

According to Heinicke it does not make sense to use integration of models written in the same COTS simulation package: "We see only more work. With discrete-event simulation we do not see the benefit for the customer. If you are running discrete-event simulation programs it is easier to run it on one platform".

Both vendors see the relevance of integration of simulation models with real equipment and controllers, because it enables one to test real equipment in a simulated factory or simulated equipment in a real factory.

McGregor mentions that the advantages of applying distributed simulation are not clear in general, and points to the fact that currently most of the simulation models are "throw away" models. Their experience is that most of the users do not maintain a simulation model across the life of an automated system.

On the other hand, when discussing distributed simulation and the potential application area where simulation integration might be of benefit McGregor argues: "Well, certainly it is a quite large area, but it remains as yet unexploited because of status of the market. One big opportunity could be integrating simulation into high level packages, such as ERP and supply chain management systems".

Difference between industrial and defence simulation

The experts agree that there is a difference between the industrial and defence oriented simulation communities in the sense that defence simulation communities mainly use general purpose languages for creating simulation models, while the industrial simulation communities use COTS simulation packages. According to McGregor the reason is that "in general industrial users are a lot less interested in technology and a lot more interested in getting a solution out as fast as possible. Whereas, in the defence they take the time, they have the budget (if not they make an argument for the budget), and they do it from scratch. By this they create a more customized solution and they have an absolute control over the system. (…) In industry the person who pays for the simulation product wants

results, he is not interested at all in the beauty of the product, in the elegance of the solution; he just wants the right results as fast and as cheaply as possible. I know quite a few people who are totally enchanted by the art of simulation and they love getting into the code, and most of them are demanding customers, because they want to see functions that they can imagine being written into our product. But at the end of the day their bosses want to see them spending less time making simulation models and more time generating revenue in other ways for the company". Thus, according to McGregor the difference between the industry and defence is a "different perspective on return and investment". He supports his observation by stating that in the military it is the question of being able to define a budget and show the advantages of doing simulation and therefore the return on investment is not a top priority.

Reusability

These representatives of the vendors have no experience with reusability of HLA compliant COTS simulation models, they have, however experience with reusability of already existing models, be it not necessarily in the context of distributed simulation. As McGregor says "reusability is something that is happening more". Most of the reusability in industry occurs at a component level. For example "the customers of Tecnomatix often reuse building blocks", while "the customers of Brooks Automation no longer start with a blank model, many of them start with a standard model, that contains a lot of their favourite algorithms." Reusability of a complete model, however, seldom occurs. The reason behind this is that, as McGregor specifies, "they are too project specific to be reused". However, sometimes this might occur, for example at DaimlerChrysler where already existing eM-Plant simulation models were reused for a virtual factory simulation, or in the semiconductor industry where simulation models created with ToolSim or AutoSched AP from Brooks Automation were used anew.

Information hiding

As Heinicke stated eM-Plant allows the user to hide parts of its models by using a key. This issue will be one of the future developments in AutoMod as well. They also agree that distributed simulation can provide a solution for hiding information within a simulation model. This is important especially, as mentioned by McGregor, "if somebody wanted us to collaborate even with another consultancy firm, using AutoMod it could be appropriate for both of us to define an interface and then not see the code that the other company has produced".

Difficulties and benefits of applying the HLA standard

None of the respondents are an expert on the HLA standard but to the question "What kind of difficulties, problems do you see in applying the HLA standard?" they have provided some valuable responses based on their research and observations.

Heinicke states that "I see only difficulties. There are no experienced users today. HLA is not that standard as we thought it should be. The complete HLA structure is a large structure, so normally simulation tools only need small part of it". Tecnomatix has done a market research on the HLA standard and their conclusion was: "There is no money in it. There is no organisation who wants to pay for it. We do not see a market where today the small or middle size companies would pay for it". Similarly, McGregor states that "The

benefit is not obvious to us. The difficulties are always going to be the mapping to different conceptual objects between systems". He illustrates his point with an example from a project where they linked together AutoMod and Witness models. The real problem was the fundamental difference in the way AutoMod and Witness viewed the world. Furthermore McGregor states that "HLA seems like an interesting idea but probably more attractive if you are in a large collaborative environment, possibly such as supply chain, possibly such as military. But even then it all comes back to each company saying: 'What do I benefit from?', actually 'How would this advance my sales?', 'How would this advance my technology?' 'May my product be more attractive to users if I opened it up to everybody else?' "

On the suggestion that creation of adaptors to the HLA standard would solve the problem Heinicke reacts sceptically. The reason behind it is that they already tried it and they did not get the results that they had expected. He thinks that even when having adaptors to some extent for each project "You have to write your own connection using HLA between two different simulation programs and this seems to be really hard coded. So the soft coding, the easy connection is not what I currently see". He states that "There is almost nobody today with knowledge of different COTS simulation packages and HLA. And I think you need to have this knowledge to be able to make the right connection". He refers to the big community who is using COTS simulation packages, namely to the simulation practitioners who daily design and develop simulation models in a particular COTS simulation package. According to Heinicke currently the people who can easily handle integration of COTS simulation models through adaptors to the HLA are especially those people who designed and developed the adaptors.

Both vendors are open to support other standards than HLA if their users request that. Brooks Automation even made an alliance with Simul8 admitting that "Currently in the simulation market there is no 'one' company that can provide complete coverage".

Integrating simulation models developed in COTS through HLA

Both vendors agree with the observation that the HLA standard is rarely used for integrating simulation models designed and developed in COTS simulation packages. Their arguments are related to the high cost and low benefit or as McGregor argues "I can only think that there has not yet been a convincing financial argument for why any particular simulation vendor should modify their product or their documentation to explain to their users why is in their benefit to talk to HLA".

'Idealized' distributed simulation architecture for industry

The 'idealized' distributed simulation that the COTS vendors imagine should be a simple architecture that satisfies the industrial domain. McGregor notices that "There is room for a standard like OPC. In other worlds if there could be a protocol between simulation products perhaps, then why not". McGregor suggests that "You should create a market for it. But I think you could create the market by having a good technical solution". So basically what McGregor means is that a technology push will pull the market.

Both vendors agree on the minimal subset of functionality that the architecture should provide, namely: data representation and exchange, and time synchronisation. Furthermore, McGregor claims that integration should be done "relatively painless", interoperation

should be efficient, the interfaces to the models should be clear and communication between the models should be clear as well.

Summary

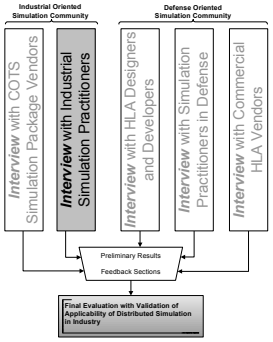
As it stands now, the COTS vendors do not see direct benefits in using distributed simulation to integrate models written using their packages. They see no economic benefit to cooperate with other vendors, and they argue that combining models written in one package can better be achieved in a monolithic way. Applying distributed simulation is also hampered by the way models are built in industry.

Industry "wants results as fast and as cheaply as possible", most models are of a "throw away" type, and they are hardly maintained during their life cycle. COTS vendors also mention the problems around mapping objects from different models and they state that most models are too specific to be reused. Applying HLA is perceived to only generate more problems, because there are no experienced users and HLA is too complex. According to these experts the adaptors available today do not offer a solution to this problem because still too much knowledge is called for to apply this technique.

On the other hand they recognize that combining COTS models using distributed simulation might be fruitful in collaborative design and development, in emulation, in combining simulation with other applications, like ERP, with specialized simulation tools, or in trying to generate speedup through parallelisation.

They believe that a good architecture might generate a technology push. The architecture should be simple, offering only minimal functionality, viz. data representation and exchange and time synchronisation. Furthermore, integration should be relatively painless and efficient, leading to perspicuous models and clear communication between them.

3.5.5.2 Industrial Simulation Practitioners



The participants of this group are both from industry and from the research community. Several researchers from academia are also involved in various industrial projects. It was therefore difficult to distinguish between experts from the academia and the industry. This led us to the decision to combine the two groups. In total we have interviewed 8 industrial oriented experts on distributed simulation. Their names and a specification of their activities can be found in Table 3.14.

Experience in distributed simulation projects

All subjects have been involved in a one till five industrial oriented distributed simulation projects. Their role in these projects covers a wide range such as developer, project manager, leader, IT expert responsible for coupling, etc. For integrating simulation models they applied either the WinSock protocol or the HLA standard. Most of them have appropriate knowledge of COTS simulation packages and general programming languages. Three experts had been active in integrating COTS simulation models through the HLA standard. Rabe integrated Arena, eM-Plant (at that time called Simple++), and Enterprise Dynamics (at that time called Taylor ED) in the IMS MISSION project[24]. Straßburger integrated SLX and Quest in projects at Magdeburg University and at DaimlerChrysler. Further, Van Houten was involved in a research subproject within the Web Integrated Logistics Designer project[25] at Genova University where they integrated Arena supply chain models through HLA.

The other five experts mainly integrated either simulation models created in COTS simulation packages through the WinSock protocol or simulation models created in programming languages through WinSock or HLA. The experts who worked with the HLA standard for their projects all applied the DMSO RTI.

Difference between industrial and defence simulation

Most of the experts agree that there is a difference between the industrial and defence oriented simulation communities, in the sense that the defence simulation community mainly uses general purpose languages, while the industrial simulation community uses COTS simulation packages. Taylor and Ryde agreed with the fact that industry is using COTS simulation packages, but they had some comments regarding the defence simulation community. Both stating that defence is using COTS simulation packages as well. Taylor claims that "When people are talking about military simulation they mean real-time training simulation". However, according to him "There is an overlapping sector" between industry and defence, like manufacturing where they use analytic simulation and there they sometimes consider COTS simulation packages as well.

The experts who agreed with the existing difference offered various descriptions. Straßburger, for example, argues that "COTS simulation packages are oriented towards the

[24] http://www.ims-mission.de/

[25] http://st.itim.unige.it/wild/

application area in the industrial field, while defence is typically building an application from scratch". Lendermann has a similar observation "Industry uses COTS simulation packages to address a large variety of strategic, tactical and (to some extent) operational challenges". According to Rabe "This is a question of the size of the projects. The typical size of a project in the industrial area is between 5.000 and 50.000 euro. It is very rare that projects are larger. If you would start a typical industrial project starting to code something you will run out of your budget very very soon. You are just forced to have something which you can use to rearrange things quite easy and only code some specific algorithms or specific strategies which you need".

Funding is another important issue that might influence the difference in the way distributed simulation is supported. Most experts believe that defence has more budget for doing simulation and this is because simulation has more scope in defence than in industry. On the other hand Taylor remarks that besides defence there are some manufacturing industries, like the semiconductor industry that also have sufficient budget available. According to Ryde the difference can be explained by the distinct objectives of industry and defence: "The objective and "number one" aim of industry is money, while in defence this is not the "number one" objective. In military projects you will find defence priorities, in terms of safety or in terms of secrecy".

Validation appears to have different priorities in defence and industry. Most of the experts believe that validation has more impact in the military area. The reason as Rabe specifies is "Because many of the very small projects have to be finished very fast and maybe nobody will use the models later, only the results are important. While in military these models are very sophisticated in some aspects, very stable, very robust and there is a need for good validation". On this issue Taylor remarks that people might define validation in different ways "I do not think that anybody actually does verification and validation, to be honest, but they say that they do. Sometimes when we talk to the simulation modeller on the street if you like, they say validation and they mean verification".

Reusability

Not all simulation experts have reused simulation models. A lot of the experts did not face this situation, like Taylor who argues that "The reason why I have not done this is because I did not face the same problem again. You could face the same problem again, but it is difficult to reuse something where there does not exists a common standard or common interpretation of a model". According to him the problem with reusability is that "We do not have a common approach to reusability".

Furthermore, people who have reused models are not always convinced of the benefit of it. Rabe, for example, who has reused HLA simulation models, declares that "Reusability has not really profited from HLA. I see some potential profit there, but a prerequisite for that would be that there is a substantial industrial HLA based application, so that when you have one project you find another one, which was already made with HLA in mind in order to reuse something and this is very very unlikely due to the very bad use of HLA today".

According to Malcolm a problem related to reusability of models arises because "In terms of verification there is no formal specification to see how the model behaves based on certain parameters and what the expected outcome is". He observes, however, this generic

reusability problem not only in the simulation domain but in the whole system engineering area as well.

An issue that needs to be resolved when reusing models is mapping the attributes of the object models. The object models that for example Straßburger and his team had to reuse at DaimlerChrysler had different attributes. They managed to map different attributes with mapping tables; in a way that participating federates could stay with their SOMs. Based on his experience Straßburger claims that reusing already existing models needed less effort than reimplementing a new one.

Other experts mentioned projects where reusability is an important issue, from which simulation practitioners could clearly benefit. Lendermann sketches an illustrative example from the semiconductor industry, where there could be existing models of two different factories and there is a need to simulate the material flow between these factories. In such a case the only feasible solution which can be considered is distributed simulation. The reason for that is "that you can put everything into one model but that is totally unpractical because you also need to maintain these models and the people are using them for all kinds of exercises". One of the benefits of distributed simulation as Rabe states is the reusability of existing simulation models. He points to an example where they wanted to see the behaviour of a production system in the context of a supply chain. They already had a model of the production system in one commercial tool and for some reason the supply chain was modelled in another tool. Distributed simulation helped them to integrate these already existing simulation models.

Information hiding

Only one of the experts has faced the problem of information hiding, namely Van Houten. The distributed supply chain simulation models they combined within the team of which he was member of, contained confident information which therefore should not be visible for all partners.

According to Lendermann, information hiding might be an issue in the future, for instance in the case that, for example, "semiconductor manufacturers cooperate with each other". This is similar to the case presented by Van Houten, where every partner is willing to use confidential information because in that way the total quality of a model of a supply chain increases thus ameliorating the benefits of the partners.

Difficulties and benefits of applying the HLA standard

As the experts from this groups are focusing on industrial problems, it is worthwhile to pay attention to their remarks about the kind of problems that they were confronted with or that they observed when they were studying or trying to apply the HLA standard in their projects.

According to Straßburger "The most difficult was getting used to the HLA style of thinking and programming. The complexity of the HLA interface has been an issue and developing low level task like building your attribute updates, and collecting the bytes and bits. These are very low level tasks that you try to get used to". This is also observed by Ryde who states that "Too much effort is required on behalf of simulation practitioner, who may not be a technical person". This is basically because, as he continues, "HLA has a quite low transparency. There is a lot of interaction required by the developer in order to

use it". Malcolm also complains about this issue, he claims that simulation practitioners who intend to use HLA "need to understand the concept, which is complex, before they can even make a very simple model". Furthermore as Straßburger states "HLA APIs are very complex".

A solution to avoid this complexity problem is proposed both by Rabe and Straßburger. Rabe calls the solution "adaptor", a technique which is developed in the IMS MISSION project, while Straßburger calls it "middleware". Both adaptors and middleware aim to hide low level details from the simulation practitioner. Considering the complexity of the HLA standard the need for adaptors is obvious. According to Rabe "If you would not have such an adaptor it would be an extreme barrier, because most of the users are familiar with using building blocks, and maybe coding some strategies in the specific language of the tool. But in order to access HLA you need to understand all the HLA mechanisms, you need to understand the internal scheduling system of the simulation system, because this interferes heavily with the HLA RTI". As a consequence, he believes that "I do not expect that the native application of HLA has any chance in this area". This expectation is also connected to the fact that some of the HLA services, like ownership management, cannot be implemented in the context of COTS simulation packages because these packages do not support it. Straßburger refers to ownership management as "a concept that is not really common to these tools. So, you have to find a way of mapping the logic to the mechanism of the package". Similarly Rabe recognizes the problems around applying the ownership mechanism "You cannot implement it within the basic structure of a COTS simulator without going into the internals of the simulator and then it is no longer a COTS simulation package". According to him this is also something that can be solved to some extent with adaptors.

As another aspect of this complexity issue experts observe that, like Ryde said, "HLA incorporates a lot of functionality that may be not useful to everybody". This observation is in accordance with a statement by Granowetter: "I think the majority of customers of our MÄK RTI use the services that allow you to join and resign from a federation, to publish and subscribe to the object classes and interaction classes, to register objects, send and receive updates and send and receive their interactions. That is what most users, maybe 75% of our users even in the military use, just that subset of HLA services". Taylor points to an article from the Simulation Interoperability Workshop (Crooks*, et al.* 2004) which provides some statistics on the way HLA is used in the defence domain. According to this paper there is indeed a gap between the options provided by the HLA and actual real use. Basically, exactly as Granowetter states, most of the functionality provided by HLA is not used by the simulation practitioners. A solution would be to use less functionality, but then the question is who will be responsible to select this functionality, i.e., the API functions needed, because, as Ryde argues, "It is incorrect to ask a simulation practitioner to be familiar with technical programming aspects".

Concerning the complexity and functionality of HLA Straßburger thinks that "You cannot really do it much smaller and still access the whole community which is addressed in HLA". However, if we would consider only industry he argues "If you say I only want a standard which is right for industrial applications, then you can reduce the standard". Malcolm has the same point of view, he thinks that for industry "you need to find out what kind of functionality they need and use only those". Furthermore he believes that reducing

the functionality would improve performance. He thinks that an open source RTI would be a step forward allowing to experiment with all these issues.

This leads to another important issue, namely the performance of the HLA RTI's. According to Rabe "I would not say that HLA RTI per se has a bad performance, but let us say it is not too easy to use HLA in an efficient way, in order to get good performance". Van Houten has also been confronted with a very slow HLA RTI when he integrated Arena models, that he thinks partially was entailed by the huge overhead due to time synchronisation. The performance problems even led him to switch to a new version of Arena and to improve the communication part of Arena, decision that helped. Malcolm has explicitly analyzed the performance of the DMSO RTI through some experiments. On this matter he has the following observation: "In terms of performance of RTI, but I do not know whether this is only with DMSO RTI or other RTI's, there are bottlenecks in terms of long delay when a federate intends to join a federation and there is also a lot of overhead in broadcasting messages or point to point message sending. So you get the feeling that the implementation is inefficient". The performance problem is also observed by Lendermann, who thinks however, that this is not necessarily an HLA problem but a generic problem in distributed simulation.

Two additional disadvantages observed by Straßburger refer to the interface and the interoperability. According to him "The interface specification is not really unambiguous. There are certain definition problems for interface services, some are missing, etc.". Further, he suggests that there is a need to solve the problem around "RTI to RTI interoperability, because currently you cannot interoperate federates which connect to different RTI's".

One of main benefits of the HLA standard, as all experts mention, is the fact that it is an IEEE standard. Straßburger emphasizes that "HLA is an IEEE standard for industrial applications, even though it has not really been widely accepted, it is a prerequisite that such a standard exists". Rabe notices that "With HLA maybe 50% of the interfaces are standardized, and there are still another 50% where you have all the problems. But 50% is better than nothing". According to Ryde "It is a good idea to have standards but this does not mean that you have to use the standards. There are many examples in many areas, both in industry and defence, where the standards have to be changed. But you need standards as milestones, something to which you can compare and to understand, you can use it as a starting point for development. When you are starting to talk about distributed simulation and intercommunication it is very important to have standards". Getting a solution accepted as a standard is not straightforward. This is emphasized by Taylor who said "There is no difference in my experience to the creation of any other distributed system standard. It is a pain in the neck, it is a very complicated issue".

A special benefit of the HLA standard, as some of the experts mentioned, is that it supports several approaches for time synchronisation and that it takes care of low level communication issues.

Integrating simulation models developed in COTS through HLA

All experts agree with our observation, that the HLA standard is rarely used for integrating simulation models designed and developed in COTS simulation packages. Rabe thinks, however, that this is "already a derivation, because the first observation is that today HLA or general distributed simulation is very rarely used in the industrial domain". As a reason for the fact that distributed simulation is not really applied in industry they mention the difficulty of understanding the benefits the users can gain from it. As Lendermann asserts "You first really need to be able to understand what kind of benefits you are generating, what kind of future exploration is really able to generate some benefits that cannot be realised in a non distributed manner". According to Straßburger "Industry does not understand yet that this is a topic of the future. If you think of topics like global supply networks and the digital factory, they will need it, because they have so many suppliers which build factories together, and they all could provide their simulation models which could be linked together in one huge distributed simulation. I think there is a need for distributed simulation in the future, because globalisation basically results in so many ways of very close cooperation between different entities or different companies". Time is needed to get familiar with distributed system knowledge, as Rabe states, and for that reason we need case studies which can show the necessity and the benefit according to Malcolm.

Straßburger assumes that "When defence designed HLA they did not have in mind any COTS simulation package". Anyway the present situation is relevant now and as Ryde observes: "A lot of people using simulation in practice have no option to use it because they have insufficient knowledge on programming or they do not have the tools available to do this. And of course it is far too costly to create middleware and complicated as well, especially if they do not have the knowledge. So there is a need, but unfortunately the tools do not exist in a simple enough way". Some tools are already available. However, as Rabe claims "The systems available today for distributed simulation are all in prototypic status. I think our adaptor, even if I would say that it is nearly a product, I would still call it somehow prototypic". Taylor observes the same phenomenon describing the adaptors and middleware available today as "workable solutions".

Taylor supports the idea of creating clearer standards for interfacing COTS simulation packages. According to him in creating such a standard the COTS simulation vendors should be involved as well, but as Rabe observes COTS simulation package vendors do not want to take too much initiative because "Simulation vendors are just trying to avoid that somebody uses other systems than theirs".

The scarce use of HLA in the industry might also be influenced by the purchase price of the architecture. Ryde has an interesting observation, regarding the willingness of vendors to pay for it: "From my experience, speaking with vendors and simulation practitioners I have found that most of them are not willing to pay much more extra for the privilege of interoperating simulation models. Many would be willing to pay an extra typically between 5 to 10% of the original price of the package". He argues that "The industrial domain requires low cost simulation packages, somewhat like the PC world. The future will be for the cheap packages and not the expensive ones. This is the history of computer software. In that context, I do not believe that HLA has a place, purely on the cost basis. In

terms of functionality as well, I do not think so that all functionality is required in that particular domain. The RTI costs too much and the customers are willing to pay at most 10% - the vendors do not want to take steps into this direction". Taylor had also discussions with vendors and based on these discussions he states that "the new features added to the package will result in it either being sold for a higher price as an add-on or being incorporated in the maintenance fee. Which is also going to be a problem", since "If vendors develop their own version of an RTI, I mean incorporate it in their package then the cost may be hidden. If they use another RTI they have to pay for the license of the RTI". On the other hand, according to Rabe and Lendermann cost is not really an issue, because the 3000 to 5000 euro RTI license is not so much compared to some of the simulation packages, which cost 30.000 euro.

'Idealized' distributed simulation architecture for industry

The experts have interesting visions regarding an 'idealized' distributed simulation architecture for the industrial domain. Here is an idea by Taylor: "Instead of having a notion of central middleware, you have a peer-to-peer implementation where each federate is capable of running in standalone mode or in network mode, in the same way that one would use remote procedure calls or remote method invocations. So the programmers when they create their program, or the modellers when they create their model, are using various things that might have a local or a remote implementation, they do not know. However, when the models are brought together, the model has a nice little tool they can use to visualize the whole thing, similar to what they are doing with hierarchical modelling, so they can apply the principles and practices they have now. Things like lookahead or automated regeneration they do not know. So end users would be using distributed simulation, and they will just call it simulation modelling, instead of distributed simulation. The architecture itself would be based on a standard". Malcolm has more or less the same wish for an idealized distributed simulation architecture. He thinks that all distributed concepts should be "transparent to the end user, the end user should use the COTS distributed simulation model as a standard standalone COTS simulation model". Rabe believes that "We have to find some type of standard or quasi standard for object description". Straßburger agrees with Rabe, he argues that "The industry has to sit together to define reference FOMs. It should really have to be industry activities and standardisation efforts on the application side and also on the tool development side".

The opinions of the experts diverge regarding whether and in what way the HLA concept should be integrated in COTS simulation packages. Most of the people think that the best option is that the COTS vendors would embed the HLA concept in their packages. Straßburger, for example, believes that HLA "should be integrated within the package. It should be integrated in the modelling paradigm of your tool". He imagines a simulation world where "Every COTS simulation package vendor should offer a module for HLA connectivity". On the other hand, Rabe argues that "implementing the concept of HLA in a commercial tool is not beneficial to the end user. Because then the end user in fact has again to deal with HLA and the HLA mechanisms". He thinks that the best solution is to use separate adaptors because with complete integration "you save the adaptor, but you are even less open than before, than having the adaptor as a separate tool. You could even run

the same commercial system with 2 or 3 types of adaptors[26] if it is necessary and if the adaptor is inside the tool you are limited to one adaptor type". He underpins his point of view by arguing that "the RTI's which are now on the market are not really compatible. That means if you exchange the RTI you have to change a little bit in the adaptor. That means if the adaptor is integrated in the commercial tool you are limited to one specific commercial RTI".

Regarding the minimal functionality that a distributed simulation architecture should provide in industry, experts envision solutions for the problem looking from different design levels. According to Straßburger the set should consists of "time synchronisation and entity passing". Malcolm extends this based on his software engineering experience: "Joining and resigning from the simulation run, publishing and subscribing for objects, and time synchronisation." He considers the rest of the HLA functionality, which is not really used (e.g., data distribution management, ownership management, etc.) as secondary.

Summary

A first relevant observation that was made several times by the industrial simulation practitioners is that there is a need to separate issues around HLA and COTS from problems that are actually rooted on a higher level, viz. distributed simulation in general.

The experts saw a clear difference in scope of the projects in defence and in industry. Industry is mainly interested in generating direct results, money is the big factor here. Projects are smaller than in defence and in general the models are not reused. COTS packages are designed for such applications. Defence has other objectives, safety and secrecy of the models are relevant, the models are sophisticated, stable and robust. Verification and validation is important, and mostly general purpose programming languages are used.

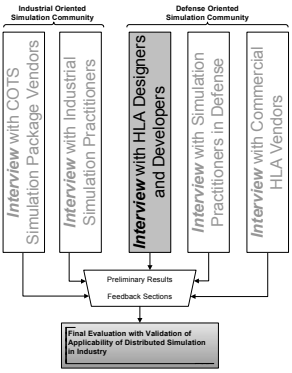
The industrial simulation practitioners mentioned several drawbacks of the HLA standard. One of these drawbacks is the performance of the current implementations. Experts relate this to the fact that HLA offers too much irrelevant functionality for industry. Minimal functionality for industry would be only time synchronisation and entity passing, possibly augmented with the possibility to join and resign from a simulation run and to publish and subscribe to objects. Other problems that were mentioned by practitioners were incompatibility of RTI's and some ambiguities in the HLA specification. Furthermore, the fact that ownership is a notion that is not implemented in COTS packages causes additional problems when one intends to apply this HLA feature. In addition the mapping problem is mentioned by the industrial simulation practitioners as well. This might be solved by defining standard object descriptions. In this process, however, COTS vendors should be involved. Finally there is the issue of cost. There is a trend towards low priced software packages. Adding the cost of integrating or interfacing to an RTI will increase cost and the customers, as it stands now, are not willing to pay more than 10% for this. On the other hand a remark was made that, given the prices of COTS packages and RTI's, this is attainable.

[26] For the definition of adaptor look at the Glossary

Although several examples from industry are mentioned which can benefit from distributed simulation, such as the virtual factory, material flow between two factories, and combining models of production systems with a model of a supply chain, currently there do not exist proper tools to solve the problems involved. These are typical applications where cooperation is of benefit to all partners involved. It is expected that in the future these type of projects will gain importance, and additionally further globalisation will generate a need for more cooperation. The industrial simulation practitioners perceive that distributed simulation has a place in industry, however, industry needs to be convinced on the benefits of distributed simulation. In order to realize this, time and realistic cases are needed.

A big hindrance is that distributed simulation and HLA are complex phenomena of a technical nature, whereas the practitioners in industry are generally not technically inclined. Although HLA takes care of quite some low level technicalities its interface is still too complex for industry. On the question whether adaptors and middleware solve this problem the opinions diverge. Even if HLA is integrated into the modelling paradigm of COTS packages, the user is still exposed to the HLA functionality. Using stand alone adaptors, that would also avoid being tied to a single RTI implementation, might give some relief here according to some experts. Others suggest to aim at an architecture that gives an interface much like a stand alone COTS implementation using which one can specify at run time whether and how the model should be distributed.

3.5.5.3 HLA Designers and Developers



We have selected two experts who were involved in designing and developing HLA. Katherine L. Morse was member of the technical support team for HLA in which she was responsible for the design of the HLA data distribution management services. The second selected expert, Richard Weatherly, wrote the original HLA interface specification and led the HLA infrastructure development.

Experience in distributed simulation projects

Besides designing and developing HLA, both experts have been involved in distributed simulation projects for more than 15 years, and in average they have carried out approximately 10-15 large distributed simulation projects. All of these projects were defence-oriented and they were designed and developed in general programming languages, such as Java, C++, FORTRAN, ADA, etc. For those distributed simulation projects where the HLA standard was used the experts applied DMSO-RTI 1.3 (DMSO 1998b) which is owned by the US Department of Defence. Currently the defence is still using this HLA implementation.

Difference between industrial and defence simulation

Both interviewed subjects agree that defence simulation communities mainly use general purpose languages, while the industrial simulation communities use COTS simulation packages for creating simulation models. Morse thinks that there are two reasons behind this: “one of them technical and one of them having to do with the business model”. Regarding the first reason Morse argues that “By their varying nature, the defence simulation models typically require more technical depth”, apparently implying that COTS simulation packages do not deliver this, while in the commercial world as Weatherly states “There is enough market for the same general kind of simulation, repeatedly applied”, that is using high level building blocks which are typically included in COTS simulation packages. The second reason has to do with the business model, Morse argues that, “In commercial and industrial simulation you are producing a model to be sold. Key to making profit is not just selling the model, but producing as cheaply as possible. So, the idea of using a simulation package makes sense because it is a cost saver in developing a model”. In contrast to the industrial simulations, the defence simulations have a tendency to be run on the long-term and to be very expensive. Weatherly argues that “There is a real problem with the requirements development process for military simulation. They try to build a small number of very large simulations, and that causes those simulations to have to answer a lot of requirements and so they become very complicated and very expensive.”
Weatherly points to another important issue between the defence and industrial simulation: “Military simulations have a high degree of interaction between the simulated entities. If you build a factory simulation you have for instance an entity that is a particular machine tool. That interacts with something that the tool is getting parts from and with something else the tool is sending parts to, so just a small number of things. The relationship between these entities is well known. You have an assembly line, while in the military, the entities on the battlefield can all affect each other, they can all be seen by a satellite in space or a

large number of entities can be damaged by a single weapon, and can be damaged by a weapon that they are not even aware of. So the degree of interoperability between entities and combat simulations has a tendency to make it more difficult than a lot of the simulations that most simulation vendors build which are simulations of networks, highways, factories, etc."

Reusability

Both Morse and Weatherly have reused existing simulation models in different projects. Weatherly considers HLA as a great tool for this approach: "The HLA attempts to encourage reuse at a very coarse level that would be the level of entire independent simulations rather than pieces or parts within any given simulation". But reusability of simulation models, even when an HLA is available is not an easy issue. Both of them had problems with the semantics on the interfaces. Morse and Weatherly consider the alignment of federation object models to be the biggest challenge in hooking simulations. The most difficult part is "What is the semantics and what is going to be exchanged?", as Weatherly said, because "The simulations were different in the first place from some reasons. So it is usually the case that a compromise is required to come up with an object model that satisfied all the parties involved so they can interoperate."

The experts do not think that it would have been easier to completely reimplement the reused model. Weatherly argues that "This is not a statement about how easy the interface is, it is more or less a statement about how difficult these models are to create in the first place." Not only the designing and developing processes but also the verification, validation and accreditation is very significant. According to Weatherly "If a model exists it might have been certified by some organisation as being a credible, believable representation of some phenomenon so in order for your results to be credible you need to use these credible models. So reimplementation is often not an option." Morse pointed to a statistics collected by DMSO regarding the cost and the amount of time needed to transfer a monolithic simulation model to an HLA compliant simulation model for reusability purposes. We have contacted DMSO and Mark Crooks, an HLA testing manager provided the following information: "The average time for a federate to meet this criterion is between 4 and 5 months. There are long periods due to rewriting of code and changing a simulator from a standalone one to being part of a federate".

Information hiding

Besides reusability, another issue in distributed projects is information hiding, especially when different organisations are involved in a project. However, in defence, as Weatherly said "You have just the defence community and the defence community contracts with multiple private companies to build things for it. So most defence simulation is actually built by private customer contractors for the defence." Morse has similar remarks: "Military owns all of these models and it pays the contractors to develop them and although the contractors maintain them, in fact the government owns them". On the other hand Weatherly argues that there are situations when information hiding can be an important issue because "Very often organisations for both good and bad reasons are reluctant to share how their simulation works. Either it might be classified, is a military secret, or they might be embarrassed about how their simulation is actually written."

Difficulties and benefits of applying the HLA standard

On the complexity of the HLA standard Morse agrees to a certain extent. Her point of view is that complexity is avoidable by using middleware: "People can build middleware layers that can abstract from the complexity in a way that is appropriate for their specific domain. So, I can imagine that commercial products would probably abstract that out in their tool for the users." Weatherly argues that "The HLA specification provides all the services in a great way and that it is maybe a weakness of the HLA that it does so much. It is frightening to people who first approach it". Regarding the heavyweight structure of the HLA, Weatherly considers this an argument in favour of HLA, because there is a need for a lot of services due to the complexity behind distributed simulation.

Morse complains about the aligning of data models, which is not so much an HLA problem, but more a general problem in distributed modelling.

Regarding the performance of HLA both experts are satisfied. In some projects Morse had the chance to carry out "a simulation with tens of thousands of entities" and it worked properly. Weatherly gives some explanations for cases when HLA is thought to slow down: "I suspect that there are people who are running simulations that do visualisation and they are refreshing their display frequently, they may feel and have experience that shows that the particular RTI that they are using imposes latencies that messes up their simulation, which can be true."

An issue that Weatherly would like to see in the next version of the HLA specification is support for efficient bridges between HLA federations.

Both Weatherly and Morse agreed that it would be great to have an open source RTI. Morse sees the following reasons for that: "First of all open sources provide a key component on the market, secondly it keeps people honest". Weatherly thinks that "HLA is mature enough now to allow an open source implementation to be available, and that would really help".

Regarding the benefits of HLA, the interviewed experts both agreed on the fact that HLA solves a number of mechanical issues for the simulation practitioner regarding low level distributed services, such as time management, data exchange, ownership management, etc., which before the HLA standard was not really solved.

Requirement list of HLA

The experts do not know about any official requirement list for the HLA. Weatherly states that "There is no HLA formal requirement document, except that the under secretary of Defence said: DMSO build me a system that unifies DIS and ALSP that can be used across the department of defence".

Integrating simulation models developed in COTS through HLA

The interviewed HLA designers admit that the HLA standard is rarely used for integrating simulation models designed and developed in COTS simulation packages. Morse suspects that this observation holds because "The focus in industrial simulation is less on distributed simulation than the Department of Defence does. So, it does not make lot of sense to make HLA compliant simulations if you are doing stand alone simulation." Weatherly notices: "If you are the developer of a COTS simulation package or general

framework, you are trying to argue that your package is extremely general purpose, does everything under the sun, it has all the capabilities, so this makes it hard for you to envision how simulations built in your package are going to deal with other simulations across the HLA." Further, Weatherly argues that, probably "HLA seems impenetrable or monolithic to them (to the COTS vendors) and things would need to be done to make, I think, its evolution and management a bit more dynamic and a bit more focused to those who are actually using it".

A remark that Weatherly makes related to this topic is: "Who should really worry about the HLA? Should it be people who build simulation packages or simulation frameworks who expect others to use it? This is not the population of people who generally build simulations". Thus, he concludes that "So it is more likely the simulation infrastructure builders would be more interested in HLA than just general simulation model builders".

Weatherly believes that, on the other hand, the defence simulation community does not focus on COTS simulation packages because "The defence simulation developers would rather develop everything from scratch. There are a lot of good reasons for it too, because a lot of these big military simulations are really running on the edge of what can be done from a performance standpoint, and you cannot tune your system unless you own it all. So *the developers of these big simulation systems are just suspicious of the COTS simulation environment, and the COTS simulation environment people are suspicious of the HLA*."

'Idealized' distributed simulation architecture for industry

Regarding an 'idealized' distributed simulation architecture for the industrial domain the experts looked at this matter from two points of view. Morse states that "First of all it should be web enabled because business has really landed on development of web technologies. I think that the architecture will need to use some of the architectures developed for e-Business. For discovery use repositories, so that people in industry can go out and find services that they want to use." With this approach there is a need to change the business model: instead of buying COTS simulation packages the end user would buy services.

Weatherly thinks that an idealized architecture for industry "needs just a subset of what is in HLA already". He envisions a subset of HLA with reduced functionality for a given community: "It might be that you pick the particular vendors and you look at all the things the people do with their simulation packages: here is the limit of what they can do so here is the limit of the services of what they need". He compares the current implementation of RTI to a distributed operating system. He imagines to reconsider this structure as "a centralized RTI server, which is much more like what is happening right now in the web world, application web service, everything runs on it and everybody connects to it, using light weight clients. *Keep the federate, keep the simulations light, keep the RTI's out of their machine and put all the RTI's on a centralized server.* So I would think that would be a fruitful path to take a look at. Furthermore it would influence your economic model. Having high speed RTI services you might charge people by the hour instead of buying a license."

Summary

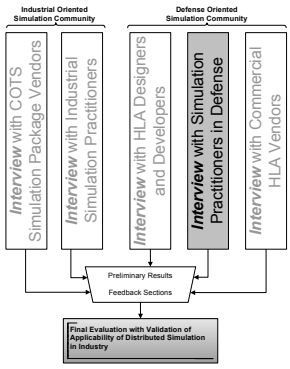
The experts provided insightful comments on the difference between simulation models built by defence and industry. In defence they observe a relatively small number of complex and expensive models. The complexity is related to the fact that there is much interaction between the constituting parts of the models, much more than in industrial models. Military models are often running on the edge of what is possible. This mandates full control over the environment explaining the choice of general purpose languages. Due to the complexity extensive requirement analysis is customary which leads to high credibility of the models. This, and the high building costs involved makes full model reuse feasible. Although the experts observe problems here, mainly related to semantic interoperability, reimplementation is not an issue. HLA is experienced as a good tool solving many low level problems.

In industry HLA designers see many models being built of a similar kind. The most important issue here is cost which has to be kept low. These are the two main reasons why the standard approach in industry is to use COTS packages. They perceive industry as less focused on distributed simulation which explains their observation that COTS and HLA hardly go together. An additional argument is that HLA is developed for the military and that typical industry related aspects have not been taken into account when designing this standard. Subsetting HLA towards the functionality needed by industry might be a good approach. In the opinion of the HLA developers, adopting HLA into COTS models is not a task for the model builder but for the package developers.

Another difference between the two application domains is that in defence there is one big player, the US DoD. This explains that in many cases, but not all, information hiding is not a big issue. In industry the experts see many players who are reluctant to cooperate out of fear to lose customers, which hampers development into the direction of coupling models written in different packages.

The experts involved in designing HLA envision a future in which ideas related to the use of Internet are implemented, repositories offering services that a customer may use, and lightweight clients. This entails a new business model in which clients are charged for services instead of packages.

3.5.5.4 Simulation Practitioners in Defence



This section aims to describe the interviews we have conducted with simulation practitioners from the he defence community. We have invited two experts to participate in the survey, namely Rick Severinghaus and Wim Huiskamp. Severinghaus is working on defence oriented simulation projects at Dynamic Animation Systems, Inc., an organisation which is dedicated to the development of distributed, interactive, and immersive virtual reality applications [27]. He is further active within SISO, as a member of the SISO Executive Committee. Huiskamp is working at TNO Physics and Electronics Laboratory (TNO-FEL)[28], which is part of the Netherlands Organization of Applied Scientific Research (TNO). TNO-FEL has the mission to carry out projects for Defence, Public Safety, ICT, Transport and Logistics, Aerospace or Electronic Systems. Huiskamp is involved in designing and developing a defence oriented simulation system within the Command & Control and Simulation Division department.

Experience in distributed simulation projects

Both experts have been involved in a number of defence oriented distributed simulation projects, typically 5 to 12. In the majority of the cases they had the role of coordinator or project leader. In order to integrate various simulation models they mostly applied the HLA standard and sometimes the DIS. The RTI that they used for HLA based projects was mainly the DMSO-RTI vs. 1.3 and in a few cases the MÄK-RTI. Additionally Huiskamp applied the TNO RTI, which is an RTI version developed by TNO-FEL. Simulation models have been mainly implemented in general purpose programming languages, such as C++ or Java, and only rarely in defence oriented COTS simulation tools, like the tools provided by MÄK.

Difference between industrial and defence simulation

The interviewed experts agree that there is a difference between the industrial and defence oriented simulation communities, in the sense that defence simulation communities mainly use general purpose languages for creating simulation models, while industrial simulation communities use COTS simulation packages. According to Huiskamp the reason behind this difference stems from the scope of the projects in these areas: the industry focuses on direct applicability of simulation by repeatedly solving similar problems, while the defence is more research oriented. So, “It is less application and more research and then you need more control over your tools and programming environment”. Further, he emphasizes the security aspects: “In defence there is more attention to security aspects. There is maybe more attention for validation of your simulations, especially if you are using them for training or acquisition applications”.

[27] See http://www.d-a-s.com/

[28] See http://www.tno.nl/instit/fel/

Severinghaus's opinion is similar, in addition he states that "In defence the scale is often much larger".

Reusability

The reusability of existing models was an important issue for both experts. They state however that it is not a straightforward task. As Huiskamp claims "There are always problems". He argues "Most of the time usually there is a need for some additional output or input in your model and you must modify your model". This is also observed by Severinghaus who draws our attention to the issue that "You have to do some coding integration work before you can make that happen in each case". However, the grade of difficulty "depends on the application" as well, as Huiskamp notices. Based on their experience, "If you stay in the same application area it is maybe 10 to 20% of the original design time, while if you want to move to a completely different application area, then it can be up to 100%, and sometimes you have to start again from scratch". The biggest problem the experts have both been confronted with, is the case when the data model does not fit. If that is the case "You need to change the HLA data model, the FOM of your model and you need to modify the internal workings of your model", concludes Huiskamp.

Information hiding

Besides reusability, information hiding was an important issue as well in the projects in which the experts participated. In some of the projects managed by Severinghaus the organisations involved intended to "protect their actual capital and expertise". Huiskamp was involved in a project with 30 different participants from 6 nations, and given the scope of the project information hiding was "a big issue, because that is also a classified project, and the models are classified, so the information that is exchanged is very restricted. You need to have a lot of agreements and contracts between the nations in place, stating exactly who has access to the data and what you can do with it, who will log the information, where and how it will be stored, etc". Information hiding in this case was solved with the help of a management plan, which "defines all the protocols and all the procedures". However, in the end it was more or less the responsibility of each participating nation to agree how to share the information with the other participants, "So when you publish any information it is your problem that you do not want others to see it". As Severinghaus points out, the problem can actually be bigger in that the participants do not even want to share all information: "In some distributed simulations in military applications there are valid concerns for not putting into the simulation some of the things that you use in the battlefield. It is depending on who you are going to network with. It is protecting the operational technique".

Difficulties and benefits of applying the HLA standard

Regarding the difficulties of applying the HLA standard, Huiskamp mentions that "There are multiple problems of course, but it is not always completely an HLA problem, it is basically a problem of distributed simulation and distributed applications where you need to interface and interoperate with other tools". According to Huiskamp the documentation of HLA "is sufficient, but this does not mean that is easy". He thinks that the simulation practitioner needs some time to get used to HLA, "It has a high learning curve". One way to avoid this problem is to simplify everything, for example through good middleware: "It would be nice if you would have good development tools and if you can sort or switch off

the different levels of complexity. You can have a sort of novice user, and they would see only the main buttons and the main functionality and the rest of it would be taken care of by the system or by the middleware which gives you default values, for example. They don't have to worry about it. And if you have a more advanced or more experienced user then he would get additional features and more control over the fields that he needs to have control over".

Severinghaus agrees that the learning curve of HLA is steep, it takes a long time to learn. Regarding defence Severinghaus thinks that "There are some issues you have to work through, but complexity itself is not the issue". On the other hand he argues that "If I am coming out from an industry perspective I would totally agree, that yes it is an issue. We do not need all this functionality".

Severinghaus points out that the problems with HLA that come up and mostly occur in defence are related to "the invisibility of the actual RTI code", referring to the fact that RTI hides low level detail. Further, often simulation models are not designed with interfacing to others in mind, which might cause additional problems. Severinghaus often faced these in the projects "We focused on trying to figure out which entities map from one simulation to other". If this process takes too long then "In many cases the issue of tying together led to the fact that there is not enough time, or budget, in some cases expertise to do all of that, so basically there is no benefit".

The performance of the HLA implementation does not seem to be sufficient for these experts. The opinion of Severinghaus regarding the performance of DMSO-RTI is that "For a lot of training and training related distributed simulations HLA performance is satisfactory. But, I have seen, heard and been in discussions where that is not the case for other applications, principally in the engineering and research development, it just doesn't keep up". According to Huiskamp "It still should be improved. I do not see a difference between the different vendors, but the DMSO RTI had a pretty poor performance and very little control over the performance". He observed that some of the other vendors like Pitch or MÄK offer products with better performance and more, easier control over the performance, but he thinks that "there is a lot room for improvement". He believes that "Some of the improvements can be achieved by giving you more control over the features of HLA, stuff like time management or some of the other features. Not everybody uses them and needs them and they still cost lot of performance, I think or so it seems. And we would like to have more control over that. If you want to use those features they should not be that expensive in terms of performance". At TNO-FEL they have designed their own TNO RTI, which is a shared memory version that is much faster than the DMSO RTI and they have full control over it. The implementation of their own RTI adheres to the HLA standard, shows the full behaviour and embraces the whole concept, but they added more time management functionality to cover a real-time scheduling problem that could not be solved with DMSO-RTI (Jansen*, et al.* 2004).

As an additional shortcoming of HLA, Huiskamp notices that there "is still not enough attention given to the hierarchical approach". To some extent the SIMULTAAN project (Brassé*, et al.* 1999) of TNO solves this problem by trying to extend the principles of HLA to a higher level, the component level. As Huiskamp describes "The idea was that you have a sort of a hierarchical approach to a federation, so you can sort of zoom in into a federate and you see another federation of the component that actually make up the

federate and within that federate you have the same HLA structure basically of the components communicating with each other and interoperating with each other". So a federate can be one monolithic application or it can be built as a hierarchical architecture or structure of multiple components that together actually represent the behaviour of the whole federate. In order to achieve this there proved to be a need for an RTI with full control. That was one of the reasons why TNO-FEL designed and developed their own RTI version.

Another shortcoming that Huiskamp notices is the limited data model provided by the HLA standard. He suggests that "there is even more information that can become part of the data model. For example, that you can define for each attribute more information, like how it is encoded, what are the security levels, etc. So you get more control over that and you still have all this information at one central location. And I think this is more or less possible with this new XML format where you can easily extend or basically add fields to the federation object model". Accordingly "For each attribute there should be more information and more control over what it represents and how the data should cross federates". Huiskamp believes that extending the data model would probably help to achieve an appropriate secure communication between federates, by introducing multilevel security. Currently it is not possible to let certain federates see some information while disallowing this to others. There is no native or built-in way within the HLA to realize that.

A benefit of HLA mentioned both by Huiskamp and Severinghaus is the fact that HLA is a standard. This provides an opportunity for designers from different organisations to talk with each other because they follow the same approach and they are talking the same language.

Integrating simulation models developed in COTS through HLA

Both experts agree with our observation, that the HLA standard is rarely used for integrating simulation models designed and developed in COTS simulation packages. They think that this phenomenon is due to the unfamiliarity of the industry with HLA. Huiskamp thinks "that in industry HLA is probably still more or less unknown to many people". Severinghaus similarly believes "That is partly because a lot of commercial industry has not really been exposed to HLA, but I think commercial industry is looking for an integration standard". On the other hand, according to him the DMSO made some efforts to introduce HLA to industry. He thinks that "In principle the DMSO effort was intended for the US DoD and translatable and extendable into the commercial area, but the history of DMSO development was basically focused on the need of defence simulation. So, if I would have to characterize, I would say the hope was that it would carry over in practice, but I do not think that it worked that well".

'Idealized' distributed simulation architecture for industry

According to Huiskamp an 'idealized' distributed simulation architecture for industry should be "based on a standard architecture and on standard methods, design and development methods, like proposed by HLA, or an improved version of HLA, and then the whole thing should be supported by a development environment that also allows novice users or less experienced users to create distributed simulation applications. And that means that he does not need to know more than what he needs to know for his specific problem and if he wants to achieve higher performance or has more requirements then of

course he has to dig deeper but if he has a relatively simple problem it should be easy to create his distributed simulation without knowing more details". The interface to this architecture "should be a sort of graphical interface where you can plug and play or drag and drop models". Further, Huiskamp imagines "a sort of a layered approach: you do not need to go any deeper than what you need for your problem, but still have the reinsurance that what you are doing is based on standards and can be extended in the future if you become an expert user or if you have more requirements".

Severinghaus envisages a simplified distributed simulation architecture: "You build up modules of defined functionality, reasonably defined and you put those together, and just buy the complementary ones that you really need".

The minimum subset of functionality needed for a distributed architecture is, according to Huiskamp, the possibility to exchange information (e.g., the publish/subscribe mechanism), functionality to control the execution of the simulation (e.g., start, stop, etc.), and time management mechanisms. If there are additional needs, like security, then they should be implemented one level deeper.

Summary

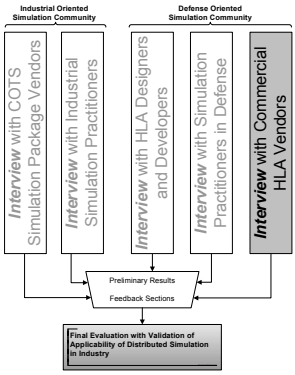
The defence simulation practitioners mention that they see simulation modelling on a larger scale in defence than in industry, and that military simulation modelling is more research like, which entails that more control is needed over tools and the environment. This explains why defence uses general purpose languages. They also observe more concern in the military domain on security aspects which leads to more emphasis on validation.

They agree that reuse is important, but again the semantic interoperability problem is emphasized. Submodels are often not designed with networking in mind with the effect that the data models do not fit together. This leads to the need to update the FOM of models in the case of applying HLA, and to change their internal workings. Sometimes one has to start anew from scratch, and it can even be the case that there is not enough time, budget or expertise to combine the submodels at all.

Concerning information hiding the defence practitioners state that it was a big issue in their projects. The experts from defence perceive complexity, lack of transparency and insufficient performance as the most severe drawbacks of HLA. The complexity cannot be avoided because distributed simulation in itself is complex. They define transparency similar to the HLA vendors as the possibility to access the levels below the HLA RTI. Insufficient performance led one of the experts to implement a new version of RTI. Hierarchical federations and the option to specify more elaborate data models were mentioned as useful additions to the HLA standard. They perceive as the biggest advantage of HLA that it is a standard.

Experts from defence propose an idealized architecture that hides as much complexity as possible from the novice user offering only basic functionality, viz. data exchange, time management, and execution control. Further, experienced users should be allowed more functionality and access to deeper layers enabling him to change default settings.

3.5.5.5 Commercial HLA Vendors



In this survey we have considered two major vendors, Pitch AB represented by Björn Möller, and MÄK Technologies represented by Len Granowetter. Both of them were involved in designing and developing commercially available HLA implementations and are occupied with selling them. These vendors provide tools, such as pRTI 1516, Visual OMT 1516, 1516 Adapter, etc. by Pitch AB[29] and MÄK-RTI, MÄK VR-Link, MÄK Gateway, etc. by MÄK Technologies[30], which are used for distributed simulation.

Experience in distributed simulation projects

The vendors have been indirectly involved in hundreds of distributed simulation projects and for integrating simulation models they have worked with their commercial RTI product, DMSO-RTI 1.3, IEEE 1516, and sometimes DIS.

Support for the industrial domain

Most of the customers of these vendors are from defence; however some of their products have been applied in industrial oriented distributed simulation projects as well. Möller points to some large industrial projects where their product was applied such as the Japanese manufacturing industry, and ship manufacturing for the oil industry. Both vendors mention the space industry, the NASA, as an important customer using distributed simulation.

The experts have not been directly in touch with industrial oriented COTS simulation packages because the models that they integrate are either designed in general programming languages or using defence oriented COTS simulation packages (e.g., computer generated forces toolkits).

Both vendors claim that industry would benefit from middleware or adaptors that they provide for easier model integration. For example MÄK provides VR-Link, which is a simulation interoperability middleware toolkit supporting both DIS and HLA with the same API, situated one level above the RTI.

The vendors are open for collaboration with the industrial world and with COTS simulation package vendors. Although MÄK has now started to build a relationship with The MathWorks (vendors of MATLAB), branching out of the military market is a relatively recent choice. As Granowetter argues, “Historically, our company has not really targeted that segment of the market very hard. We haven't gone to conferences for people who are doing industrial simulation, and have focused our marketing efforts towards the military”.

[29] For more product information see http://www.pitch.se/

[30] For more product information see http://www.mak.com/products.htm

Granowetter's observation is that "The majority of the HLA users are in the military, because of the requirements of different countries, where departments of defence prescribe you to use HLA. So a mandate stating that 'you must use HLA' opened a market for it". However not all defence simulation practitioners welcome this mandate. Weatherly, for example, observed a negative reaction after the appearance of this mandate: "Very often people are searching for reasons not to use HLA or find problems with it, because they say: If I can find a problem with HLA then I will not be forced to spend money to change my simulation to use it." But anyway after a while (probably thanks to the mandate) HLA was accepted and as Granowetter stated it has "certainly created a market within US Defence". According to Granowetter some similar market force may be needed in order to jumpstart demand for HLA in other industries.

Difference between industrial and defence simulation

Although these vendors are oriented towards defence they believe that they can state that there is a difference between the industrial and defence simulation community. They cannot say too much about industry but regarding the defence they agree with the thesis that the majority of the simulation models are implemented by programming languages. Most of the large simulation models as Möller said "use traditional languages, mostly C++ and Java and some of the systems are built in FORTRAN as well". For those industrial projects for which HLA was considered, for example in the oil industry, customers "use things like old FORTRAN models that they have built on their own."

Reusability

The experts agree that models are often reused in defence. Möller pointed to one of their customers, Saab, the Swedish airplane industry, who made efforts for adapting their existing models to HLA and then reused the adapted systems in many other contexts. The biggest difficulty in achieving reusability, as Granowetter stated, is aligning the federation object model.

Information hiding

A nice example of information hiding was given by Möller who points to Simulation Based Acquisition (SBA), which is not a new phenomenon in defence. SBA is a "strategy that employs a number of new technologies, including the robust use of simulation, to address what we buy and how we buy it." (Albergo and Thomen 2000). Möller noticed "When the military wants to buy a new airplane and they want to use it in a specific operation, they go to the different manufactures and they say: could you suggest an airplane and a price. What we want is not only a specification on paper, but we want a specification which is an executable model and maybe one that is exercised over the Internet. So you connect through the Internet and the defence organisation can evaluate how this airplane will work in their operation. Can it fly long enough? Can it fly fast enough? Can it turn around quick enough? What they get is information about the model and its external behaviour but they do not know how the airplane was built. Finally when they decide to buy from one of the airplane vendors they already know what will be produced and what they can expect". Möller believes that buying things based on simulation will also gather ground in industry; he thinks that "it is happening right now in defence, but in the future it will occur to a bigger extent in the civilian area".

Difficulties and benefits of applying the HLA standard

From their own experience in defence simulation, the experts could make valuable observations regarding the difficulties and benefits of using the HLA standard.

Granowetter thinks that one of the biggest difficulties is "getting your base data model to match, getting your base FOM's to match. To come up with FOM's that actually make sense, that are easy to use, that have data representations that are easy to understand." Further, Möller notices that for the simulation practitioners who intend to apply HLA "a smaller or bigger nightmare is that it takes some time to understand it". Granowetter has the same opinion, he states that "the learning curve is kind of high, and that is a fair criticism on HLA". Möller believes that HLA requires people able to think at a high level, because they need to understand the concepts of modelling, information models, architectures, etc. As he argues even if people "know the programming level good enough, they need to understand system architectures and how to create models of the world around them".

Regarding the documentation Granowetter thinks that it can be one of the keys to figure out how to use the HLA. He has the opinion that the "HLA standard is pretty well documented. It is large and complex, which sometimes is a barrier or it is hard to get over". This relates to what Möller observed above, even if it is well documented you need the knowledge to understand it.

Regarding the performance, the experts conclude that the different HLA implementations perform differently. As Granowetter said this is one of the technical issues they compete with DMSO RTI. A comparison of MÄK RTI and DMSO RTI vs. 1.3. can be found in (Burks*, et al.* 2002), which reports that MÄK RTI performs much better.

Granowetter draws attention to another technical issue that they introduced in their MÄK RTI product, namely the extendibility of functionality, which is largely absent in the DMSO RTI.

There are some additional features that vendors would like to see in the next version of HLA. For example Möller would like to have a better fault tolerance. "Maybe somebody cuts the network wire, or maybe there is a problem with one of the computers, how can the whole simulation keep running when one system is not working properly?" Further he would prefer a better authentication to have more security. "There is usually no security problem with HLA when you run it in defence environment, because in the defence environment you have a secret network. If you are on the secret network you do not have a security problem. However if you want to run HLA over standard networks, you need to add more security, because then you are running on open networks"

A disadvantage of HLA, according to Granowetter shows up when "not all the federations are using a common RTI implementation. I think when different companies are involved in simulation that can cause difficulties".

Regarding the transparency of the architecture, simulation practitioners from defence community complain that HLA is not transparent in the sense that they cannot see what is happening below the RTI. In order to solve this problem MÄK designed a tool called RTI Spy which is on the top of the RTI functionality. This allows the simulation practitioner to analyze and to generate debug information, to help federation developers to find out what

is going on, to observe where there is a connectivity problem, who is not making a right RTI call, etc.

Besides the disadvantages mentioned above, of course, HLA has its own benefit. According to Granowetter "One of the most important benefits of HLA is that you can write a federate without any knowledge of wire communication or some other network infrastructure knowledge". Möller considers HLA as a "very general simulation architecture that is not locked to one specific area" and "You can implement the communication layer in many different ways".

Integrating simulation models developed in COTS through HLA

Regarding our second observation, namely "The HLA standard is rarely used for integrating simulation models designed and developed in COTS simulation packages" both of the vendors agreed. The reason for that, according to Möller is, that in the civilian market "HLA is still reasonable new and there have not been many forces bringing it outside the defence market yet". Möller argues that the technology is not widely known because "When COTS simulation tool vendors develop these products and add modules for more and more types of simulations there maybe an interest in keeping the users inside their own architecture and not interoperate with other types of architectures. They do not want you to connect to competitors, so they want you to stay with the same simulation tool and the simulation tool should cover more and more functionality, which leads to a customer lock situation".

Granowetter's explanation is more or less the same, he thinks that "The biggest reason is that there is no incentive for different vendors of different simulation packages to agree on a common interoperability standard". In defence this incentive was much clearer because one of the biggest customers for defence simulation, the US Department of Defence, is in the position to mandate the vendors to provide tools and models which use a certain standard for interoperability. According to Granowetter the commercial world is different, because there is no one big customer, and because it is probably less common that many different companies meet to get together because they want to join on distributed simulation. He argues that in industry it is always difficult to create an agreement between competitors, a typical example that Granowetter points to is the video game industry.

'Idealized' distributed simulation architecture for industry

The experts had different views regarding an 'idealized' distributed simulation architecture for the industrial domain. According to Granowetter "I think there are good arguments on both sides of the API standards versus wire standards debate. One benefit of a wire standard, is that it's easier for somebody to decide to implement some wire standard, a network protocol, than to use somebody else's piece of software. I think that many people are reluctant to say: I am going to rely on somebody else's software, a commercial RTI. Those people might prefer a wire level standard that they can implement themselves".

Möller believes that "The future of simulation will be very much in the spirit of grid computing, where you have just a big sea or big lake of resources of scenarios, data models, and without going anywhere just like sitting with the web browser you can put together those components, you can search and find components and put them together and evaluate scenarios with parts of models from different companies. I do not care where they are, I

just want to connect them using grid technology. So just like what happened with the Internet that suddenly you could just jump around between different places in the world without caring of where they are using it just as a big information resource, that in the same way with grid computing you would have one big world, in which you could click around collecting not only information but simulation resources as well. This needs to be combined with business models, for example you can use my airplane model for 15 minutes if you pay me $50".

Summary

Although the commercial HLA vendors are open for collaboration with industry, they are squarely targeting the defence domain. The main reason is that they hardly observe HLA projects in industry whereas, due to the US DoD mandate, defence has adopted HLA. The HLA vendors observe that COTS vendors are reluctant to cooperate with each other. They can understand this because they see no added value for them in agreeing on a common standard. Instead, they observe the tendency that COTS vendors try to lock their customers to their own package.

Concerning information hiding, a nice example is given by Möller, viz., simulation based acquisition. Both experts stated that reuse is common in defence. They also mentioned the difficulty of semantically aligning the models to be combined. Not surprisingly, they see HLA as a good tool, that hides low level technical details, is general and not locked to one specific area. As drawbacks of HLA they mention the steep learning curve and the need for users who are not only good programmers, but high level thinkers as well. As regards performance they admit that the DMSO RTI is not efficient enough, and that their products try to remedy this. Features that they would like to see added to the HLA standard are extendibility, fault tolerance, better security, the possibility to cooperate between different RTI implementations and better transparency in the sense that lower layers should remain visible through HLA.

In order to obtain greater acceptance of distributed simulation in industry HLA vendors mention middleware and the need for a protocol on the wire level which would enable package builders to write their own implementation rather than relying on RTI implementations built by others. Similar to the HLA developers, one expert envisages a new business model in which the modeller would combine parts obtained from the net which then would be put to work using grid computing.

3.6 Overview

Now that the results of both the questionnaire and the interview survey have been presented (Section 3.4.5 and 3.5.5), we arrive at the next phase in our research, the analysis of the data. According to the Delphi method, as discussed in Section 3.3 and 3.5.4, the final step is to give a coherent interpretation of the data. This will be done in the next chapter. Before doing so the qualitative analysis within the Delphi method prescribes a coding activity, cf. Section 3.5.4. Coding involves extracting from the data obtained the relevant topics that were raised, and dividing these into subtopics. This section is devoted to this activity. The resulting list forms the basis of the "theory" that we will develop in the next chapter.

As a starting point for our list we consider the questions on which our interview was based, cf. Section 3.5.1. We will use these questions as a preliminary list of topics. An analysis of the answers that we obtained on these questions will generate a list of subtopics, one for each question. During this categorisation process we will observe that some subtopics are reconsidered in the answers on other questions as well. This will lead to a regrouping of the subtopics in our final list, and also to the introduction of new topics. At the end of this section we will present the final list in a succinct form.

Question 1 of our interview was included to get an idea of the experience of the interviewed experts. This question was intended for validation purposes and the answers did not yield insights in the survey question we are studying here.

Question 2 focused on the difference between simulation in defence and in industry. We obtained several categories of answers. One category was related to the difference in the business model of defence and industry. This influences the goals of the models built in the various domains, and the experts observed differences in the structure and type of the applications. Also differences in software engineering aspects were mentioned. The experts also pointed at similarities, in the sense that in industry models are built with the typical attributes of the ones in defence and vice versa.

Question 3 dealt with reuse. Three subtopics emerge here. First of all there is the issue of full model reuse versus reusing parts of a model. Secondly, reuse has been discussed on a rather high level, viz. the level of distributed simulation in general. Issues here are aspects related to the complexity of the models, and aspects related to the trust in the models, which are influenced by software engineering issues around specification, verification and validation, cf. some of the answers on Question 2. The third subtopic deals with the problems encountered. There are low level technical problems (partly) solved by the HLA standard, and there is the big mainly unsolved problem of semantic interoperability.

Question 4 discussed information hiding. The answers to this question can be categorized into two subtopics, we obtained examples of information hiding, both in defence and in industry, and we observed that the opinions diverged regarding the extent to which information hiding is an issue in current simulation projects.

Question 5 was aimed at the difficulties and the benefits of applying the HLA standard. Benefits could be subdivided into three categories, HLA hides low level details, HLA is a standard, and HLA offers a general solution not tied to one specific application area. As

regards the difficulties we encountered two types of comments here. The first type is purely HLA related, the complexity of HLA, efficiency issues and the transparency of HLA. The second type has to do with answers given from a higher level viewpoint, namely distributed simulation in general. Notice that we observed the same subdivision when discussing Question 3. Issues on this higher level are the low level knowledge and the knowledge on general systems needed from the practitioners, and issues around cost and benefit. A final group of comments triggered by Question 5 focused on desirable additions to the HLA standard.

Question 6 discussed integration of HLA and COTS. Our impression that this hardly occurs in industrial projects was confirmed by almost all experts we interviewed. The first subtopic deals with the reasons behind this. We observed remarks related to the fact that HLA is applied mainly in defence, to the fact that HLA uses notions that are not available in COTS packages, viz. ownership. Further there were answers focusing on the inaccessibility of the lower levels of COTS packages, e.g., the simulation engine, on the fact that HLA is big, and on the difference between the high level COTS interface versus the low level HLA RTI calls. The second subtopic we observe in the answers to Question 6 deals with the pros and cons of middleware and adaptors. These issues were again raised in the answers to Question 9.

Question 7 was only posed to the COTS vendors. It dealt with the advantages and disadvantages of combining COTS models. We derive two subtopics here, on consisting of the opportunities in industry (collaborative design and development, heterogeneity, emulation). The other subtopic was already identified when discussing Question 3, viz. the differences in the business model of industry and defence.

Question 8 was posed to obtain information on support for HLA for industry. This question was only addressed at HLA vendors. The outcome here was univocal; the vendors only address the defence domain.

The last question was on an ‘idealized’ architecture for industrial distributed simulation. We observe the following subtopics here. First of all there is the idea of a solution applying Web service like notions leading to a new business model. Secondly there were answers discussing the feasibility of adaptors and middleware. Another subtopic centred around industry needing only part of the functionality that defence applies. Here two approaches surfaced – either define a subset of HLA to work with, or introduce a new lightweight standard with restricted functionality. The final subtopic focuses on the need for a high level COTS-like interface for this new architecture.

The above considerations are captured in Table 3.15. However, notice that some regrouping has taken place. One new topic has been introduced in the list, characteristics of projects favouring a distributed simulation approach. Under this new topic some old ones have been grouped. Furthermore subtopics which occurred in the above discussion more than once have been put in one place.

Table 3.15 Topics Derived from the Survey

Topics	Subtopics

The difference between defence and industry in applying distributed simulation	• Difference in the business model
	• Difference in the goal of the models built
	• Difference in the type and structure of the models built
	• Difference in software engineering aspects
	  ▪ Specification
	  ▪ Verification
	  ▪ Validation
	• Similarities for some models in the two domains

Characteristics of simulation projects making a distributed approach feasible	• Reuse
	  ▪ Full model reuse vs. partial reuse
	  ▪ Arguments on the level of distributed simulation
	  - Complexity of the models
	  - Trust in the models
	• Information hiding
	  ▪ Some examples
	  ▪ The extent to which information hiding plays a role
	• Other characteristics (collaboration, heterogeneity, emulation)

Difficulties and benefits of applying HLA	• Benefits
	  ▪ HLA hides low level details
	  ▪ HLA is a standard
	  ▪ HLA is a general solution
	• Difficulties
	  ▪ On the level of distributed simulation in general
	  - Semantic interoperability
	  - Low level knowledge needed from the practitioner
	  - Cost benefit issues
	  ▪ On the level of HLA
	  - Complexity
	  - Efficiency of HLA implementations
	  - Transparency
	• Desirable additions to HLA

Integrating HLA and COTS (HLA hardly applied in COTS related projects)	• HLA mainly applied in defence • HLA notions not covered in COTS packages • Closed feature of COTS packages • HLA is too big • High level COTS interface vs. low level HLA RTI calls
'Idealized' architecture	• Ideas from the Web • Feasibility of adaptors and middleware • Industry needs only restricted functionality • High level COTS like interface

4 DISTRIBUTED SIMULATION IN INDUSTRY: OVERVIEW AND PERSPECTIVES

4.1 Introduction

Based on the results of the survey (questionnaire and interview) and the literature study, in this section we would like to propose answers to the questions we have posed at the beginning of the previous chapter, namely:

- *Is it true that the HLA standard is hardly applied in industry?*
- ***Why** is the HLA standard hardly applied in industry?*
- ***How** can we solve this problem?*

Although the survey specifically focused on the applicability of HLA in industry, from the given answers of the experts it turns out that the problem should be studied in a more general context in the first place. Accordingly, first of all we should focus on the place in industry of distributed simulation in general, as formulated in the additional question:

- *What are characteristics of industrial-oriented simulation projects for which distributed simulation would provide an added value?*

In the first chapter of this thesis we presented some cases from industry where distributed simulation plays an essential role. There we have already pointed to some project characteristics, which suggested to apply distributed simulation in these cases. Thus, before focusing on investigating the position of HLA in industry we take a side road and try to find a comprehensive answer that includes a list of the relevant project characteristics.

Raising new questions during qualitative research is not unusual. This arises from the fundamentally interpretive feature of qualitative research (Creswell 2003). This means that the researchers construct an interpretation of the data by filtering them through a personal lens. This includes a description of individually collected data, analyzing data to find themes or categories, and finally building an interpretation or drawing conclusion about its meaning. Formulating an additional question, in our case, can be attributed to the unstructured open ended character of the interview, which gave the experts the opportunity to generate new insights, questions, and points of view. In our case, during the interview, as can be seen from the answers described in the previous section, most of the experts explicitly pointed to and discussed the possible application and usefulness of distributed simulation in industry.

The four questions stated above guide the structure of this chapter. Given the generality of the additional question, we start by discussing the characteristics of industrial projects that might lead the developers to apply distributed simulation rather than monolithic simulation. Then we turn to the specific questions concerning HLA. The second part of this chapter covers the first two questions, trying to figure out to which extent the HLA standard is applied in industry, and why people choose to apply or not to apply it. The interpretation and evaluation of the interview and questionnaire survey serves to validate our initial

hypothesis stating that HLA is rarely applied in the industry. Finally, considering the vision of experts regarding the future of distributed simulation in industry, we describe some perspectives that might stimulate industry to apply distributed simulation on a bigger scale.

4.2 Characteristics of Distributed Simulation Projects in Industry

Simulation model-based problem solving strategy is a multi-step approach that helps the simulation practitioner to carry out simulation projects for solving certain problems (see Chapter 2). Before building a simulation model a methodology should be identified for designing and developing this model. Based on the project requirements and the methodology chosen the simulation practitioners can then choose the most appropriate simulation tool (Tewoldeberhan*, et al.* 2002). As we stated at the beginning of the previous chapter, simulation practitioners in industry mainly apply COTS simulation packages. This statement has been validated and supported in the interview survey: all experts both from defence and industry agree. The main reason behind this choice is that these packages are tailored for industry, they hide low level programming details from the simulation practitioners, and provide a visual interface that allows fast and easy design and development. A comprehensive list of advantages of COTS simulation packages in contrast to general programming languages can be found in (Law and Kelton 2000).

COTS simulation packages have predefined building blocks or modules which help simulation practitioners to obtain high level solutions for creating simulation models in a short time period. This is relevant because industry, as McGregor states, is "less interested in technology and a lot more interested in getting a solution out as fast as possible". Currently, making decision in a short term period is a requirement posed by the industrial market. By providing advanced COTS simulation packages, the simulation vendors help the simulation practitioner to accelerate the pace of their problem solving strategy, and by this the simulation practitioner obtains a quicker response to the decision that needs to be made.

In many cases the best option is to create a monolithic simulation model designed and developed in a single appropriate COTS simulation package. Often this is more advantageous than designing and developing the same model in a distributed manner. The advantages include:

- *Performance.* A monolithic simulation model within one package performs faster than a collection of coupled models, because there is no need for explicit time synchronization, data exchange, etc., between the submodels.[31]
- *There is no need for distributed system knowledge.* The simulation practitioner is not hindered by notions from distributed systems and architecture, such as time management, data representation and exchange, etc.
- *Simulation practitioners can profit from the high transparency of the packages.* By providing predefined building blocks, the COTS simulation package vendors offer high level transparent solutions for designing and developing simulation models.

[31] Excluding cases where the distributed models are set up explicitly to use parallel simulation techniques for speeding up the simulation run

> Consequently the simulation practitioner does not need to go into details, all low level technical details are invisible for him/her. However, this is not the case with distributed simulation. Due to the fact that currently the COTS simulation packages do not provide high level building blocks for distributed simulation, as it stands now, in order to integrate distributed simulation models the simulation practitioner is forced to look at low level technical details, residing at a lower level than the standard predefined building blocks. Further, time synchronization, data representation and exchange, etc., should be solved on a level with which the regular simulation practitioner has no experience. And even when the practitioner has this knowledge and experience, working on such a low technical level will eliminate the economical benefits of applying COTS simulation packages, which consists of designing and developing simulation models in a short time period.

Taking into account the arguments above, we can state that: *considering the current market needs, if a problem can be solved by a monolithic simulation model created in a single COTS simulation package, and the problem owner does not explicitly ask for a distributed solution, the simulation practitioner should certainly choose for this monolithic solution in the selected COTS simulation package.* As Straßburger said: "if you can build a monolithic application which is still maintainable and reusable, meaning that you have modularity in it, then build it in a monolithic way, there is no point in building it in a distributed platform".

There are, however, simulation projects for which, given their characteristics, distributed solution seems more advantageous and straightforward. In this sense, based on the results of the survey and the literature study, we perceived four types of simulation project characteristics which demand distributed solution:

1. *Reusability*
2. *Heterogeneity*
3. *Collaborative Design and Development*
4. *Information Hiding*

4.2.1 Reusability

One of the main economic reasons for designing and developing the HLA standard was to *reuse* already existing defence simulation models for other projects (Kuhl, et al. 1999). To reuse existing simulation models in defence is indispensable. As Weatherly states: "if a model exists it might have been certified by some organizations of being credible, believable representation of some phenomenon so in order for your results to be credible you need to use these credible models. So reimplementation is often not an option". Even if it takes 4 to 5 months to adapt a defence simulation model to be HLA compliant so that it can be reused, it is not an option to reimplement, because the cost of reimplementation is much higher than the cost of adaptation.

Reusability is a central characteristic in general distributed application projects because it, as McIlroy's law states, "reduces cycle time and increases productivity and quality" of the project (McIlroy 1969). Due to its central role, we tried to elaborate in detail on this issue during the interviews, and we investigated to which extent and how models are reused in

practice. In this research we focus only on reuse of a Full Model since that is the only form of reuse that is specific to distributed simulation (see Chapter 2). The deep investigation of reusability provided us with a lot of material, and as a consequence we discuss it in more detail than the other characteristics. We base our analysis on reusability on the results of the interview survey that we have conducted. First we summarize the opinion of experts from defence and industry separately and then we compare and discuss their opinions.

For the defence simulation community we can conclude that:

- Most of them reused defence simulation models as Full Model reuse;
- Most respondents comment that applying reusability is not a simple process, because of the adaptations needed. They state explicitly, however, that reusability is a big issue in defence and the reimplementation of existing models is generally not an option;
- Most of them reused HLA compliant defence simulation models;
- The most difficult issue that they have encountered and that still needs to be solved is the semantic inconsistency when aligning data objects.

The experience and opinion about reusability of the experts from the industrial domain differs from defence. For industry we can conclude the following:

- Four out of ten industrial simulation experts applied the Full Model Reuse approach to reuse industrial simulation models;
- Four out of ten industrial simulation experts explicitly stated that reusability might become a big issue in industry;
- Only one of the interviewed industrial experts and only one of the COTS vendors who filled out the questionnaire (see Table 3.7) reported on reuse of HLA compliant industrial models;
- Most of them had difficulties with reusability and some of them pointed at semantic inconsistency as an important issue that needs to be solved.

The observations given above point out that there is a different opinion between the defence community and the industrial community on the necessity of distributed simulation. Comparing the opinion of the two communities related to the first observation we can conclude that Full Model reuse is more accepted in defence than in industry. According to the second observation, reusing full models plays a very important role in defence and reimplementation of models is not an option, while in the industry the majority of interviewed subjects do not explicitly highlight the benefit of reusability. Further, from the third observation we can conclude that while in the defence the HLA standard is a well established tool for reusing simulation models, in industry this is not the case. The only area where experts from defence and industry agree is the problem of the semantic inconsistency when trying to integrate simulation models. In order to identify the reasons behind the above differences, we elaborate next on the factors that cause these differences.

In the first place we consider the first two observations which are closely related to each other. While Full Model Reuse approach turns out to be an important issue in defence and

the defence community regularly applies it in their projects, it seems that in industry it has not gathered ground. We should thus try to find the reason behind this phenomenon. We need to analyze therefore the following question:

Why does not industry, in contrast to defence, regularly apply Full Model Reuse?

In order to answer this question we have to look at the differences between the industrial and defence simulation domain. From the survey we observe a main difference between industry and defence, namely the character of the business model that is the mechanism by which a business intends to generate revenue and profit.

Usually defence simulation models are more complex than industrial ones. As Morse argues "they require more technical depth" and as Weatherly states, practitioners in the defence have to "build a small number of very large simulations, and that causes those simulations to have to answer a lot of requirements and so they become very complicated and very expensive".

In contrast to defence, according to McGregor and Rabe, industry is more interested in getting a solution out as fast as possible. Therefore, their business model is mainly based on "throw away" models which are very project specific and the users usually do not maintain these models later on. McGregor points to the fact that "you often see a material handling company doing a minimalist simulation just to satisfy the client. Fortunately, in most cases the material handling companies themselves will see the advantages of doing proper simulation, but even then if you can take half the time to get the same results, take half of the time. If you do not need to simulate every single, tiny movement, because that will exert a 2% influence on the results, and we know that the real production is not completely defined by the client yet, then they will go for the shortest possible solution". Some experts both from defence and industry observe the same phenomenon and claim that validation in industry has not such an impact as in defence. Defence spends a big amount of money on verification, validation and accreditation (VV&A) of the simulation models. Simulation models that pass this VV&A procedure can be considered as trusted models, and thus, reused without any problem.

Related to this Paul and Taylor mention that model reuse is dependent on trust (Paul and Taylor 2002). If a simulation practitioner cannot trust a model, he will certainly not use it. From their point of view reusing the model and building trust around a model seems to be a longer process than just simply rebuilding the model from the beginning. In industry models are not generally trusted due to the lack of VV&A, but for defence, as most of the experts state, VV&A is an important issue and rebuilding a model which already passed the VV&A procedure is not an option.

Summarizing the arguments above, we identify the following issues related to the business model that entail the fact that the Full Model reuse approach is rarely applied in industry:

- *The scope and funding of industrial simulation models is limited.* Most of the simulation models are very project oriented and are developed as "throw away" models created with a limited budget. The designers of these models are not providing solutions for reuse such as appropriate documentation or specification, simply because of the limited time and budget. Their primary aim is to finish a project as fast as possible and to get an *acceptable* result as soon as possible. If somebody wants to reuse such a model, due to the incomplete specification, he

must delve deeply into the code in order to understand it, which might be a longer process then redeveloping everything from the scratch.

- *Trust of the model.* Assume that somebody provides a model which is properly documented and based on the specification it seems to be an appropriate solution for our purpose. We will still hesitate about reusing it when there is no guaranty provided that this model has been properly verified and validated.

Limited scope and funding, and missing trust are reasons that are mainly related to the business model. This concludes our discussion regarding the first two observations. To some extent this also explains the third observation, namely the fact that the HLA standard is mainly applied and considered in defence and rarely in industry. Due to the fact that defence has the budget and the time to design simulation models that are "prepared" for reuse and they are properly validated, they have a better chance for Full Model reuse. Reusability has such a big scope in defence that "preparing for reuse" is forced by mandates, like "you must use HLA", and this is plausible when reimplementation is not an option. While the experts from defence claimed that in defence there is regular HLA model reuse, as Table 3.7 illustrates there is almost no reusability of HLA compliant models in industry. This is an illustration of the phenomenon as Rabe states, "very bad use of HLA today" in industry.

As stated before, the only point where experts from the two communities agree is the unsolved problem of semantic inconsistency when aligning data objects of reused models, a problem that not only plays a role in the simulation community but also in the whole software engineering area. This is the biggest problem that the defence community is confronted with when they reuse a model, and most of the time it can be solved only through regular conventional meetings. The HLA standard does not cover this issue and currently there is no standard approach to solve this problem.

In spite of the problems and differences stated above, there are some projects in the commercial area in which Full Model reuse is applied. For example Lendermann and Malcolm point to supply chains in the semiconductor industry, in which they simulated the material flow from one factory to another one and they wanted to reuse already existing factory models. As Lendermann argues "you can put everything into one model but that is totally unpractical because you also need to maintain these models and the people are using them for all kinds of exercises". Rabe also points to the benefits of reusability when applying distributed simulation. The example he gives is the simulation of production systems in the context of a supply chain. Since some models of the production system were already available in one commercial tool and it was more advantageous to model the supply chain in another tool, distributed simulation helped them to reuse the already existing models. Another example is given by Straßburger who was involved in a project within DaimlerChrysler where they designed and developed a simulation model of an automaker digital factory. The objective of a digital factory is to have a detailed digital representation of the whole real factory, which covers all relevant causal relationships. Some of the processes within the real automaker factory, like the paint shop model, had already been designed and developed independently and analyzed separately for local purposes. In order to achieve their objective they have integrated and reused the independently designed and developed simulation models (Straßburger, et al. 2003).

Summarizing, a significant advantage of distributed simulation above monolithic simulation can be observed in projects where simulations practitioners intend to *reuse already existing models in combination with other models.* In our discussion with Fujimoto he stated that this is one of the most important reasons that can convince simulation practitioners to apply distributed simulation.

4.2.2 Heterogeneity

The complexity of systems that need to be designed and researched is gradually increasing. It might happen that the problem that needs to be analyzed requires a simulation model that cannot be designed and developed in one single COTS simulation package because the requirements of designing and developing such a model is beyond the scope of any single COTS package. An example given by Heinicke concerns a factory model for simulating chemical reactions. The eM-Plant package provides useful tools for creating general factory models, however, it does not provide features for modelling the effects for chemical reactions. For the simulation of chemical operations there are specific tools available. Consequently, it makes sense to design the general factory part of the model in eM-Plant and the chemical reaction part in a tool that supports this operation and then integrate them. So, the first situation in which a distributed simulation is a natural technique is the case of *coupling heterogeneous models.* A second situation where distributed simulation is a useful technique is in *coupling a simulation with real equipment*.

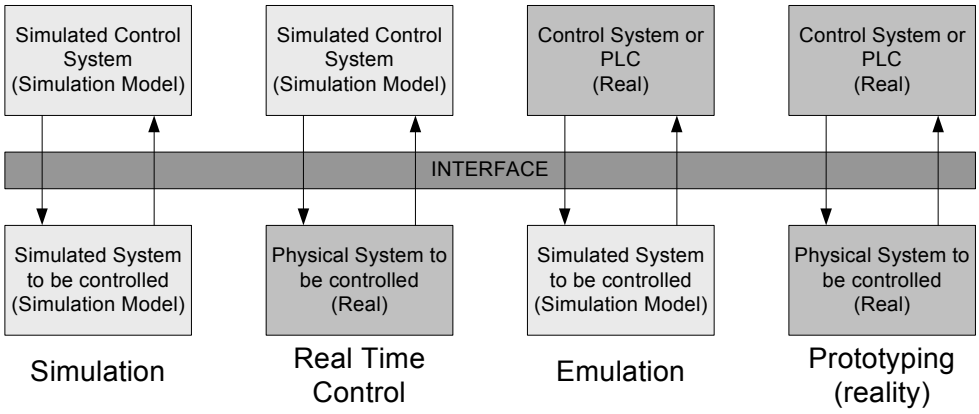


Figure 4.1 The Four Possible Ways of Testing Systems (based on (Auinger, *et al.* 1999, pg. 799))

One of the aims of simulation is to provide performance indicators about real systems before investing in their development or adaptation (Zeigler, *et al.* 2000). Consider the design of a complex technical system with complex control components. In order to fulfil their role, the equipment and the control system should collaborate and communicate in a well defined and well designed manner. The development of a complex real system which is controlled by a separate control system may include one or more of the following phases

(Figure 4.1), which aim to test the system during different design stages (Auinger, *et al.* 1999):

- *Full simulation*: includes the simulation of both the equipment and the control system;
- *Real-time control*: uses real equipment and simulates the control system;
- *Emulation*: simulates the equipment and uses the real control system;
- *Prototyping*: involves tests with real equipment and real control system.

Real-time control and emulation is a quite rapidly growing field (Rohrer and McGregor 2002). McGregor and Heinicke, for example, mention some instances. Emulation and real-time control have the advantage that they can be carried out in a cheaper way than full prototyping, and stay closer to reality and are therefore less time consuming than full simulation.

We have to take the fact into account that the models and the real components are: (1) inherently distributed and so they are placed on physically distinct places and (2) for several experiments one might integrate real equipment instead of a simulation submodel or vice versa in the experimental model. Obviously distributed simulation is a natural method in this case.

4.2.3 Collaborative Design and Development

In the survey there was not much response on this issue. The only example mentioning collaboration was given by McGregor. The need for collaboration was the main reason for choosing distributed simulation instead of building a monolithic model in the US Postal Service project mentioned by him. Although only one organization was involved, Brooks Automation, the project was very complex, many template models had to be produced for different distribution centres and because those models were so big it became preferable to build them separately by different teams and then join the different parts together using a communication protocol. In this case the project characteristic that leads practitioners to use distributed simulation was *collaborative design and development* which helped the team to speed up and control the design and developing process.

In Chapter 2 we presented four important constraints (expertise, time, money and competition) why organizations or individuals might collaborate. In the example given by McGregor the time was the only constraint that led them to choose collaborative design and development.

4.2.4 Information Hiding

During the interview we explicitly asked the experts about their experience and opinion regarding information hiding within simulation models. The opinions of defence simulation experts regarding this topic diverge. There are experts who state that information hiding is not a big issue in defence, while others claim that information hiding is essential. Analyzing the data, we have recognized that experts who state that information hiding is not a big issue refer to the ownership of the models that are generated. An example is given by Morse who argues that “the military owns all of these models and it

pays the contractors to develop them and although the contractors maintain them, in fact the government owns them. So, if there were problems connecting to simulations that are related to internal data structures or for example the algorithm that a particular simulation is using, the government is in a position to go to the contractors and say you have to tell how it works, but contractors are not in a position generally to refuse to do that. So it is not so much an issue".

Experts from defence who claim that information hiding is an important issue mainly refer to simulation projects which cross boundaries, in which organizations are not willing to grant ownership of their models to other organizations and then information hiding suddenly becomes a sensitive problem to solve. One example, mentioned by Huiskamp, is related to a classified project, which involved different nations. The models designed by the different nations included sensitive information, so they tried to achieve a very restricted information exchange between the simulation models. Furthermore as we have seen from the example given by Möller, Simulation-Based Acquisition is a typical example in case. For example, when defence intends to buy airplanes, interorganisational collaboration is established between defence and the airplane factory. Before buying the airplanes their behaviour should be evaluated. Therefore, besides the specification on paper, defence also requests a model of the airplane that can be tested in their simulation test beds. The airplane factories do not disclose the whole internal structure of the model of their product, they just provide a restricted interface which is sufficient to investigate the behaviour of the airplane during operation.

The industrial simulation experts, as we have perceived, have not really considered the problem of information hiding in their simulation projects. Only Van Houten gives an example. He had the chance to participate in a distributed supply chain simulation project. One of the main issues here was the need to hide sensitive information from the other participants.

Although most of the industrial simulation practitioners who applied distributed simulation did not have to focus on information hiding, they agreed that distributed simulation is an appropriate technique to provide a solution for this problem. McGregor, who represents a COTS vendor, states for example that distributed simulation might be a great solution for information hiding if they would collaborate with a consultancy company. In such a case the only thing that they need to define would be an appropriate interface that allows the internal simulation code to remain hidden. Lendermann points out that information hiding is an important issue in the semiconductor industry as well when two manufacturers cooperate with each other. Due to the fact that they might be competitors they certainly will want to hide internal information.

So information hiding within simulation models refers to protecting two issues:

- intellectual property (e.g., special algorithms);
- sensitive internal data (e.g., company or client information).

Fujimoto recognizes that distributed simulation is a good approach for information hiding within simulation models, but he states also that "within industry unfortunately we are not yet there". Möller believes that buying things, like airplanes by defence, based on simulation will gather ground also in industry, that "it is happening right now in defence,

but in the future it will occur to a bigger extent in the civilian area" and then information hiding will gather ground in industrial simulation.

Concluding this section, we state, that another project characteristic that could lead practitioners to apply distributed simulation is the necessity to *hide sensitive information within simulation models* in collaborative interorganisational simulation projects.

4.2.5 Concluding Remarks

According to the discussion presented above, the answers to the additional question we posed in the introductory part of this section, namely:

> *What are characteristics of industrial oriented simulation projects for which distributed simulation would provide an added value?*

distributed simulation in industry induces added value in situations in which we need to:

- *reuse already existing models in combination with other models;*
- *couple heterogeneous models or real equipment;*
- *collaboratively and separately design and develop a complex model ;*
- *hide sensitive information within simulation models.*

As Table 3.2 and Table 3.3 indicated, several projects exist where both homogeneous and heterogeneous simulation models are coupled for interoperability purposes. In our questionnaire survey more then 50% of the COTS simulation package vendors recognized that either they or their customers have already carried out distributed simulation projects successfully. In each of the projects that they have specified, one or more of the four project characteristics given above can be observed.

4.3 The HLA Standard in Industry

In the previous section we indicated some project characteristics indicating when it is more advantageous to use distributed simulation than building a monolithic model. The industrial community recognizes in some measure the added value of distributed simulation for projects with these characteristics, since, as the questionnaire survey indicates, there are successful projects carried out in industry in which distributed simulation is applied. However, in these projects various homespun distributed solutions for integrating simulation models (Table 3.2 and Table 3.3) are applied and rarely use is made of the HLA standard. In this section we aim to analyze why HLA did not gather ground in industry.

As Table 3.13 illustrated, in only 2 out of 19 industrial-oriented distributed simulation projects the HLA standard is applied. This number seems to support our hypothesis that the HLA standard is hardly applied in the industrial domain. However, in spite of the fact that the questionnaire survey seems to strengthen our hypothesis we cannot confidently state that the hypothesis is validated because of the inconsistencies we have discovered during analysis of the answers in the questionnaires (see Section 3.4.5). The problem stems from the fact that the participants interpret the notion "support for the HLA standard" in

different ways, which makes it difficult to evaluate, whether the HLA standard is in fact applied and supported.

In order to validate the hypothesis we have explicitly incorporated the statement: "The HLA standard is rarely used for integrating simulation models designed and developed in COTS simulation packages" in the interview and asked the experts' opinion on it. All experts agreed with the statement. We believe that this together with the result of the questionnaire is convincing enough to justify the hypothesis. This together with the fact that the industrial simulation community mostly applies COTS simulation packages (also validated by the interview) affirmatively answers the first question of the previous chapter, namely:

> *Is it true that HLA is hardly applied in industry?*

The few industrial simulation projects in which the HLA standard has been considered are big, complex simulation projects such as the ones mentioned by Möller (e.g., the Japanese electronic manufacturing industry, ship manufacturing for the oil industry) or they are close to defence or government type of applications like the ones from the NASA space industry. In these industrial simulation projects the simulation models required technical depth and mostly general purpose programming languages have been applied for designing and developing them.

Next we aim to identify the reasons behind the phenomenon that HLA is hardly applied in industry. We believe that this will lead us to factors that can make distributed simulation more attractive for the industrial simulation community. So we try to find the answer to the following question:

> ***Why** is the HLA standard hardly applied in industry for integrating simulation models developed in COTS simulation packages?*

Analyzing the data we collected we can observe that there are three main groups of arguments related to the perceived rare use of the HLA standard in industry. In the first place, the arguments are related to distributed simulation in general, secondly to HLA itself, and finally the relation between HLA and COTS packages exerts influence on the application of HLA in industry.

These arguments are deduced from the survey and literature study and mainly point to the perceived weaknesses of distributed simulation, especially HLA. It does not mean, however, that distributed simulation has only disadvantages and no benefits. The value of distributed simulation is highlighted for example by the four simulation project characteristics covered in the previous section. Next let us discuss the three groups of arguments mentioned above in more detail.

4.3.1 Arguments Related to Distributed Simulation in Industry

According to Rabe the observation that the HLA standard is rarely applied in industry is already a derivation, and follows from the fact that distributed simulation in general is rarely applied in industry. Opinions in literature and other results from the interview support this observation. We should thus first analyze why distributed simulation in general is rarely considered by the industrial simulation practitioners.

In the questionnaire the COTS simulation package vendors explicitly mention only 19 successful distributed simulation projects. Unfortunately we do not know the ratio of distributed simulation projects versus monolithic projects, the questionnaire did not contain explicit questions regarding this issue and the vendors did not provide additional information. So the number 19 can serve only as an indication.

As we perceive and interpret from the results of the survey, the reasons behind the unpopularity of distributed simulation in industry are primarily market related. The *market reason* has to do with the *cost benefit ratio*[32]. Experts argue that currently, solving a problem using distributed simulation leads to a cost which is too high compared to the benefits that they can gain from it. According to McGregor "the amount of asset that we would have to go to, to map our objects to their objects would probably outweigh any financial benefit that we would get". Although Heinicke recognizes some benefits of distributed simulation, he claims that as it stands now "we see only more work".

Next to the benefits we already covered in the previous section, we aim to analyze what the costs of distributed simulation are compared to the traditional models. So the next question is:

> *Why does industry perceive distributed simulation as more costly than traditional monolithic simulation?*

In other words, we would like to discuss, why the advantages of distributed simulation discussed in the previous subsection are not strong enough to convince simulation practitioners to apply this approach more often. There are two aspects here. On the one hand the added value of distributed simulation can indeed be too small compared to the costs. On the other hand, it can be the case that the benefits are invisible for some practitioners and end users.

Distributed simulation, additionally to monolithic simulation, requires a distributed design and development of the models and a solution, e.g., a tool, for interoperation. We analyze the additional costs of distributed simulation along three dimensions:

- Time – The time required for designing and developing models in a distributed way, and the time required for applying (or even building) interoperability solutions;
- Monetary cost – The time spent to design and develop models entails monetary costs. Additionally, there are direct monetary costs involved, i.e., the purchase price of an existing interoperation tool or the monetary costs of developing such a solution;
- Quality – It might be the case that the added value of distributed simulation is countered by shortcomings on the quality of the resulting model.

Time

Distributed modelling in general is more complex than building a monolithic model, it involves additional concepts, such as operating the model at run time, synchronizing time,

[32] The cost benefit ratio is defined as "the net present value of an investment divided by the investment's initial cost", from http://www.trading-glossary.com

representing the data to be exchanged, exchanging the data, and managing ownership of an object, concepts which the regular modeller is not used to. In a monolithic style the simulation practitioner either does not need to take care of all these issues or it is much easier. These are high level concepts from distributed simulation. However, when applying a tool, like HLA, the user also has to get acquainted with the way the tool implements these concepts. Möller states that applying the HLA standard requires high level thinking people because they need a detailed understanding of many concepts such as modelling, information models and architectures. This observation applies for distributed simulation as well.

Although our preliminary observation was that good programming knowledge can mainly solve most of the problems related to distributed simulation the interview with experts clarified that there is more needed than just good programming knowledge, because, as Möller argues, even if people "know the programming level good enough, they need to understand system architectures and how to create models of the world around them". Most of the experts, both from defence and industry, claim that, currently the simulation practitioners who daily design and develop simulation models using high level COTS simulation packages do not possess the knowledge that is necessary for creating distributed simulation.

Distributed computing in general is too complex for the simulation practitioners who are not used to it and therefore it requires lot of effort needed to learn to deal with it. Time spent to learn to deal with this complexity entails hidden costs which might be too high related to the expected benefits.

Monetary cost

One approach to avoid the costs of learning low level concepts needed for distributed solutions, is to provide tools for the user, which might hide the low level details. The HLA standard is such an approach, and as discussed in Chapter 2, it is the most advanced one. Why HLA, in spite of its aim and maturity, is not frequently applied in industry is the topic of the next section. Here we aim to discuss what the costs of applying such a solution are.

The monetary costs of a distributed solution do not seem too high to hinder people to buy and apply it. Only two experts from the research area, Ryde and Taylor, argue that the scarce use of HLA in industry might be influenced by the purchase price of the architecture. Ryde argues that "I do not believe that HLA has a place, purely on the cost basis. In terms of functionality as well, I do not think so that all functionality is required in that particular domain. The RTI costs too much and the customers are willing to pay at most 10% for an additional distributed simulation feature in a package - the vendors do not want to take steps in this direction". The purchase price of the architecture might be too high for academic researchers and it certainly adds extra cost to the cost benefit ratio. However, as it stands now, we see that there are more important issues and other types of costs than the price of the architecture. In spite of the fact that the DMSO RTI is a free solution that exists since 1996 almost none of the COTS simulation package vendors applies it as an additional feature for their customers. This indicates that the purchase price is not the main reason for not applying distributed solutions. Although purchase price might be an issue if we would consider a commercially available RTI, Rabe and Lendermann both think that even then the cost of these architectures currently is not an issue, because a 3000 euro RTI

license is not so much in comparison with some of the simulation packages, which cost 30.000 euro.

If the price is not the issue, the cost may be hidden in the effort needed to apply the existing distributed architectures. How much additional effort is needed for integration of the concepts of a specific distributed solution into a COTS simulation package, and further, how well are these concepts hidden from the simulation practitioners? Currently, there is no compact tool that can be easily used to integrate simulation models from arbitrary packages, without any adjustment to the packages or the models.

Quality

Next to the monetary costs and the costs related to time consuming, *the quality of the solution* might hold people back to apply distributed solutions. Although distributed simulation, as we presented in Section 4.2, can provide added value for some projects comparing to the traditional monolithic solution, according to some experts the end solution can be often qualitatively deficient. One of these qualitative aspects is performance. Regarding performance Lendermann thinks that distributed simulation in general should be improved. Personally we see that distributed simulation theory has already brought about efficient algorithms, some of them being discussed in (Fujimoto 2000), however they need to be implemented. Besides a performance price we also mention that verification and validation of distributed systems is more challenging than verification and validation of monolithic ones (Page*, et al.* 1997), and sometimes it is not even solved. In this way developers would loose quality when choosing a solution based on distributed simulation.

Furthermore, as it stands now, the distributed simulation community is confronted with the problem of the semantic inconsistencies that might occur during interoperation. The unsolved problem of *semantic interoperability* when aligning the data models is the main reason that hinders developers to apply distributed simulation. According to Fujimoto "semantic interoperability is a big issue. This is a hard problem, and probably much harder than the problem for which HLA was designed and developed". Semantic interoperability is not only an issue for the simulation community but for the whole system engineering community as well. For example, for the web community, which is quite large, tools for semantic interoperability could help eliminate the limitations of keyword based matching techniques. Users searching the web based on a keyword often experience the problem that either they get no results back or they get too many irrelevant results. For instance, a problem is that words can have synonyms, having the same meaning, or one word might have multiple meanings. However, if there would be a more formal specification of the semantics behind languages used to describe web pages, then a user could specify a query in the terminology that is most convenient, and be assured that correct results are returned, regardless of how the data is represented in the sources. So not only the distributed simulation community suffers from this problem.

4.3.2 Arguments Related to the HLA Standard in Industry

In order to solve and to hide the complexity of distributed simulation, the simulation defence community designed and developed the HLA standard. One of the most important benefits of HLA, as some of the experts claimed in the interview, is that it solves the low

level technical distribution details for the simulation practitioner. Analyzing the questionnaire and interview results we identify three possible reasons that can explain why industry does not apply the HLA standard. These three reasons are:

- *The HLA standard is too complex for industry*
- *The HLA standard is not transparent enough for industry*
- *The current implementations of the HLA standard, especially the DMSO RTI, is too inefficient for industry*

The HLA standard is too complex for industry

Complexity especially refers to understanding the concepts behind this standard and using it for distributed projects. Although the HLA standard was intended to hide the inherent complexity of distributed simulation, it seems that it did not succeed to hide it completely because, as most of the experts, both from defence and industry, in the interview argue, the learning curve of applying this approach is very steep. Some of the experts from industry, such as Malcolm, state that it is quite difficult to overcome this learning curve even when just trying to build some very simple examples. Ryde argues, that "too much effort is required on behalf of simulation practitioner, who may not be a technical person". According to Straßburger "the most difficult was getting used to the HLA style of thinking and programming. The complexity of the HLA interface has been an issue and developing low level task like building your attribute updates, and collecting the bytes and bites".

The HLA standard implements a complex approach, and as Möller observes, to be useful it needs high level thinking people who have a proper understanding of the concepts of modelling, information models and architectures. HLA, as it stands now, does not really have a place in industry where as Heinicke states "there are no experienced users today". This is also affirmed by Rabe who claims that "I do not expect that the native application of the HLA has any chance in this area".

Regarding the complexity of the structure covered by HLA, Heinicke argues that "the complete HLA structure is a large structure, so normally simulation tools only need small part of it." McGregor also observes the big structure of HLA and he claims that "it seems like an interesting idea but probably more attractive if you are in a large collaborative environment, possibly such as supply chain, possibly such as military". Ryde points as well to the complex structure of HLA, he claims that "HLA incorporates a lot of functionality that may not be useful to everybody". The industry, as most of the experts argue, needs much less functionality than what is currently provided by the HLA standard. Weatherly argues that "something that you can do is changing the HLA specification, I believe. You could say, here is a much more limited definition of interoperability and that might make it easier for the simulation developers". Fujimoto has the same opinion, he thinks that "for particular branches of industry their requirements are much narrower so probably a subset of HLA suits better, optimized for their particular purpose". According to Malcolm there is a need "to find out what kinds of functionality do they need and use only those".

The HLA standard is a good approach that intends to solve problems for "all kinds of" distributed simulations that might come into people's mind. In order to cover so many issues, it has become quite complex. This feature is also mentioned by Weatherly who

argues that HLA provides all functionality that one can imagine for distributed simulation and that this is maybe the weakness of HLA because it is does so much. Some experts claim that if the functionality which is currently not useful could be hidden or eliminated in some way, then the architecture would become clearer, it would certainly flatten the learning curve and stimulate its applicability in industry.

The HLA standard is not transparent enough for the industry

Some of the experts complain that HLA is not a transparent solution. Before starting the discussion around transparency we like to mention Fujimoto's comment on this point, namely "I think what industry is saying is that it is too complex". So basically this reason relates to the previous point.

Regarding the transparency of the architecture an interesting difference in viewpoint between experts from defence and from industry surfaced. Simulation practitioners from the defence community complain that the HLA standard is not transparent in the sense that they cannot perceive what is happening in the RTI implementation. In order to solve this problem MÄK designed a tool called RTI Spy which sits on the top of the RTI functionality. This allows the modeller to analyze and debug information, to help federation developers to find out what is going on, where the connectivity problems are, who is not making a right RTI call, etc. In contrast to modellers from defence, simulation practitioners from industry complain that HLA is not transparent in the sense that it offers too many functions which are not relevant for them. Based on this observation we have concluded that it seems that, while for defence transparency means *visibility* of the underlying layers that is, for the industry it means *invisibility*.

In information technology the concept of transparency has a meaning of *invisible* or *undetectable*. Software programs or procedures that are said to be transparent are typically those that the user is, or could be, unaware of. Transparency is considered to be an especially desirable feature in situations where users who are not particularly technically inclined would tend to be confused by seeing or having to interact directly with low level programming components. The domain name system (DNS), for example, operates in a transparent manner, resolving authorized domain names into Internet Protocol (IP) addresses, all without the user's knowledge. The need for transparency is recognized in distributed systems (Tanenbaum and Van Steen 2002). For instance, transparency is related to the way distributiveness is hidden from the user (Tanenbaum 1995). Considering the transparency of a distributed system, Tanenbaum gives an illustrative example of a distributed system which is supposed to appear to the user as a traditional uniprocessor timesharing system. "What happens if a programmer knows that his distributed system has 1000 CPUs and he wants to use a substantial fraction of these for a chess program that evaluates boards in parallel? The theoretical answer is that the compiler, runtime system, and operating system together should be able to figure out how to take advantages of this potential parallelism without the programmer even knowing it." (Tanenbaum 1995, pg. 24). Such a feature would be desirable for simulation practitioners, which are experts in COTS simulation packages but not necessarily familiar with low level technical details.

So basically what industry needs is a transparent or invisible architecture that helps simulation practitioners to integrate different simulation models without requiring distributed system knowledge.

This invisibility, as some of the experts argued, can be achieved by means of *middleware* or *adaptors*. Adaptors can hide low level technical details from the simulation practitioner and for handling distributiveness they provide high level building blocks on the same level as the traditional building blocks for creating monolithic simulation models. Advanced HLA adaptors or middleware for COTS simulation packages, that we observed, are designed and developed for Arena, Simple++ and Taylor ED in the IMS MISSION project (McLean and Riddick 2000), (Rabe*, et al.* 2001), for SLX and Quest at Magdeburg University (Straßburger 2001), and for Arena at Genoa University (Revetria*, et al.* 2003). Although these are first steps towards hiding HLA details from the end users, many people express the opinion that transparency is still a problem that is not satisfactorily solved by the adaptors or middleware available today. According to Rabe "the systems available today for distributed simulation are all in a prototypic status. I think our adaptor, even if I would say that is near a product, I would still call it somehow prototypic". Taylor claims that the current adaptors or middleware are "workable solutions", which prove the concept but they are not end results. Heinicke similarly believes that even when having adaptors distributiveness is not completely invisible as "you have to write your own connection using HLA between two different simulation programs and this seems to be really hard coded. So the soft coding, the easy connection is not what I currently see". According to him currently the only people who can easily deal with integrating COTS simulation models through adaptors to HLA are those people who designed and developed the adaptors. Taylor agrees that we need clearer standards for interfacing COTS simulation packages.

As it stands now adaptors are improvements aiming at achieving higher transparency of the HLA standard. As such, they decrease the complexity and increase the applicability of HLA in industry.

The current implementations of the HLA standard, especially the DMSO RTI, is too inefficient for industry

Another issue that hampers the acceptance of the HLA standard in industry is the perceived performance problems of the currently available HLA RTI implementations. Due to the fact that almost all experts who applied the HLA standard made use of the DMSO RTI, the complaints discussed here mainly refer to the performance of this version.

The opinion of the experts in defence regarding the performance of HLA implementations deviates. On the one hand there is a group who is in general satisfied with the performance of the HLA implementations, while on the other hand there is another group which complains about performance and thinks that it should certainly be improved. For the experts from industry the opinion is more univocal, all experts who have experience with the HLA standard complain about the performance of the DMSO RTI.

Not surprisingly, the HLA designers and developers are the ones that are the most satisfied with the performance of the HLA implementations. They state that early versions of the RTI were slow but that newer versions perform satisfactory. According to Morse "people who complain make a lot more noise than people who do not complain". She claims that "using the DMSO RTI we have supported simulations with tens of thousands of entities". The RTI that DMSO produced, as Morse argues, was designed and developed to be a general purpose architecture, it was supposed to implement all services possible. DMSO's vision on how the market would evolve was that at some point the full RTI will be taken

over by the commercial market and the commercial market would produce RTI's for specific user domains with a performance satisfactory for that community. Actually this is what happened with the RTI, which was produced by MÄK Technologies whose primary aim was to serve the real-time defence community. The differences between the performance of the DMSO RTI and the MÄK RTI is described in (Burks, *et al.* 2002). Both Weatherly and Morse argue that even sometimes people blame the performance of the RTI when this bad performance might arise from inefficient design and development of the simulation models or inefficiencies of the network. According to Weatherly these complaints stem from the fact that "RTI is hiding the network from you, so if you do not have enough network capability, then it appears that the RTI is slow, and you say that HLA is slow". Although Weatherly basically agrees with Morse, he points to some cases where he can imagine that the DMSO RTI imposes latencies which slow down the whole simulation. This situation can occur for example, as he mentions, if somebody is doing visualization and frequent refresh of the display is required.

The HLA vendors have more or less the same opinion as the HLA designers and developers. Granowetter claims that "the early implementation of the DMSO RTI definitely had some performance problem" but as he states "the DMSO RTI now is suitable for large federations". Both Granowetter and Möller claim that their HLA implementation, pitch RTI and respectively MÄK RTI, performs better than the DMSO RTI.

Not only the vendors but also other experts, like Huiskamp, declare that the commercial RTI's perform better than the one from DMSO. According to Huiskamp "it still should be improved. I do not see a difference between the different vendors, but the DMSO RTI had a pretty poor performance and very little control over the performance". He is not completely satisfied however, he thinks that commercial RTI's could still be improved as well. At TNO-FEL[33] they have designed a shared memory version of RTI, they have full control over it, and it performs much better than the DMSO RTI. Another defence simulation practitioner, Severinghaus, states that he had experience with training related distributed simulation where the RTI performed satisfactory but also applications where there were problems regarding the performance of the RTI.

The industrial simulation practitioners seem to have more negative experiences with the performance of the DMSO RTI. Van Houten perceived an unsatisfactory performance when he coupled two Arena simulation models. He ascribes the problem partially to the performance of the DMSO RTI, and partially to the huge overhead due to time synchronization. He mentioned that changing to another Arena version helped. Rabe also complains about the performance of the RTI, he claims that it is very difficult to get an acceptable performance when coupling COTS simulation models. Malcolm, whose activity at SimTech is to research on the performance of distributed simulation in general, has explicitly analyzed the performance of the DMSO RTI. His conclusion was that "there are bottlenecks in terms of long delay when a federate intends to join a federation and there is also a lot of overhead in terms of broadcasting messages or point to point message sending. So you get the feeling that the implementation is inefficient". In the questionnaire survey some COTS simulation package vendors complain about the performance as well. Krahl

[33] http://www.tno.nl/instit/fel/

argues that they have done distributed simulation using Extend for aerospace companies simulating large scale communication system, and as he states "HLA was rejected for these projects because of performance issues".

In view of the diverging opinions of experts from the different groups it is difficult to draw a clear conclusion regarding performance. On the other hand, focusing on industry, the domain we are analyzing, it seems that there is a need for an approach which performs better than the DMSO RTI. Some experts think that it does not matter how large the specification is and how much functionality are provided because this should not influence performance. Fujimoto's opinion diverges here, claiming that "you can gain efficiency by trimming the specification". At GeorgiaTech experiments have been conducted with a research version of RTI, called RTI-Kit[34], which had less functionality than the DMSO RTI and performed better. As Fujimoto states "DMSO argues that it does not matter how much functionality one is using, the performance will remain the same. My opinion diverges with those. If you use less functionality you will get a better performance". Huiskamp has the same opinion as Fujimoto, he thinks that there is functionality that not everybody uses and needs which costs a lot of performance. There is need for a simpler solution than the HLA standard, with less functionality, only the one that is currently needed for industry. The expectation is that the implementation of such a solution will perform better than the RTI's that are currently available.

4.3.3 Arguments Related to the Relation between HLA and COTS

The arguments discussed above were related to features of the distributed solutions. In this section we focus on COTS simulation packages and the way and the extent to which they support HLA. Analyzing the data collected both in the questionnaire and the interview survey we can conclude that:

> *Hardly any COTS simulation package allows appropriate creation of HLA compliant simulation models.*

According to the evaluation of the questionnaire survey presented in the previous chapter, we concluded that, although there seems to be active interest in HLA by about half of the COTS vendors, this interest is rather tentative and aiming at low level solutions.

Regarding "HLA compliancy" several vendors state that their package supports it. They claim that their package is HLA compliant, and that this can be achieved by the low level communication protocols provided. To some extent they are right, because if you use a low level communication protocol, like WinSock, you can connect to anything you want.

What in fact do we mean by HLA compliancy at a higher level? In order to clarify this notion we have posed two questions to distributed simulation experts:

- *When would you say that a COTS simulation package is fully HLA compliant?*
- *What features do COTS simulation package need to be fully HLA compliant?*

[34] http://www.cc.gatech.edu/computing/pads/tech-highperf-rti.html

The answers to the questions diverge in some points. In order to understand this, we first of all have to clarify the notion of HLA compliancy: Are we talking about HLA compliancy of a package or HLA compliancy of a simulation model?

If we are talking about HLA compliancy of a package, an HLA compliant COTS simulation package, according to Huiskamp, should "actually have an HLA interface and you have a well-defined FOM or SOM that you support". Granowetter has more or less the same opinion, he states that "the simulation package should have built in HLA interfaces". Weatherly states that "in order to be HLA compliant you would need to supply to the COTS simulation package a mapping between the things that are in your simulation, objects within the simulation, you need to be able to tell the package which ones to expose to the HLA federation, and how should they be exposed, how do your things inside the simulation map to the FOM". Weatherly then proceeds with stating that "unfortunately it may not be that easy because if they could do that then it may be a long way towards hiding the details of HLA yet". So, basically what experts mean by HLA compliancy of a COTS simulation package is the *support of the package for easily creating simulation models that can participate in an HLA federation*.

On the other hand there are some experts like Straßburger who claim that because "HLA compliance is always bound to a certain federate and to a certain simulation object model, you cannot say that a tool is HLA compliant, but only a federate is HLA compliant". Rabe has the same opinion, he thinks that "commercial simulators do not need to be HLA compliant". He thinks that "it is important that we have a specific system which allows the automatic adaptation of the model into a specific federation".

So it is not necessarily the package that should be HLA compliant, rather the package should provide a way to effortlessly create HLA compliant models. Of course if a tool satisfies these requirements it might deserve the description "HLA compliant COTS simulation package". Hardly any COTS simulation package provides interfaces that enable a user to create HLA compliant simulation models with small effort. Probably if the COTS simulation packages would evolve to "HLA compliant COTS simulation packages" as the experts define it, then there would be support to efficiently creation of HLA compliant simulation models.

The HLA standard supports distributed execution through an RTI, which is a software application that behaves like an operating system (DMSO 1998b). In order to enable distributed execution the RTI provides integrated services, such as federation management, declaration management, object management, ownership management, time management and data distributed management (Kuhl, et al. 1999). These services are accessed by simulation practitioners through predefined, approximately 150, interface functions (DMSO 1998b). Simulation practitioners are thus forced to consider within their COTS simulation models a certain number of RTI calls. According to Rabe "in order to access the HLA you need to understand all the HLA mechanisms, you need to understand the internal scheduling system of the simulation system, because this interferes heavily with HLA RTI".

Even assuming that such a simulation practitioner, with high level programming knowledge, and deep knowledge about HLA, exists and wants to access HLA from a simulation package, his task is still not simple due to the fact that COTS simulation packages are closed in the sense that they do not allow access to internal variables which

are essential for distributed simulation. Such a variable is, for example, the event calendar which is needed in time scheduling. Furthermore, there are concepts that exist in the HLA standard, which currently are not implemented in COTS simulation packages. Straßburger who has experience with ownership management between COTS simulation models claims that ownership management is "a concept that is not really common to these tools. So you have to find a way of mapping the logic to the mechanism of the package". Rabe recognizes this feature as well "you cannot implement it within the basic structure of the COTS simulator without going into the internals of the simulator and then it is no longer a COTS simulation package".

The COTS simulation packages are too closed to allow integration with the HLA standard and from the interview results we conclude that the vendors are not willing to change this. Instead most of the vendors want to maintain a customer lock situation by developing their own package to provide more functionality, instead of integrating it with other packages. For example, Möller argues that "when COTS simulation tool vendors develop these products and add modules for more and more types of simulations there maybe an interest in keeping the users inside the own architecture and not interoperate with other types of architectures. They do not want you to connect to competitors, so they want you to stay with the same simulation tool and the simulation tool should cover more and more functionality, which leads to a customer lock situation". Weatherly has the same opinion, he notices that "if you are the developer of a COTS simulation package or general framework, you are trying to argue that your package is extremely general purpose, does everything under the sun, it has all the capability, so this makes it hard for you to envision how simulations built in your package are going to deal with other simulations across the HLA". Granowetter claims that "there is no incentive for different vendors of different simulation packages to agree on a common interoperability standard". Rabe observes as well that COTS simulation package vendors do not want to take too much initiative because "simulation vendors are just trying to avoid that somebody uses other systems than theirs".

The willingness to support the HLA standard or to open up a COTS simulation package again relates to the cost benefit ratio. As McGregor, a representative of a COTS vendor argues "I can only think that there has not yet been a convincing financial argument for why any particular simulation vendor should modify their product or their documentation to explain to their users why is in their benefit to talk to HLA". Further he states that "Why would we encourage our users to integrate our products with our competitor's products? That is the basic question you have to ask. What is the economic reason for working with a competitor? So simple as that". This does not mean that they are not open for supporting distributed simulation or the HLA standard but as he states "if we could find a way that it would help increase our sales as well as theirs, perhaps, then you might have a business case for it, but otherwise there is no financial motivation to do that". Heinicke's observation regarding HLA support is more or less the same. He did a market research and he realized that "there is no money in it. There is no organization who wants to pay for it. We do not see a market where today the small or middle size companies would pay for it".

Considering the opinion of COTS vendors, depicted in Table 3.9, it seems that the COTS vendors do not necessarily exclude themselves from supporting distributed simulation, however they do not want to invest too much in it, because the benefit is not clear. Both Heinicke and McGregor explicitly state that they are open and interested to support any

distributed simulation standard, and if it is the case and they see benefit they will build alliances with other vendors, like Brooks Automation did with Simul8. Furthermore we should not just blame the COTS vendors that they do not support HLA in their packages, we should also take into account the complexity of the HLA standard and the domain for which was designed and developed. Elder from Simul8 Corporation, which is an organization that is quite willing to open up its simulation package for distributed simulation purposes, claims that "HLA is very military specific and so is weighted down by support for features not required in many cases". Straßburger thinks that "when defence designed HLA they did not have in mind any COTS simulation package". Straßburger's observation is supported by Weatherly who states that the defence simulation community does not focus on COTS simulation packages because "a lot of these big military simulations are really running on the edge of what can be done from a performance standpoint, and you cannot tune your system unless you own it all". Regarding these two groups, COTS and HLA, we are led to agree with Weatherly's observation, who states that "*the developers of these big simulation systems are just suspicious of the COTS simulation environment, and the COTS simulation environment people are suspicious of the HLA*."

Based on what the experts propose, a possible solution would be to aim at something simpler than the HLA standard, which would give COTS simulation package vendors the opportunity to implement the needed distributed functionality within their package in an easy and inexpensive way.

4.4 Concluding Remarks Concerning Distributed Simulation in Industry

In the previous sections we presented some arguments, divided into three groups, why industry does not perceive much use of the HLA standard in industry. As we have mentioned in the introductory part of this chapter, the problem is not so much HLA specific, but more general and relates in fact to the reluctance to use distributed simulation in industry. The role of HLA, the most advanced and well-known distributed simulation solution approach, is more the one of a test bed and a basis for comparison. Further, it could also play the role and take the opportunity to promote distributed simulation.

It seems that the main reason why industrial practitioners do not seem to apply distributed simulation in general is the perceived cost benefit ratio. Although one of the aims of HLA is to minimize cost, as we have perceived the cost still remains too high comparing to the benefits. The cost benefit comparison however, would not be fair if we would focus only on the cost side and not on the benefit side as well. For this reason, we should examine whether simulation practitioners are fully aware of the benefits. Although there are some explicit benefits of distributed simulation, as we discussed at the beginning of this section, we believe that distributed simulation, being a quite new approach in industry, has some benefits that are as yet invisible for the end user.

The invisibility of the benefits, according to McGregor, relates to the business model currently applied, namely designing and developing "throw away" simulation models. The primary aim of industry as McGregor, Rabe and Morse state is to get a solution as fast and as cheap as possible and they have a tool, COTS simulation packages, to achieve this aim. However, these tools are designed and developed with the above business model in mind;

they are not supporting design and development of distributed simulation in a proper way. Simulation practitioners who regularly apply this tool do not have the possibility to carry out distributed simulation, they do not have a chance to do so, therefore the benefit that they might discover remains invisible. Of course, a straightforward way would be to first disclose the benefits and then provide tools. Very often, however, added value can be discovered only through trial or application of the tool itself in this case. But neither practitioners nor vendors are willing to take a risk to invest in invisible benefits. This is reflected by Ryde's observation regarding the willingness to pay for distributed simulation. Ryde argues that "speaking with vendors and simulation practitioners, I have found that most of them are not willing to pay much more extra for the privilege of interoperating simulation models. Many would be willing to pay an extra typically between 5 to 10% of the original price of the package". Less then 10% reflects a minor interest. The question here is: who is going to take the first step? The first step could be taken by vendors investing by including distributed simulation in their package and allowing the users to experiment with it. By using it, certainly new opportunities and benefits will be discovered. This is what usually occurs. Compare this with the personal computers we are daily applying. Initially computers were designed for complex arithmetic calculations for defence and the space industry, but later, especially after appearance of cheap personal computers, people started to see more benefits.

Although some simulation practitioners perceive benefits of distributed simulation, unfortunately the inherent complexity of distributed design and development and the absence of tools which might handle this complexity lead to the fact that they scarcely apply it.

Basically we experience a chicken and egg scenario. On the one hand simulation practitioners do not see the benefits of distributed simulation because they do not have a tool to experiment with, and therefore they do not request from the tool vendor a tool that supports distributed simulation. On the other hand tool vendors do not provide a tool because end users do not request it.

Nevertheless, time seems to be ripe. McGregor argues that distributed simulation is a "quite large area, but remains as yet unexploited" or as Straßburger said "industry does not understand yet that this is a topic of the future". McGregor also thinks that the main cause is marketing. He thinks that there is room for distributed simulation and for protocols that implement the concept. The most important need is to create a market for it. By providing easy and appropriate technical solutions for end users who can play with it, a market will be created, so basically *pushing the technology will create a market pull*.

4.5 Perspectives on Distributed Simulation in Industry

In the previous sections we have validated the hypothesis "The HLA standard is rarely used for integrating simulation models designed and developed in COTS simulation packages". Further, we highlighted several reasons for this, reasons relating to distributed simulation in general, to the HLA standard itself, and to the relation between the COTS simulation packages and the HLA standard. Interestingly all three categories connect to the very general economic reason of the cost benefit ratio.

Based on these reasons, in this section we aim to answer the third question we have posed at the beginning of the survey and, accordingly, we aim to discuss:

- ***How*** *can we make distributed simulation and existing distributed solutions, like HLA, more attractive to the industrial community?*

In order to make distributed simulation more attractive for industry there is a need for a COTS based distributed simulation architecture which might improve the perceptible cost benefit ratio. In general system engineering theory there exists a system development life cycle which describes the required steps for designing and developing a system, like a distributed architecture (Hoffer*, et al.* 2002, pg. 18). The third step of this development life cycle is the analysis. The analysis aims to define the requirements for designing the architecture. This entails two main steps. First, requirements should be determined and structured. Then, alternative system design strategies should be generated based on the requirements and the most suitable one should be selected (Hoffer*, et al.* 2002).

Therefore, in the first place we have to:

1. *Identify and structure the requirements for an industrial COTS based distributed simulation architecture.*

Then, in the second phase, we should:

2. *Identify the possible perspectives for implementing these requirements*

Based on this subdivision the rest of this section is divided in two parts.

4.5.1 Requirements for Industrial COTS based Distributed Simulation

4.5.1.1 Sources for identifying the requirements

In order to deduce the requirements for industrial COTS based distributed simulation we use two sources:

- Literature;
- The questionnaire and interview surveys.

Requirements extracted from literature

Given that the HLA standard offers the most mature solution at the moment, the most logical way to start with would be to look at the official requirement list of the HLA standard itself. This list would offer a good indication on which our requirements could be based. With this reasoning in mind, we have investigated some potential sources, like the DMSO website, where we thought we might come across a kind of requirement list, but unfortunately we could not find such a list.

As a next trial, we asked the experts during the interview where can we find the requirements. The most well informed person who could give some hints related to this problem, Weatherly, the leader of the HLA infrastructure development team, unfortunately believes that “there is no HLA formal requirement document, except that the undersecretary of defence said: ‘DMSO builds me a system that unifies DIS and ALSP that can be used across the department of defence’ ”. A fixed official requirement list does not

exist because apparently HLA went through a spiral evolution, where the initial capability was built by defence and the specification evolved based on feedbacks by users who had tested it. Unfortunately we were not in the position to get documents related to this process from which some requirements could have been extracted.

Although we could not find the official requirement list of the HLA standard, we have extracted some of them from the benefits of HLA mentioned in the literature by (Kuhl*, et al.* 1999), (DMSO 1998b). This list of benefits was enriched with the answers of the experts in the interview who explicitly pointed to the advantages of the HLA standard. An additional source of requirements stem from the literature on the design and development of distributed systems in general in the software engineering area, described in (Tanenbaum and Van Steen 2002). Through this process we uncovered some high level requirements for distributed simulation in general and low level requirements regarding the technical design and development of a distributed simulation architecture.

Requirements extracted from the survey

Besides the source provided by literature, the results of the interview suggest requirements. In the previous sections on the evaluation of the survey we can find valuable remarks on this issue. Basically the arguments pointing to the drawbacks and difficulties in applying HLA can be used to identify new requirements aimed at eliminating these deficiencies. More specifically, the arguments pointing to the reasons why HLA has not gathered ground in industry are very useful to help us to identify the requirements for an architecture which is meant to be primarily for industry and for simulation practitioners who regularly apply COTS simulation packages.

Apart from the survey results that we discussed before, there are as yet unevaluated answers that can point at possible requirements. In order to disclose these features we asked the experts what they expect from an idealized architecture:

> *How do you imagine an 'idealized' distributed simulation architecture for the industrial domain where COTS simulation packages are frequently applied?*

In order to determine the requirements even sharper, we posed several subquestions related to this issue, such as:

- *What functionality should the architecture offer?*
- *What is the minimal functionality that the architecture should offer?*
- *How should the interfaces look like?*

The aim of these questions is to help us to validate and extend the requirements which we could extract from literature. Given that various experts on distributed simulation participated in the interview, both from industry and defence, with diverse viewing angles, knowledge and experience, we expected to collect comprehensive requirements. COTS simulation package vendors, on the one hand, can indicate requirements which can be associated to the relation between the distributed simulation architecture and their COTS simulation package. Industrial simulation practitioners, on the other hand, can express their expectations regarding the ease of applying distributed simulation and will therefore indicate technical requirements.

Requirement categories

We observed in Sections 4.3 and 4.4 that in order to make distributed simulation more attractive to the industrial community, we need to specify an architecture that manifestly improves the cost benefit ratio of distributed simulation in industry. In the same sections we analyzed this cost benefit ratio, and we identified several arguments for the high cost and the low benefit as felt in industry. Considering these arguments we distinguish at a high conceptual level four main requirements related to the design and development of a COTS based distributed simulation architecture, which can be divided into more detailed requirements. Accordingly, we believe that a COTS based distributed simulation architecture:

- Should support in a natural way the projects which have the earlier mentioned characteristics that *make using distributed simulation feasible;*
- Should provide *in an* efficient *and effective way services* for interoperating simulation models;
- Should have a *natural* relation *with COTS simulation packages* because we aim to use it for models created in these packages;
- Should not require additional distributed system knowledge from industrial simulation practitioners, it should be *transparent* to the user.

The first requisite is a sine qua non. If the proposed architecture will not be of help to for those projects that have the characteristics that encourage the user to apply distributed simulation then it will never obtain an entry in the industrial simulation community. The second issue is related to the technical design of the architecture that should provide services for integrating models in an efficient and effective way, which should be comparable with monolithic solutions. In the third place, due to the fact that this architecture is aimed to be applied for interoperating simulation models designed and developed in COTS simulation packages, it is necessary that it has a natural relation with these packages. Finally, the fourth general requisite takes into account the expectation that industrial simulation practitioners expect to be able to apply the solution in more or less effortless way.

In order to group the requirements in a structured and straightforward way we order them according to the four main requisites presented above. Thus we arrive at the following groups:

- *Requirements related to the characteristics favouring distributed simulation projects.* In this group we collect high level requirements that are related to the project characteristics which we have discussed in Section 4.2.
- *Requirements related to the technical design of the architecture.* Whereas the first set of requirements is high level, these requirements are mainly low level and discuss the services that architecture should offer.
- *Requirements connected with the relation to COTS simulation packages.* Since we specifically focus on the industrial area, the distributed simulation architecture should support COTS based distributed simulation. The requirements in this third category refer to the relation as desired between the distributed simulation architecture and the COTS simulation packages.
- *Requirements related to the efforts needed from the simulation practitioners.* We found that the complexity of existing distributed simulation architectures discourages simulation practitioners from applying this concept. The requirements belonging to this category aim to minimize the involvement of the user when applying the architecture.

Next we identify the requirements that belong to each of the four categories laid out above.

4.5.1.2 List of requirements

Requirements related to the characteristics favouring distributed simulation projects

The different project characteristics we discussed in Section 4.2 are a suitable guideline to identify the requirements in this group. The project characteristics can straightforwardly be related to requirements:

1. *The architecture should enable reusability of already existing simulation models.*
2. *The architecture should support integration of heterogeneous simulation models.*
3. *The architecture should facilitate collaborative simulation design and development.*
4. *The architecture should allow for information hiding between simulation models.*

Requirements related to the technical design of the architecture

Distributed simulation architectures should be designed carefully since there are many pitfalls for the unwary (Tanenbaum and Van Steen 2002). Tanenbaum specifies five key design issues: transparency, flexibility, reliability, performance and scalability (Tanenbaum 1995, pg. 22-31). In laying out our requirements we will take each of these issues into account separately. An additional issue is how the architecture will support integration and interoperability of different simulation models. We base our discussion on this issue on Linthicum's work, who discusses different possibilities for distributed application integration (Linthicum 2003). Additionally we try to derive requirements from the benefits of existing distributed simulation architectures, and last but not least from the results of the survey. These requirements can be formulated at a rather low level, that is technical and/or programming level, but should be translatable to the level on which the simulation practitioners operate.

Service Oriented Structure

5. *The overall functionality should be organized as a set of services.*

   In order to carry out distributed simulation, specific distributed services are needed, such as services that take care of data exchange between simulation models or time synchronization between the models. The currently available COTS simulation packages do not provide such services. Therefore in order to connect two COTS based simulation models and allow distributed execution these kinds of services or functionality should be part of the distributed simulation architecture. The various services can be either implemented as separate modules, or can be encapsulated in one application.

6. *The services provided should be COTS packages independent.*

   In order to have a general, widely applicable architecture the services should not be defined in terms of concepts implemented in a specific COTS simulation package that might not be present in other packages. The architecture should be based on general distributed concepts and general COTS concepts.

7. *The services should be separate from the simulation models.*

   We deduced this requirement from Weatherly's opinion, however his opinion was stronger. According to him, besides the separation of the services, they could be placed on a centralized server to which clients could link. In order to achieve a reasonable solution he advises to "keep the federate, keep the simulation light, keep the RTI's out of their machine and put all the RTI's on a centralized server".

8. *The architecture should provide at least three basic groups of services. That is:*

   - *Services for starting, stopping and periodically checking the execution of the collection of distributed simulation models.*
   - *Services for data representation and exchange.*
   - *Services for time synchronization*

This minimal functionality was explicitly mentioned by the experts in the interview. However, as it stands now, these basic functionality should be sufficient for industry, in order to enable distributed simulation to gather ground. Probably once this is arrived at the set will increase and additional functionality will be added.

Flexibility

9. *The distributed simulation architecture should be extendible in the sense that it should allow simulation vendors and practitioners to provide additional services that are specific for their tasks.*

   Since the field of distributed simulation is just in its infancy, the design should be made with the idea to allow for future changes. The extendibility of the architecture is a required feature because it will make distributed simulation more flexible, and applicable for a variety of problems. A danger is that too many extensions could lead to the reinvention of the HLA standard. As Weatherly states "extendibility of the functionality is a trade-off (…) with extendibility we can innovate things", however he warns us that both adding new services or extending existing ones forces the implementer to reconsider the implementation of the already existing services, because of intricate relationships that might exist between the services. It even could lead to a change in the specification of existing services. Several experts, such as Weatherly, Fujimoto, Morse, Huiskamp, and Malcolm, express the opinion that it is interesting to investigate the extendibility of the functionality for research purposes. For this reason they imagine an open source RTI that would be maintained by a community and that would be useful for the research community to introduce and test innovative functionality. The extendibility of functionality would not be a completely novel issue, Granowetter draws our attention that the MÄK RTI supports extendibility of functionality, which is a missing feature of the DMSO RTI.

10. *The distributed simulation architecture should allow for replaceable elements (e.g., it should allow simulation vendors and practitioners to replace already existing functionality with more efficient ones).*

    In contrast to extendibility, the replaceable feature should not influence the specification of already existing services. Given a certain functionality, like time synchronization, which can be triggered through different functions, the end user can replace the algorithm behind this service while still adhering to the specification of the functions. Of course this will require extensive programming and system architecture knowledge, but we propose that the architecture should be flexible enough to accommodate that somebody might be able to improve its efficiency.

Transparency

11. *The architecture should hide low level implementation details from the simulation practitioners.*

    Although the functionality of the architecture should be extendible and replaceable for advanced users, in general the architecture should be built in such a way as to hide low level details from the simulation practitioners. It should provide interface functions directly related only to the basic functionality which should be clearly

defined and which can be easily understood and applied by all industrial oriented simulation practitioners.

Data representation and exchange/communication

12. *Communication used by the architecture should be based on one of the standard communication protocols supported by most COTS simulation packages.*

    The data or objects that are exchanged between simulation models should be represented in a common, standard way, not associated to one ore more specific COTS simulation packages. In order to exchange data between two computers, there is a need for a low level communication protocol, such as WinSock, CORBA, .NET, etc. The services supported by the distributed simulation architecture should be based on one of these effective and efficient standardized communication protocols that are available. In order to apply such a protocol, designed at a quite low technical level, the general simulation practitioner should possess adequate general programming knowledge. To avoid this, the architecture should be designed in a way as to hide the details of the low level protocol chosen from the simulation practitioner, by offering easily understandable and applicable interfaces.

13. *Representation of the data and object model and data exchange should be based on standards or quasi standards which are defined by the industrial community. This industrial community should be represented by simulation practitioners, simulation researchers and COTS simulation vendors.*

    This requirement is deduced not only from the opinion of the interviewed experts, but also from ongoing research initiated in this direction by the HLA-CSPIF forum[35], which contains members from these three communities. One of the aims of the forum is to develop standard representations for data exchange and to develop a standard exchange mechanism. The forum recently initiated a product development group which was approved by SISO under the name COTS Simulation Package Interoperability Product Development Group (SAC-CSPI-PDG).

Performance

14. *The architecture should perform efficiently when simulation models interoperate with each other.*

    Efficiency is a very important issue, which was already discussed in more detail in Section 3.3.2. The pitch RTI and MÄK RTI already proved that it is possible to create faster RTI's than the RTI offered by DMSO. Industry, as we have seen before, is accustomed to well performing monolithic COTS simulation models executed on one computer. Integration of these models through the HLA standard degrades performance in an unacceptable way. Apart from efficiency of simulation execution, efficiency of the setup time of a simulation run should be considered and improved in distributed simulation architectures as well. It seems that this feature is not yet solved in HLA: based on his own experience, Malcolm complains about the long

[35] http://www.cspif.com

setup time in HLA, the simulation requiring "long delay when a federate intends to join a federation".

Scalability

15. *The architecture should be scalable in the sense that it should handle hundreds of coupled simulation models in a relative efficient way.*

    The architecture should allow integration of any number of distributed simulation models without any constraints other than that they should adhere to the requirements prescribed by the integration architecture. The architecture should be robust enough to not to impose constraints regarding the number of participating simulation models.

16. *Besides coupling simulation models, the architecture should enable coupling of external applications or real equipment.*

    By external applications we also take into consideration algorithms implemented in general programming languages that execute other tasks than simulation that play a role in the whole distributed simulation. This requirement enables integration of heterogeneous COTS simulation models and software (e.g., traffic control software) or applications designed and developed in general programming languages (e.g., an algorithm for chemical reactions), which serves a certain purpose within the distributed simulation execution. These external applications should connect either directly if it is possible or through middleware to the protocol of the distributed simulation architecture. Additionally the distributed simulation architecture should provide the possibility to integrate real equipment.

Miscellaneous

17. *The architecture should be based on TCP/IP as the main protocol for Internet.*

    The Internet is defined as a network of many networks that interconnect worldwide and use the TCP/IP protocol. Internet applications offer the infrastructure and capability to support geographically dispersed users with different language and cultural requirements. The Internet provides support for globalization that entails collaboration of different organizations from different countries. This requirement is meant to enable collaborative simulation design and development between geographically dispersed organizations which are involved in an interorganisational simulation project. In this way organizations can keep their employees at their office, they can design and develop their own part within the office and at any time they can test their model with other organization's models through the Internet. Besides the collaborative aspects, this requirement enables information hiding between organizations as well. The sensitive algorithms and data need not be physically moved to another location where they might be vulnerable. Instead they are kept in house sharing only required data through the internet.

18. *The architecture should incorporate fault tolerance mechanisms.*

    This is an important issue, mentioned in (Fujimoto 2000) as well. Fault tolerance mechanisms are intended for cases where one of the simulations breaks down for some reason. In such a case the mechanisms should take care that for example other

simulations pick up the job of the failed simulation and, if possible, allow the simulation run to proceed despite the failure. Next to internal system related breakdowns, external failures might occur as well, like breakdown of the network wire connection. Möller points out that one of the major things, that should be improved in HLA as well, is a better fault tolerance, "because maybe somebody cuts the network wire, maybe there is a problem with one of the computers, how can all the simulations keep running when one system is not working properly".

Requirements connected with the relation to COTS simulation packages

This group of requirements especially focuses on the relation between the distributed simulation architecture and COTS simulation packages. As a basis for these requirements we consider Section 4.3.3, which presents arguments regarding the relation between HLA and COTS that might be one of the causes that the HLA standard is rarely applied in industry.

19. *The architecture should support integration of simulation models designed and developed in a large variety of COTS simulation packages.*

    In order to be accepted by the industrial community the architecture should support as many COTS simulation packages as possible. Basically, an architecture developed independently from the packages providing an easily applicable, high level tool for the integration of simulation models written using COTS simulation packages, would stimulate COTS vendors to further take the initiative to adapt their COTS simulation package to support this architecture.

20. *The services offered by the distributed simulation architecture should be naturally expressible in terms of concepts used in COTS simulation packages.*

    This requirement stems from the opinion of most of the experts who explicitly stated that the distributed simulation concepts should be hidden from the end user and it should be presented on high level concepts offered in COTS simulation packages.

Requirements related to the efforts needed from the simulation practitioners

In the interview several experts agreed that the HLA standard has a quite steep learning curve, requiring a lot of time from practitioners to get used to it. Furthermore, even if someone has learned how it works, it still requires additional effort to apply it given its low transparency. These features make HLA unattractive for simulation practitioners who are not used to low level programming. Experts' opinions regarding these characteristics of HLA, and a discussion about these issues is presented in Section 4.3.1. The requirements belonging to this fourth category are mainly deduced from this discussion.

21. *Building a COTS simulation model which is aimed to be used as a submodel in a distributed simulation should not require more effort and knowledge than building it as a standalone model.*

    This is one of the most important requirements which strongly relates to the economic reason that distributed simulation is hardly applied. An architecture with this property is bound to improve the cost benefit ratio by reducing learning costs as much as possible, to provide practitioners an easily applicable tool, and,

consequently to let them discover new opportunities for and benefits of distributed simulation. If the effort required for designing and developing distributed simulation would be comparable with the monolithic simulation case then there would be a chance that simulation practitioners would start experimenting with it thus discovering new potential. Consequently distributed simulation would gain more ground in the industrial community.

22. *The effort from the simulation practitioner needed to setup and manage an execution run of a distributed simulation model should be minimal.*

    Apart from design and development, execution of a distributed simulation should require minimal additional effort compared to the effort needed for executing a monolithic simulation. This could be achieved if, for the end user the way of starting and running a distributed simulation would be more or less the same as starting and running a monolithic simulation within one COTS simulation package.

23. *The architecture should provide a graphical user interface (GUI) facility that allows the simulation practitioner to configure and to monitor a simulation run.*

    Users should be able to configure the simulation run settings in an easy way through a clear graphical interface. After starting the simulation run the end user should be able to manage the simulation execution. Insight in the distributed simulation execution when required should be provided by a GUI.

24. *The architecture should allow or even facilitate the possibility to debug the simulation.*

    An interactive debugging capability might allow the simulation practitioner, for example, to see the status of the objects they simulate. COTS simulation packages already provide built-in functionality for debugging a simulation run. In distributed simulation debugging is not as straightforward as in the monolithic case. In order to let users verify the execution of distributed simulation the architecture should provide or facilitate debugging facilities. As a starting point, the solution can be based on something similar as the RTI Spy by MÄK Technologies, which “is on the top of RTI functionality”, as Granowetter says, allowing the user to obtain diagnosis and debug information, e.g., regarding connectivity problems or incorrect RTI calls, etc. This is a useful feature for those people who are interested in what is happening underneath the distributed simulation, although this is still quite low level for general simulation practitioners. Debugging facilities are needed on a higher lever.

25. *The user should be able to access all simulation results in a consistent format in order to examine them after completion of a simulation run.*

    The initial aim of a simulation, just like the aim of other problem solving strategies, is to solve a problem or to answer a question. To this end, execution of a simulation will generate data to be analyzed in order to solve the problem or answer the question. When executing a monolithic simulation within a specific COTS simulation environment the COTS package collects all data and sometimes provides additional tools for analyzing them. If we carry out a distributed simulation then we have to collect data produced by different models. Therefore either a centralized

store should be available or one of the packages should be able to collect the simulation data.

The list of requirements identified above is not an exhaustive one, it is just an initiative that can be extended and further improved on. We collected requirements that we expect a COTS based distributed simulation architecture to satisfy, however when we intend to design such an architecture we certainly do not expect that all requirements will be fulfilled at once. Rather, we expect the design of such an architecture to be evolutionary; first only the most relevant requirements will be taken into account, and the less important ones could be skipped. However, finally we should strive to satisfy all of them in order to allow the industrial community to recognize the advantages of such a distributed architecture.

Furthermore, very probably, additional requirements that are not in our list, will crop up during the design and development phases of the architecture, or even while applying the architecture for specific simulation integration problems. In this case we should take into account those requirements and extend the list. Although our set of requirements is in an initial phase, we believe that the requirements that it contains provide a beneficial starting point for stimulating a technical push of distributed simulation in industry.

4.5.2 Possible Implementations

Given the list above, the next step is to plan how to implement distributed simulation architectures that satisfy these requirements. This entails three decision problems. First of all, we need to find individuals, communities or organizations which can implement the requirements. Then it should be specified who will be the owner of this solution, and finally, the acceptance of the implementation should be stimulated. Regarding these three problems we make the following statements.

1. The distributed simulation architecture should be designed and maintained by a group of experts from heterogeneous areas, involving industrial simulation practitioners, simulation researchers, software engineers, and, last but not least, COTS simulation package vendors.
2. The distributed simulation architecture should be *open source.* Several experts, such as Weatherly, Morse, Fujimoto, Huiskamp and Malcolm explicitly pointed to the benefits of an open source distributed simulation architecture. Morse mentions two important reasons why she prefers an open source approach. The first reason is that “open source provides a key component on a market”, and secondly “it keeps people honest”. For the groups identified above the open source approach seems to be an appropriate way for designing and developing the distributed simulation architecture for industry.
3. The distributed simulation architecture should be or become a *standard.* Experts claim that one of the most important benefits of HLA is the fact that it is an IEEE standard. When the open source prototype reaches a level of a product that can be applied by a broad industrial community it should be turned into a standard.

Basically we recognize two possibilities to implement the above requirements. We could *design a new architecture specifically for COTS based distributed simulation* or we could

adapt the HLA implementations to support more requirements. This latter approach raises a few questions: What is the most appropriate choice? Would it be possible to consider one or more of the existing HLA RTI's to implement the above requirements? If so, in which measure and how can we take these into account? How can we tailor the existing HLA RTI's to satisfy all the above requirements and further, how can we profit from this approach?

In order to answer these questions we need to investigate the two possible ways to implement the above requirements more deeply:

- *Implement the above requirements based on one of the existing HLA RTI's and try to adapt this architecture to the requirements.*
- *Implement the above requirements as an innovative lightweight distributed simulation architecture which is specifically oriented towards integrating industrial COTS based simulation models.*

We believe that there is a trade off between the two choices. Both approaches have their advantages and disadvantages. In order to identify them, next we briefly elaborate on and analyze the two of them.

Implementation of the requirements on top of an existing HLA RTI

Implementing the requirements on top of an existing HLA RTI is attractive especially if one considers the fact that the HLA RTI implementation already exists and further, that HLA is an accepted IEEE 1516 standard. In this case, there is no need to implement most of the low level technical services because those are already provided by the HLA RTI.

Analyzing the identified requirements in this section, especially the ones regarding the technical design of the architecture, we can deduce that we aim at an architecture that implements less functionality than the HLA standard does. We need only the functionality that is essential for industrial COTS based simulation projects. In this way the envisioned architecture is more *lightweight* than the current HLA RTI's. Therefore, if one considers an HLA RTI one should ignore irrelevant functionality, simplifying the original architecture.

In order to hide the technical complexity of an HLA RTI, well defined adaptors or middleware could be designed and developed. These adaptors offer only the required services hiding those services which are not needed for the industrial simulation practitioners. Earlier we already pointed to research that aims to design and develop such adaptors or middleware (McLean and Riddick 2000), (Rabe*, et al.* 2001), (Straßburger 2001), (Revetria*, et al.* 2003). These adaptors provide solutions towards hiding the complexity of HLA and for connecting COTS simulation packages through an HLA RTI but as we have discussed before they are still prototypes still requiring specific knowledge to apply them.

On the other hand, this approach precludes the possibility to easily fulfil some other requirements, currently not present in RTI's, because access to the full source code is needed. Such requirements are extendibility of the functionality (unless we are using the MÄK RTI), or replaceable functionality. Furthermore, the simulation practitioner must tolerate the inefficiency of the performance of the selected HLA RTI implementation.

In order to provide an acceptable solution when choosing for this approach to implement the requirements, a choice of an efficient and powerful HLA RTI implementation is obligatory. However, we must recall Fujimoto's argument, that a fully functional HLA RTI is always less efficient than a reduced one. Besides the original HLA standard requirements, this HLA RTI should additionally satisfy the architectural technical design requirements discussed above. In order to hide low level details from the end user one needs to provide transparency, which can be realized by well designed and developed adaptors or middleware. These adaptors should fit naturally within a COTS simulation package environment and should hide details of the distributed solution from the end user. Furthermore, in order to encourage reusability and to avoid inconsistency problems during data exchange, standard reference object models are needed which can be put to use by the adaptors.

The efficient and powerful RTI's, required in this case, should be provided by HLA vendors, who need to extend their market niche towards the COTS based industrial area. The adaptors could be designed and developed by COTS vendors or third party adaptor vendors, while the standard reference object models could be maintained by non-profit organizations, similar to the Object Management Group (OMG), which primarily aims to produce and maintain computer industry specifications for interoperable enterprise applications[36].

Implementation of the requirements through a novel lightweight distributed simulation architecture

The second way to implement the above requirements is to design and develop a *lightweight distributed simulation architecture* from scratch. The advantage of this architecture would be that it will not depend on any HLA RTI implementation, instead the whole concept within the architecture will be designed along the above requirements only. This is feasible in view of the fact that the architecture is intended for a special community, namely the industrial COTS based simulation practitioners, and in view of the fact that it should be easily supported by COTS simulation packages.

Comparing to the previous choice, this option offers freedom to the designer who is not limited by a tool in which essential details are barely accessible and who is not forced to implement the 'lightweight' requirements on top of a selected heavyweight HLA RTI. The freedom that one can gain by this choice enables one to decide from the beginnings for an appropriate structure of the architecture. One can decide, for example, to go for a centralized distributed simulation server which provides services through the web. Furthermore, one has control over how the services will be implemented: in one monolithic application, such as an HLA RTI, or in separate service components. This approach offers more flexibility for designing and developing the architecture on which the whole concept will rest.

As our 20th requirement states, the services offered by the architecture should be naturally expressible in term of concepts used in COTS simulation packages. The previous approach considered adaptors for expressing the distributed services into COTS package concepts. As we have presented in Section 3.5.5 the simulation practitioners have different opinions

[36] http://www.omg.org/

on the way HLA concepts should be included in COTS simulation packages. On the one hand, some experts believe that the HLA concept should be included in the modelling paradigm of the COTS simulation tool. On the other hand, other experts argue that implementing the concept of HLA in the commercial tool is not beneficial for the end user, because then the end user in fact has again to deal with the HLA and the HLA mechanisms. For instance Rabe thinks that the best solution is the use of adaptors or middleware because with complete integration "you save the adaptor, but you are even less open than before, than having the adaptor as a separate tool. You could even run the same commercial system with 2 or 3 types of adaptors if it is necessary and if the adaptor is inside the tool you are limited to one adaptor type". He underpins his point of view by arguing that "the RTI's which are now on the market are not really compatible. That means if you exchange the RTI you have to change a little bit the adaptor. That means that if the adaptor is integrated in the commercial tool you are limited to one specific commercial RTI". According to him, using intermediate flexible adaptors we have more chance for compatibility. The lightweight distributed simulation architecture we propose should allow an implementation according to both trends: it should give COTS vendors the possibility to incorporate the concept within the tool, but it should also support creation of adaptors for tools that do not choose for include the concepts. If the interfacing is simple enough we expect that the COTS simulation package vendors will follow the first trend, namely that they will be willing to include the function calls within their package and they will express them in the terms of the concepts used in the package. Otherwise, adaptors can be provided either by COTS vendors or third party developers.

The trade off is that the previous approach has the benefit that it is based on an already existing implemented architecture, whilst this approach has the strong benefit that it can lead to innovative solutions. Furthermore, the architecture can be designed and developed as an open source allowing the research community to experiment with it, for example to investigate and to improve the performance of certain services.

In contrast to the approach to build on top of HLA this one offers more flexibility for the different communities. However after a while, in order to be accepted it should certainly go through a standardization process. Since HLA is already an accepted standard, the first approach has an advantage in this respect.

Our position

Although HLA is an accepted standard for distributed simulation and HLA implementations exist that might provide the basis for implementing the above requirements, we believe that more standards might deserve a place. We do not really believe in a universal standard which can solve all problems, instead it is important to design and develop the most appropriate tool for a certain set of problems.

HLA is a good standard but it was designed and developed for the defence community and it is commonly hardly used outside this community. The HLA implementation is complex as it is intended to solve complex problems. On the one hand thus, HLA has its place in the defence community or in any community where complex large scale distributed simulations are conducted. On the other hand, a large number of industrial projects are characterized as small or middle size projects and these might not benefit from HLA, be it only on cost considerations. We think that industrial applications require less complex solutions and by keeping the complex HLA implementation as a basis of the architecture, inherent complexity and performance inefficiencies will remain. As we see now, using HLA in industry is prohibitive because we keep carrying the burden of the mismatch between HLA and industrial simulation. In this sense we must pay for all the penalties, such as inefficiency, complexity, unwillingness by COTS vendors to offer support, etc. This forces us to consider the second approach, a more lightweight distributed simulation architecture, an architecture that will eliminate some of the complexities, would be more suitable for the COTS simulation world and would have its place besides the implementations of the HLA standard.

Although in this research we focus only on the second approach, it would be worthwhile if another research, e.g., a forthcoming Ph.D. study, would be set up in order to investigate the first approach. Then in the end these two approaches could be compared which would provide a comprehensive picture for the communities involved. Furthermore, we expect that the implemented solutions with proven industrial case studies will provide escape from the chicken egg scenario and will create a market pull.

5 THE FAMAS SIMULATION BACKBONE ARCHITECTURE

5.1 Introduction

In addition to the theoretical research presented in Chapter 3 and 4 we have carried out practical industrial projects, in which we studied the possibility to apply, and applied distributed simulation. In the previous chapter we concluded that a lightweight architecture for connecting simulation models might be appropriate for industry. In this chapter such an architecture is presented that we developed and used in order to be able to carry out the distributed simulation projects. The FAMAS Simulation Backbone Architecture, which we refer to, was designed and developed as part of the FAMAS.MV2 project. Although the practical projects were independently carried out from the research presented in Chapter 3 and 4, we would like to test to what degree the FAMAS Simulation Backbone Architecture satisfies the requirements presented in Section 4.5.1.

The methodology used to describe this architecture follows the standard system development life cycle and involves four phases: problem description, analysis, design and implementation. The description of the problem that initiated the backbone is covered briefly in Section 5.2. The aim of the analysis phase, discussed in Section 5.3 is to identify the requirements that the FAMAS Simulation Backbone Architecture should satisfy. These requirements are specific for the FAMAS project and are independent of the requirements presented in Section 4.5.1. Design details based on the previous steps are extensively presented in Section 5.4, while the implemented solutions in Section 5.5. The implemented simulation backbone is evaluated in the next chapter. There, first we evaluate the architecture through two case studies against the FAMAS project requirements presented in the analysis phase. Then, in Chapter 7 we evaluate it with respect to the general requirements presented in Section 4.5.1 in order to test its appropriateness as a distributed simulation solution for industrial domain.

5.2 Problem description

Rotterdam desires to become one of the best ports in the world and to keep its leading position in the European market. In order to achieve this, future developments are planned especially in the direction of trans-shipment of containers, in the field of chemistry, and of distribution[37]. In order to secure innovation, further growth and development in these areas additional room is needed. Therefore a new location has been assigned, called Maasvlakte 2, with a good connection to port related companies. In order to achieve the stated goal and to effectively use the new territory the FAMAS (First All Modes All Sizes) plan was initiated, that embraced the FAMAS.MV2 subplan. The aim of the FAMAS.MV2 subplan was to "conceptualize and design, in public private partnership, standard high throughput automated container terminals with a connecting Inter Terminal Transport (ITT) system, on the basis of the requirements for Maasvlakte 2 in the port of Rotterdam for the year of 2020" (De Hartog, *et al.* 2001). Although, the land reclaim still has to take place, the aim

[37] Source: http://www.portofrotterdam.com/organizations/NL/Themas/Maasvlakte2/Index.asp

of the subplan was to prepare a concept design in order to look far in advance how the future land should be used. The planning was to present the concept design in 2003 and to have the terminal operational in 2020.

In (De Hartog, *et al.* 2001) a vision is presented on container logistics in the port of Rotterdam for the year 2020, based on market research into the current state of the art in container logistics, the expected developments and future trends. The report contains a preliminary investigation of the concept design of this new port and the performance of the operation of the new port is estimated. In order to look forward and analyze how the future port will operate simulation modelling approach was applied. This approach was chosen to improve the total quality of the working of the future container terminals.

In the projects within the FAMAS.MV2 research plan several organizations participated which all were developing their own simulation model to represent a specific process or part of the port. One organization was, for example, responsible for designing the container terminals, another one for designing inter terminal transportation between the terminals, while a third one designed models of the AGV's (Automated Guided Vehicles) that carry containers within, or between container terminals. The participants representing different organizations have been designing and developing their models independently and in parallel, and they were required to collaborate in some sense to carry out certain projects within the whole research plan. The various organizations had the freedom to use any programming environment or COTS simulation package to develop their models, fact that lead to the presence of heterogeneous models within the projects.

Some organizations had already been involved in and invested in other FAMAS related projects, and, as a consequence, they could and even whished to reuse already existing simulation models from these projects. Reusability of existing models in combination with new models was, thus, an important consideration within the FAMAS.MV2 research plan, especially for economic reasons.

Summarizing the considerations above, the projects within the FAMAS.MV2 research plan can be characterized as projects where some models are reused, models are heterogeneous and collaborative design and development is applied. Reusability, collaborative design and development, and the ability to combine heterogeneous models are the main benefits of distributed model based problem solving strategy, as discussed in detail in Section 2.3. This nature of the FAMAS projects implies thus distributed model based problem solving strategy that requires the integration of the various models. Integration enables one to analyze interactions and side effects that can be never investigated if we simulate the models separately. Furthermore using integration on can obtain feedback regarding the performance and achievement of the whole integrated terminal system.

In accordance with this reasoning, the organizations responsible for the FAMAS research plan recognized that in order to investigate the global behaviour of the simulated container terminals, integration of the models was needed. The need for integration was the main motivation behind initiating a new project, called the FAMAS.MV2 Simulation Backbone Project, also called FAMAS.MV2 Project 0.2[38], for designing and developing a *simulation*

[38] The official web site of FAMAS.MV2 Project 0.2 – Simulation Backbone: http://www.famas.tudelft.nl

backbone which would support integration of simulation models designed and developed by different organizations within the FAMAS project.

The backbone project was initiated by different parties. We mention Connekt[39] as the main organization who initiated and sponsored the FAMAS.MV2 Simulation Backbone Project. In Appendix D a complete list of the participating organizations and their representatives is provided. Next, in the following sections, we present this project in more detail.

5.3 Analysis

System analysis is the part of the system development life cycle in which one determines how the current system functions and assesses what users would like to see in a new system (Hoffer, *et al.* 2002, pg. 202). The main objective of the analysis phase is to determine system requirements.

Once the FAMAS project management granted permission to start development of the simulation backbone, and once the project was initiated and planned, we begun to determine what requirements the backbone should fulfil. The requirements had to be in correspondence with the primary goal of the simulation backbone that, as presented in the previous section, is to integrate simulation models. The main requirements of the backbone can be deduced thus, from the nature of the projects for which it was initiated. Accordingly, the backbone should support reusability of already existing models, and the integration of models developed in heterogeneous COTS simulation packages. Since some of the participants had already been involved in distributed simulation projects, their own experience constituted the primary source for specifying the rest of the requirements. Additionally, literature study was another way to identify requirements (DMSO 1998a), (DMSO 1998b), (DMSO 1998c), (Fujimoto 2000), (Straßburger 2001), (Tanenbaum and Van Steen 2002), (Dahmann, *et al.* 1998), (Kuhl, *et al.* 1999), (McLean and Riddick 2000), (Brassé, *et al.* 1999), (Connekt 2001). The requirements were established in regular meetings between the participants.

The requirements that govern the nature, design and development of a simulation backbone can be viewed from different perspectives. We distinguish three perspectives based on (Verbraeck, *et al.* 2002):

- The perspective of the *users*, the users being the persons who develop simulation models and use the services provided by the simulation backbone.
- The perspective of *developers*, who are the persons who design and develop the backbone solution.
- The perspective of the *administrators*, who are responsible for the maintenance of the simulation backbone.

Next we present the requirements according to these three perspectives.

[39] http://www.connekt.nl/

5.3.1 Requirements from the user's perspective

The first group of requirements are the ones as viewed from the perspective of the end users. We refer to them as *U*-requirements. The end users are the simulation practitioners that **need to apply the simulation backbone** to integrate different models. Based on discussions with the users and the goals of the project we define the following requirements from this perspective:

U1. Quality.

The simulation backbone should support each partner within the FAMAS.MV2 project by providing a qualitatively satisfactory solution in the sense that it should provide extra functionality for simulation projects for an acceptable cost. The extra functionality might refer to the reuse of a model as a submodel in a distributed simulation project, to the integration of two models developed in different COTS simulation packages, to the integration of simulation models with external tools, and so on. The implementation of an extra functionality usually requires additional costs. However, in the case of non-existence of a backbone the cost can easily exceed the benefit that one can gain from. The cost-benefit ratio problem was discussed in more detail in Section 4.3.1.The cost required for extra functionalities varies from model to model, and project to project. The backbone should be designed in such a way that linking models developed in different simulation environments to the backbone structure should be as cheap as possible.

U2. Reusability.

Given that some simulation models already developed in earlier FAMAS projects needed to be reused in the projects within the FAMAS.MV2 research plan, integration of these already existing models with the new ones is required to be possible for an acceptable cost. The cost should be acceptable in the sense that it should be less than reimplementing the model from the scratch.

U3. Structure transparency.

Distributed systems are inherently complex. We have observed that most of the end users are not familiar with distributed architectures and modelling. Therefore, using the backbone should introduce as few distribution concepts as possible. The simulation backbone should, thus, be a transparent architecture in this sense for the end users in order to enable the modellers to integrate their simulation models.

U4. Support for heterogeneous COTS simulation packages.

Given that different organizations have been involved in the FAMAS.MV2 projects, applying different COTS simulation packages for designing and developing simulation models, the simulation backbone should provide support for integrating simulation models created in heterogeneous packages.

U5. Centralized view for conducting distributed simulation.

In contrast to running a monolithic simulation model on a single computer, the execution of a distributed simulation model is more intricate. Consequently, end users who conduct distributed simulation can confront with several additional issues,

that they did not face in the case of the monolithic case, such as the sequence of starting or terminating the distributed simulation execution, the way of controlling the simulation execution, the definition of a distributed scenario, and collection of data from distributed models. In order to let the end users easily tackle these problems a centralized view of the simulation runs should be provided.

U6. Acceptable speed of simulation runs.

Due to the information exchange that is required and the need for time synchronization, distributed simulation might perform slowly compared to monolithic simulation. This phenomenon especially occurs if the synchronization algorithms are inappropriately implemented or the network causes latencies. The backbone, therefore, should be a solution with an acceptable performance.

U7. Scalability.

The FAMAS Simulation Backbone should be scalable for any complex simulation project in the sense that it should allow the integration of a large number of simulation models without relative loss of speed and control.

5.3.2 Requirements from the developer's perspective

The second set of requirements is concerned from the developer's point of view, denoted as *D*-requirements. These are the ones that mainly deal with the functionality of the architecture of the simulation backbone and they concern mainly technical requirements.

D1. Service based architecture implemented in modular way.

Given the distributed nature of the system that needs to be modelled, the architecture of the simulation backbone should include the essential simulation services that are needed for interoperability between simulation models, such as data exchange and time synchronization. The implementation details of the distributed services provided by the simulation backbone should be isolated as much as possible from the simulation models that are to be integrated. For this reason, modular implementation of the services is appropriate. This requirement follows in fact from the requirement on transparency of the structure as required by users in order to not overburden them with notions from distributed systems. The modular structure of the architecture creates the possibility for the organizations to specialize in providing a specific distributed service, such that they can adapt and improve their own services without having to deal with the other ones.

D2. Efficient communication.

The different integrated components, such as simulation models implemented in various COTS simulation packages, real equipment, and software applications exchange data between each other during a distributed execution. Therefore, there is a need for a common communication protocol that enables efficient data exchange between these components.

D3. Generally applicable solution for interfacing with COTS packages.

Due to the fact that within the FAMAS.MV2 project different COTS simulation packages are considered for designing simulation models, the solution should be

general enough to easily allow for interfaces to be built in order to interface various COTS simulation packages with the simulation backbone. This solution should be designed in a way that it adequately hides the distributiveness from the end user.

D4. Support for debug facilities.

In order to verify whether simulation runs execute properly, concepts should be implemented facilitating this verification.

5.3.3 Requirements from the administrator's perspective

The third set of requirements is related to the administrator's perspective. We refer to them as *A*-requirements. The administrators are responsible for maintaining the simulation backbone. This group of people remains in contact with the end users, who provide feedback concerning the applicability of the backbone, for example with respect to changing a service, requiring a new service, or reporting bugs. In order to be able to appropriately react to such feedbacks, we have identified the following requirements from the administrator's perspective.

A1. Maintainability.

The FAMAS Simulation Backbone should be designed and developed in a way that if new services need to be added or adapted it should not influence the other services. That is, it should be possible to maintain certain parts of the backbone without reconsidering other parts. If maintenance of the services does not entail reconsideration of the structure of the backbone, the cost of maintenance involves only the cost of refinement, in the worst case replacement, of the services concerned.

A2. Extendibility.

The rapidly growing technology might lead to changes and new concepts within container terminals which might entail simulation models that require additional distributed simulation support for implementing these new concepts. The backbone, therefore, should be easily extendible in order to accommodate to such changes.

The ***FAMAS Simulation Backbone project*** was set up to design and implement a backbone that should meet the identified requirements. In the next section of this chapter we present the design phase.

5.4 Design

The simulation backbone is intended to be a solution to be applied to integrate several simulation models. As we presented in Chapter 2, solutions already exist for simulation model integration. Accordingly, the first issue we considered when designing the backbone was to identify the most appropriate distributed simulation architecture available. Since HLA implementations were already available at that time, and since HLA was considered the most advanced solution for distributed simulation, our choice naturally turned to consider an HLA RTI as the basis for the desired simulation backbone.

The participants conducted some analysis concerning the HLA implementation (DMSO RTI) and perceived that using an HLA RTI as a basis for the simulation backbone would make it difficult to fulfil the requirements generated at analysis phase. The HLA RTI provides a large amount of interface functions that simulation practitioners should consider in their model if they want to design and develop an HLA compliant distributed simulation model. The implementation of the HLA distributed services is hidden from the simulation practitioner: they are implemented in a compact application that encloses all the services and behaves like a heavy operating system. The complexity and the heaviness of the Run Time Infrastructure cannot be reduced because the source code is not freely available; it belongs to the US Department of Defence. The participants at that time thus came to the conclusion that although the HLA RTI intends to hide the distributiveness from the end user, that is the simulation practitioner, it does impose low level implementation tasks on them.

After investigating HLA and our requirements we decided that for our simulation backbone we needed a simpler and more flexible solution than an HLA RTI. Instead of having one compact application that includes all services, we aimed to design the backbone in a modular way, with a more lightweight structure than an HLA RTI. Consequently, as a solution we developed a new *architecture for a simulation backbone*, which is more lightweight and modular than HLA RTI. Since, as discussed in Chapter 2, the integrated models exchange data and since the primary aim of a distributed simulation architecture is to provide and support distributed simulation services, like time synchronization, the architecture of the backbone should be designed in such a way that it offers and supports a *standardized communication protocol* for data exchange and *distributed simulation services*. Additionally, the architecture should be able to interface in a natural way with COTS simulation packages in which most of the simulation models are designed and developed. In order to create an interface between COTS simulation packages and our simulation backbone we design a *middleware* based solution.

Next we elaborate on design issues presented above in several subsections. We start with focusing on the architecture in general, and then we turn to the design details regarding the communication protocol, the distributed services and the middleware.

5.4.1 Architecture Design

In order to integrate different components, such as simulation models, real equipment, controllers, and other software applications, one can apply advanced software engineering solutions from distributed system theory (Tanenbaum and Van Steen 2002). In contrast to a monolithic application, a distributed one requires certain distributed services needed for interoperation between distributed models. The services can be either part of the distributed components or they can be separated. If designed to be separated, the services can be implemented and offered by special so called *technical components*, contrasting there from simulation models, real equipment, controllers and other software applications which are referred to as *functional components.* Separating service components from functional components is beneficial because the designers of the functional components do not require additional knowledge to understand the distributed concepts that need to be included within the functional components. In light of the fact that one of the requirements of our simulation backbone is to hide the distributiveness from the end user and to provide a transparent structure, this separation provides the first step in this direction. In this manner, in contrast to the HLA standard which provides a compact RTI application that encloses all the services, in our backbone the services are designed and developed in a more modular view as easily exchangeable components.

The simulation backbone should contain technical components to support several standard distributed services. Since data exchange and time synchronization are basic requirements for distributed simulation (see Chapter 2), a well defined technical component should be responsible for exchanging data between the simulation models, while another technical component should provide time synchronization services. Additionally, a component is needed to handle the starting and stopping sequence of simulation models, and another one to monitor the distributed simulation execution.

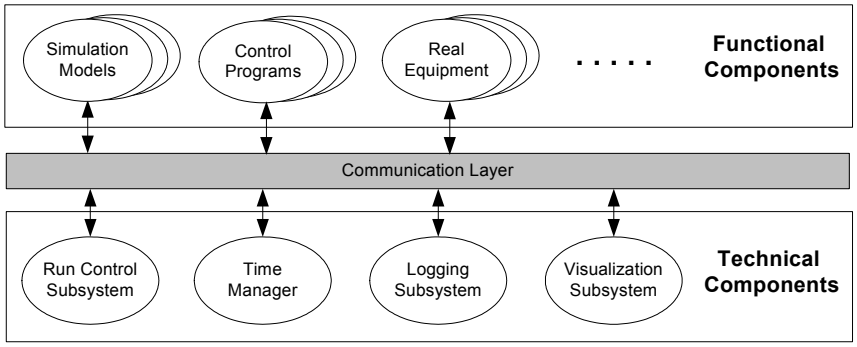


Figure 5.1 FAMAS Simulation Backbone Architecture (Boer, *et al.* 2002a)

In order to design the simulation backbone we introduce a distributed simulation architecture which is based on component oriented software design (Szyperski 1998), a new approach for representing distributed simulation services as technical components. According to this approach, Figure 5.1 depicts the design of the architecture for the simulation backbone, which is in line with the *D1* requirement. In the future we refer to this architecture as the FAMAS Simulation Backbone Architecture.

5.4 Design - Architecture Design

The FAMAS Simulation Backbone Architecture has both technical and functional components. The user defined components, such as simulation models, control algorithms, real equipment and real controllers, are the functional components, while the technical components provide common distributed services to the functional components. In HLA terminology, the overall system that consists of technical and functional components is called a *federation*, where the components that are connected to the backbone are *federates*.

Each technical component has its own special purpose. During the design phase we found the following technical components indispensable that should be implemented within the FAMAS Simulation Backbone Architecture.

- *Run Control*: responsible for overall control of experiments including scenario management; it starts, stops and monitors the simulation process.
- *Backbone Time Manager*: synchronizes simulation time among different federates.

For intended use of the backbone additionally two more technical components were identified:

- *Logging Component*: collects logging information from the federates into a central database for experimental analysis.
- *Visualization Component*: provides separated or common visualization views for the federates or the federation.

These technical components are elaborated in more detail in Section 5.4.3. Since the FAMAS Simulation Backbone Architecture has a component based structure, it provides new opportunities such as extendibility, replaceable feature and easy maintenance of the distributed services, which cannot straightforwardly be done within the HLA implementation[40]. Accordingly, it is not limited to technical components mentioned above: if future projects require it, the list can easily be extended with new technical components that provide additional services.

The separately defined technical and functional components give a modular structure to the architecture (Figure 5.1). Both the functional and the technical components can be easily replaced by other ones, for example a conservative time manager component can be replaced by an optimistic time manager component. This provides support for proper maintainability of the architecture. Furthermore the number of service components can be extended or reduced depending on the project requirements. For instance, a simple project might use only a small number of services that are provided by a minimal subset of technical components. The given maintainability and extendibility features are completely in consonance with the *A1* and *A2* requirements.

The technical and functional components communicate by means of *messages*. The communication protocol of the simulation backbone that supports this message exchange is discussed in the next subsection.

[40] Recently we have realized that there is one HLA implementation provided by MÄK Technology that provides extendibility of functionality to a certain extent (see Section 3.5.5.5).

5.4.2 Communication Protocol Design

The *D2* requirement from the analysis phase, illustrated by the architecture refers to the need for an efficient communication. The participants investigated various possibilities to design such a communication protocol for the backbone and identified the following three prerequisites that should be fulfilled:

1. The communication should be based on a *standard communication protocol.*
2. The frequency of communication over the backbone should be minimized.
3. The content of the exchanged messages should be *readable and interpretable* by simulation practitioners*,* for debugging and verification purposes.

Identification of a standard communication protocol

We chose the Transmission Control Protocol/Internet Protocol (TCP/IP) as the standard communication protocol for the FAMAS Simulation Backbone Architecture. TCP/IP is the basic communication protocol for the Internet (Comer 2000). Besides the fact that it is the most accepted standard communication protocol, some COTS simulation packages and most widely programming environments offer the possibility to send and receive messages according to this protocol. These two reasons motivated us to choose this protocol.

Peer-to-peer communication structure

TCP/IP uses the client/server model of communication, the server application providing services, such as processing database queries or sending states of the system, and the client application requesting and using the services provided by the server (Comer and Stevens 1997). Most business applications being written today apply this model of communication.

The TCP/IP protocol provides reliable peer-to-peer, or point-to-point, communication, which means that each communication is from one point, or host computer, in the network to another point or host computer. In order to communicate over TCP/IP, a client program and a server program establish a direct connection to each other, each program binding a socket to its end of the connection. When communicating the client and the server read from and write to the socket bound to the connection. Normally, a server runs on a specific computer and has a socket that is bound to a specific port number. The server just waits, listening to the socket, for a client to make a connection request (Comer and Stevens 1997).

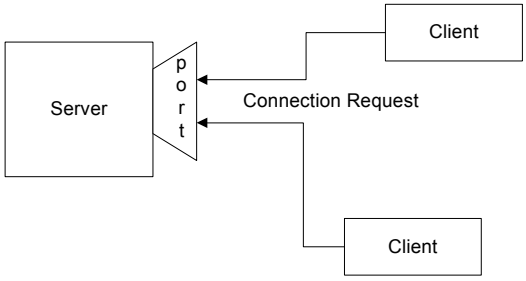


Figure 5.2 Setting up a Client Server Connection

As depicted on Figure 5.2, in order to establish a connection to the server, the client should know the hostname of the machine (IP address) on which the server is running and the port number to which the server is connected (Comer 2000). In order to make a connection request, the client tries to establish a rendezvous with the server on the server's address and port. On the server side, once the connection is accepted, the server acquires a new socket bound to a different port. It needs a new socket and consequently a different port number so that it can continue to listen to the original socket for connection requests while tending to the needs of the connected client (Boer*, et al.* 2002b). On the client side, if the connection is accepted, a socket is successfully created and the client can use this socket to communicate with the server. The socket on the client side is not bound to the port number used to rendezvous with the server. Rather, the client is assigned a port number locally to the machine on which the client is running to keep the original socket free for new call-ins from other clients (Figure 5.3).

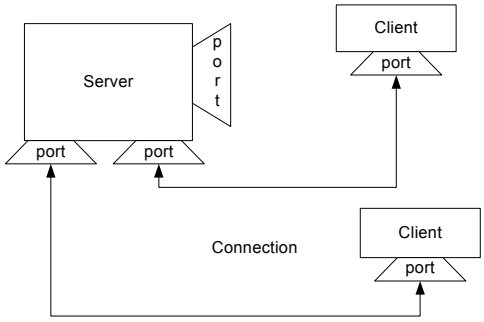


Figure 5.3 Established Connections between the Server and the Clients

The client and server can now communicate by writing to or reading from their sockets. So a socket is an endpoint of a two way communication link between two programs running on the network. A socket is bound to a port number so that the TCP handler on the computer can identify the application for which the data is meant.

In the backbone each component will have one server socket and a number of client sockets. In this way a direct peer to peer communication, between components is established which minimizes communication over the backbone compared for example to the approach that sends all information through a technical component that is responsible for data distribution, behaving like a switching board. This latter approach is used in HLA RTI's and, as a consequence, if the number of integrated components increases it can be easily overloaded. Due to the peer to peer communication the FAMAS Simulation Backbone Architecture eliminates this heavy approach, and provides a more *lightweight* solution. This proposed solution is in accordance with the *U6* and *U7* requirements.

In order to establish connections between technical, respectively functional components, a special technical component, namely the Run Control component has been designed. The Run Control provides help for other components to define their own unique server port, and to find IP addresses and ports of other components. The only information that each component should know, when coupling to the backbone, is the server socket and port number of the Run Control technical component. Therefore the Run Control component must be always started first when a distributed model is set up.

Readable message protocol

The TCP/IP is just a low level communication protocol that provides the possibility to send data from one model to another one. However, there is still need for agreement on the data that will be sent over the network and on its format. For this reason we have defined a protocol for the message structure on top of the TCP/IP protocol which results in messages that are readable and interpretable for simulation practitioners. According to the protocol each message must consists of four mandatory fields, separated by a forward slash. The standard format is:

```
Sender/Receiver/Message Type/Parameters/
```

- `Sender` is the name of the sender component, e.g. Simulation Model 1.
- `Receiver` is the name of the receiver, e.g. Simulation Model 2.
- `Message Type` is used to identify the message. The identifier is a name in a string format such as `AskJoinFederation` or `NextEvent`.
- `Parameters` is a list of parameters representing the content of the message. All parameters are separated by a forward slash.

A list of predefined messages can be found in Appendix E. This list contains 22 predefined messages, which are essential for the case studies discussed in this thesis. If it is necessary the message list can be easily extended by any user. With this possibility the FAMAS Backbone Architecture provides flexible support for interfacing different components. This solution has added value comparing to HLA, which provides more than 150 interface functions for communication. Furthermore, as we have stated before, we have chosen for a protocol of readable and interpretable messages. Alternatively, we could have chosen another protocol that handles unreadable binary formats or other readable formats, like XML, but for our purpose this format proved to be sufficient. The reason for choosing a readable format is that it enables the simulation practitioner to easily interpret the content of some messages during simulation run. Readable and interpretable messages can help the simulation practitioner to trace anomalies during interoperability, so it provides support for debug or verification activities, which is in line with the *D4* requirement.

5.4.3 Design of the Distributed Simulation Services

In this section we present the design of the technical components of the FAMAS Simulation Backbone Architecture, components that provide the distributed services required to accomplish the role of the backbone, as presented in the introduction part of this Section 5.4 and depicted in Figure 5.1. We identified four important technical components: Run Control, Time Manager, Logging and Visualization. Additionally, during the design phase, we realized that there is a need for a kind of Scenario Object that specifies the scenario for a distributed simulation execution. According to these five issues this section is split up in five subsections.

5.4.3.1 The Run Control Component

The *Run Control* technical component is the main controller within the FAMAS Simulation Backbone Architecture. As mentioned in the previous subsection and depicted in Figure 5.4 it communicates directly with all technical and functional components. Run Control is the only technical component that has direct access to the Scenario Object, that includes relevant information about the distributed execution, participating components, public and private variables, and so on. We elaborate in more detail on the role and design of the Scenario Object in Section 5.4.3.5.

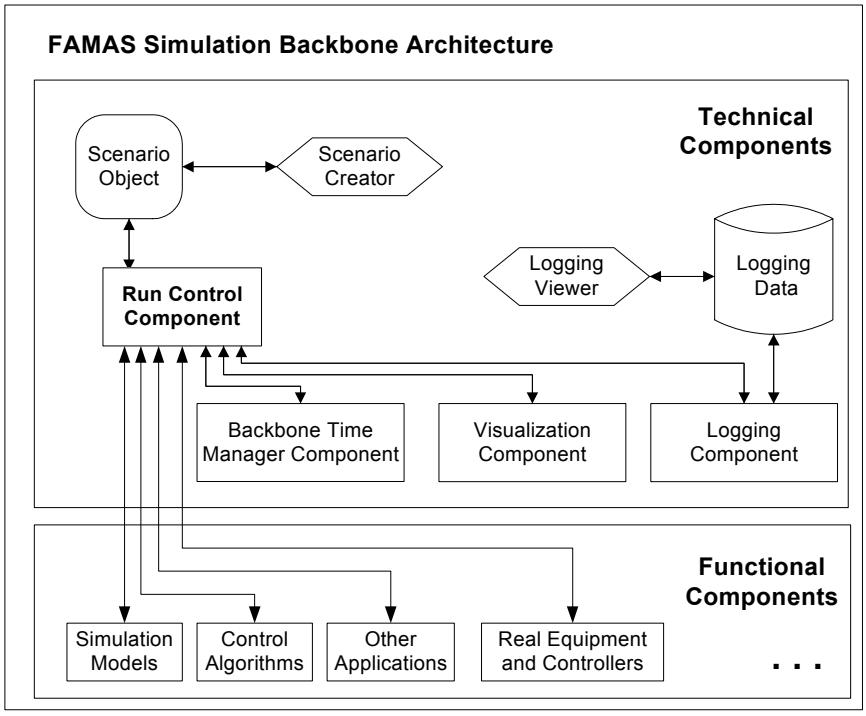


Figure 5.4 Central Role of Run Control within FAMAS Backbone Architecture

The Run Control component is responsible for three activities (Boer, et al. 2002c):

1. Initialization and start of a distributed simulation execution.
2. Special activities during distributed simulation execution.
3. Termination of a distributed simulation execution.

The first and the third activities are related to the basic activities of a distributed simulation architecture (see Chapter 2). Settling the start and termination mechanisms of distributed applications is, in fact, a significant issue not only in distributed simulation but also in general distributed software engineering (Tanenbaum and Van Steen 2002).

As the second point specifies, the Run Control component, besides these two basic activities, needs to perform additional special tasks as well. One of these activities is for example, to provide information to other components concerning the content of the Scenario Object, like the publicly available value of a variable used by a component.

Next we elaborate on activities performed by the Run Control component in more detail.

Initialization and start of a distributed simulation execution

In order to initiate a distributed simulation execution using the simulation backbone, we must start the Run Control technical component and specify the settings that need to be applied during actual execution. The settings refer to relevant information about the models which will participate in this distributed simulation. It contains information about the treatment of the simulation, as, for example, the simulation length. For an easier and flexible simulation set up we decide to specify the settings in a separate component, called the Scenario Object, a component on which we will elaborate later on in this subsection.

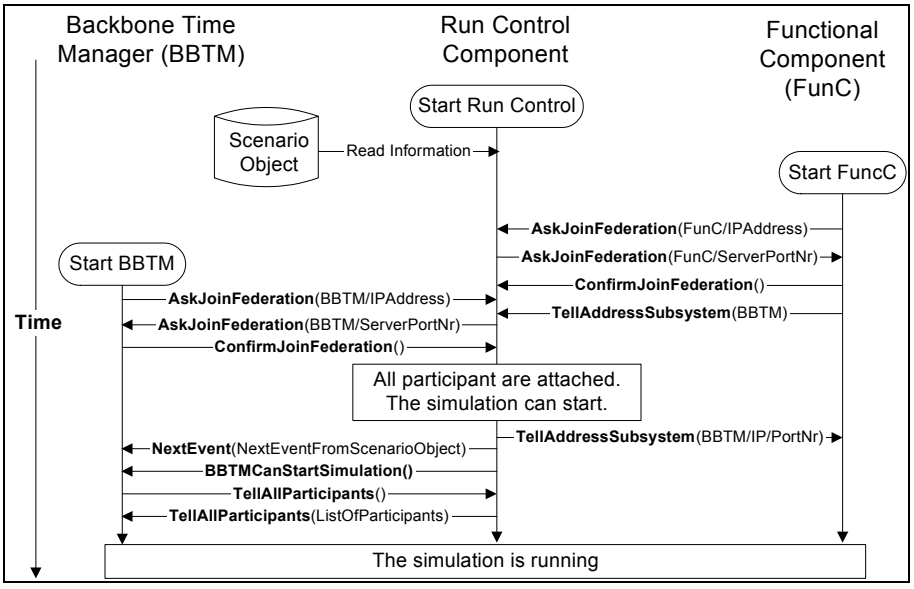


Figure 5.5 Initialization and Start of a Distributed Simulation

The example depicted in Figure 5.5 shows the starting sequence of a distributed simulation run with one functional component (FunC) and two technical components attached. The vertical line on the left hand side represents wall-clock time.

The first action is, thus, to start the Run Control technical component, which first reads the Scenario Object. Based on the scenario selected, the Run Control component waits for each specified participant to make contact, in the example above that is the FunC and the BBTM. FunC requests the Run Control component through an `AskJoinFederation` message permission to join the distributed simulation run. The reply of Run Control is affirmative and it provides a port number to FunC in order to set up a server socket. Consequently, from this point on, FunC can be reached by all other participants, because Run Control knows, registers and makes available its IP-address and port number. In the above example, besides FunC, a backbone time manager (BBTM) contacts the Run Control component as well. The task of BBTM, as we will discuss later, is to synchronize simulation time. In order to be able to start the distributed simulation, the BBTM needs from all participants the time of its next event. First Run Control sends a `NextEvent` message. The content of this message is an event time determined by the duration of simulation run, specified in Scenario Object. Next the Run Control component notifies BBTM through `BBTMCanStartSimulation` that the simulation can start. As a reaction to this message BBTM will ask Run Control a list of all participants to be able to check if they all have sent a `NextEvent` message.

Special activities during distributed simulation execution

In order to control the execution of a distributed simulation run the Run Control component is responsible for several activities. It is designed to carry out several special tasks, among which the most important ones are the following three:

- *Consistency checking*. The Run Control component periodically checks each participating component concerning their connection to the simulation backbone. Having joined a distributed simulation, none of the components can leave without informing the Run Control component. This is necessary because the components can communicate directly with each other and if one of them leaves without informing Run Control anomalies might occur. For example if a certain component continuously expects information from another one which beforehand was regularly provided and the provider component for some reason is not a participant anymore in the distributed execution a deadlock situation can occur. In this sense the Run Control should take care of consistency concerning the participating components, for example periodically sending “still alive” messages to the participants.
- *Information serving*: The Run Control needs to provide information about other components, such as the address of components and values of variables. Fort this reason, it reads the Scenario Object to get the required information.
- *State logging*. Additionally, in order to allow the user to investigate the communication between Run Control and participating components, for example to inspect the starting sequence of a distributed simulation run, the Run Control component has the task to send special state information about the distributed simulation execution to the Logging component.

Termination of a distributed simulation execution

Besides initialization and start of the simulation run, the Run Control is responsible for ending the simulation run. A simulation run can be terminated in two ways:

1. Based on the script in the scenario
2. Caused by a stop simulation message from an arbitrary component

Figure 5.6 presents the first alternative for stopping a distributed simulation. In this case, Run Control reads the stopping request script from the Scenario Object and sends it as a `NextEvent` message to the BBTM. When the event time is reached the Run Control component receives a notification event from the time manager which allows it to execute the task associated to the message. Then Run Control sends a `StopSimulationNow` message to BBTM, which then forwards this message to all participants except Run Control. After receiving this message, all participants are supposed to close down, including Run Control and BBTM.

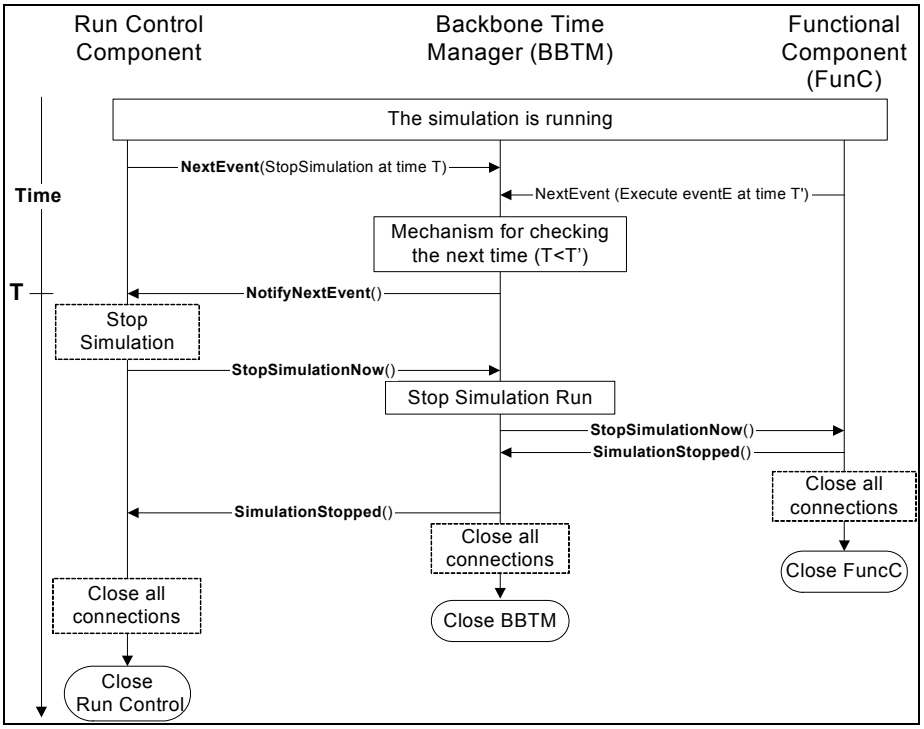


Figure 5.6 First Alternative for Stopping a Distributed Simulation Run

The second alternative for closing a distributed simulation run is depicted in Figure 5.7. The closing procedure is now initiated by an arbitrary functional component (FunC), by sending a `StopSimulation` message to Run Control. From that point onward, the process is analogous to the sequence described in Figure 5.6.

The three activities provided by the Run Control component, initialization and start, monitor, and termination of distributed simulation execution, play an important role to satisfy the *U5* requirement.

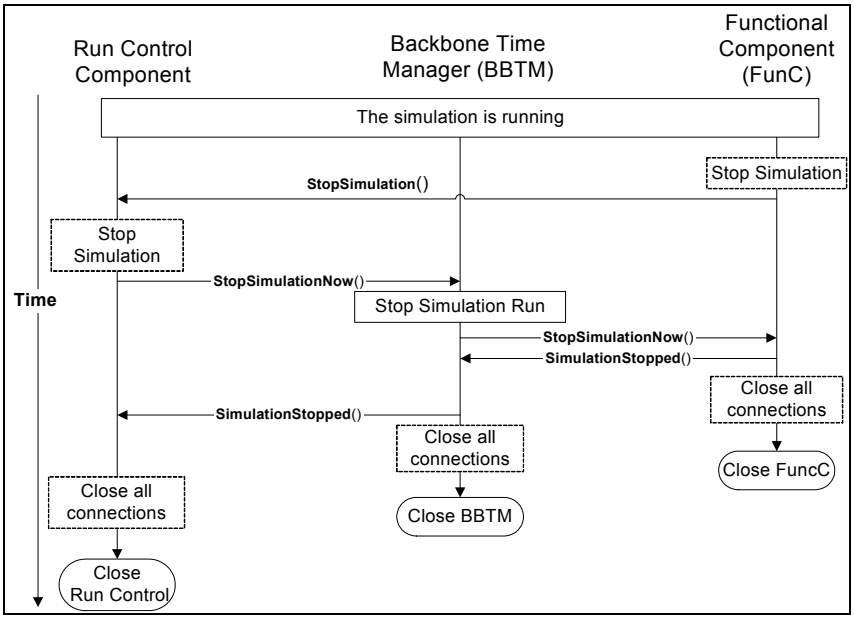


Figure 5.7 Second Alternative for Stopping a Distributed Simulation Run

5.4.3.2 The Backbone Time Manager

Time management plays an important role in distributed simulation. It aims to synchronize simulation time among different simulation components. The technical group was responsible to design and develop a component that accomplishes this time synchronization. The component that was designed for this purpose is called the *Backbone Time Manager (BBTM)*. It implements two types of time synchronization mechanisms, namely *conservative* and *real-time*. Conservative time synchronization is desired in order to achieve synchronization between discrete-event simulation models, while real-time synchronization aims to provide support for experiments when real equipment is involved.

All functional and technical components that are defined in the scenario of an experiment are considered participants and their first event must be scheduled on the time axis before a distributed simulation run can start. The basic principle for synchronizing the activities of the participants on the same time axis using a conservative mechanism is as follows. Each participant is assumed to send its first future event time, as a next event time stamp, to the backbone time manager as a `NextEvent` message. Then the time manager selects the participant with the smallest time stamp event and gives permission to perform this event by sending a `NotifyNextEvent` message. After completing the event, the participant sends its next future event time stamp to BBTM again. Participants sending the same event time are handled in first in first out (FIFO) sequence: the one who sent its event time first is allowed to proceed first.

In this way, during conservative time synchronization only one model is considered as "current" at any moment. This entails, however, performance inefficiencies during a simulation run. In contrast to conservative time synchronization, the optimistic version provides a more efficient solution. Fujimoto describes several algorithms that can be applied in order to accomplish optimistic time synchronization (Fujimoto 2000). Although optimistic time synchronization might provide a better performing solution, due to its complexity to implement it for currently available COTS simulation packages we did not consider it. Instead, as an alternative, the technical group decided to provide conservative time synchronization and improve the synchronization mechanism. To this end we intend to use several approaches that can optimize conservative synchronization. One of them is to minimize communication by offering each model a time horizon, lookahead value, and conditions under which it can act autonomously without consulting the backbone time manager.

An advantage of conservative time synchronization is that it provides support for full reproducibility that can be maintained in a given start up sequence (Veeke*, et al.* 2002). However, if COTS simulation packages mature to the level to support saving the state of a simulation execution at any simulation time than it might be possible to apply some rollback mechanism which is in fact a sort of optimistic time synchronization solution. The FAMAS Simulation Backbone Architecture, having a component based structure, is open to support additional technical components that are for example responsible for optimistic time synchronization.

Next to integrating simulation models, the FAMAS Simulation Backbone Architecture is aimed to support integration of real equipment and/or controllers. In the case of simulations which besides simulation models contain integrated real equipment and/or controllers, simulated time must be forced to advance in synchrony with wallclock time, otherwise the simulated virtual environment appears unrealistic (Fujimoto 2000). On the one hand if simulated time advances more slowly than wallclock time, the virtual environment would appear to be sluggish and unresponsive to user or real equipment actions, on the other hand if simulation time advances more rapidly than wallclock time, real participants would be at a disadvantage. When integrating real equipment, it is impossible to apply conservative synchronization as the time for real equipment (wallclock time) cannot be stopped. Simulation executions where advances in simulation time are paced by wallclock time are often referred to as *real-time executions,* and simulators designed to operate in this model are called *real-time simulators* (Fujimoto 2000, pg. 29). For these cases, the Time Manager provides a real-time synchronization mechanism. In order to design this mechanism we follow the idea of interval based synchronization, or as Fujimoto defines time stepped execution (Fujimoto 2000, pg. 31). In time stepped simulation, simulation time is subdivided in a sequence of equal sized time steps, and the simulation advances from one time step to the next. The simulation must control advances in simulation time to be in synchrony with wallclock time. The implementation of the time synchronization mechanisms of the Backbone Time Manager are presented in Section 5.5.1.

5.4.3.3 The Logging Component

In a distributed system where many components interact over the backbone, a data log can be used to check and analyze the performance and correctness of the system and subsystems. In a data log information can be recorded about the running system, such as output data, statistics, model errors, and warnings.

For logging data in an efficient way a separate *Logging Component* is designed which can be connected to the backbone. Although the aim of the Logging component is data collection in a common data store, components can store their information locally as well. We distinguish three types of mechanisms to be implemented by the Logging component:

1. *Log everything.* When this mechanism is used all available information about the component is logged during the simulation process. This way of logging results, however, in heavy network traffic.
2. *Log only relevant data.* This mechanism logs only selected data during the run. This mechanism results in less traffic, but forces the model builder to list the data needed in a detailed way.
3. *Log at the end*. Data is gathered by the Logging component only at the end of a run. This mechanism reduces network traffic the most. However, it has the disadvantage that in the meantime components must store logging information locally, and data might get lost if the model crashes before the simulation run terminates.

The data collected by the Logging component is stored in a relational database. We designed a relational database management system for which it is easy to write SQL queries to select and categorize the information stored.

The data sent by various components can be stored in one or more tables in the database. We distinguish two important messages that can be sent to the Logging component, namely `CreateTable` and `AddRecord`. When the Logging component receives a `CreateTable` message, it will create a table with a lay out (table name, field names, field types, etc.) specified by the sender component. All components in the FAMAS Simulation Backbone Architecture have access to this table in order to store the specified type of information. In order to insert data in the table, the components send an `AddRecord` message to the Logging component. During the simulation run there is a possibility to close data tables. In order to realize this we introduced the `CloseTable` message.

It is very important to take care that every component sends its information to the Logging component before it is closed and the simulation process ends. A well defined closing mechanism is designed as part of the FAMAS Simulation Backbone Architecture. As we have discussed before, at the end of the simulation run the Run Control is the last component that is closed down in the distributed simulation. The Logging component waits until the last moment for logging information. This means that it must close down immediately before the Run Control component.

The implementation of the Logging component, based on the design details elaborated above, is discussed in Section 5.5.1.

5.4.3.4 The Visualization Component

The aim of the *Visualization Component* is to present simulation results during and after the simulation run in the form of an animation. In this sense we can also refer to it as an animation component. This component should present as many instances of 2D or 3D viewing as necessary. There should be possibilities to animate the whole distributed system on one screen or just part of it on different screens.

The Visualization Component is designed in a way to present the modelled world as a fixed background having dynamic figures placed upon it. It is up to the participating components to fill the background and show the figures. In order to minimize communication, the graphical objects are inserted in the background only once. Defining objects in the background in this way is appropriate for non moving elements. In order to represent moving elements the Visualization component needs to collect the background objects and its coordinates, and after any change the screen that represents these moving objects needs to be refreshed. In order to show many types of shapes representing different simulation components of the project, various figures are defined within the visualization component. Shapes are specified as wire frames having their own origin. Specifications contain the coordinates and the planes of the shape in terms of these coordinates. In order to be able to handle the Visualization Component easily, a number of shapes can be predefined in template libraries. It is to be expected, that different FAMAS.MV2 projects will apply the same appearance for the same kind of equipments (e.g. AGV's, Carriers and quay cranes).

In Section 5.5.1 different camera snapshots of the implemented Visualization component are presented. Both the visualization component, that provides a centralized animation, and the logging component, that provides centralized data collection, are in consonance with the *U5* requirement.

5.4.3.5 The Scenario Object

As introduced in Section 5.4.3.1, a Scenario Object defines a simulation run of a distributed model. It is directly embedded in the Run Control component which in fact interprets the information provided by the Scenario Object. A Scenario Object has a unique identifier and consists of three parts:

1. *Variable Declaration Section*
2. *Initialization Script Section*
3. *Scenario Script Section*

In the Variable Declaration Section the values of the required parameters playing a role in the simulation run can be specified. Both 'global' variables that are used by more than one component and 'local' variables pertaining to one specific component can be included in this section. An entry can be either static or dynamic. A static variable cannot change during the simulation run, while a dynamic variable can be updated during the run, allowing other models to be informed about the new value.

The Initialization Script Section is introduced to define the 'set-up' of a simulation run, it identifies the technical and functional components to be used during a given specific experiment.

In the Scenario Script Section the way the simulation process should be executed can be specified. Here, events with a predefined simulation time can be defined, that the Run Control component should take care of. In this sense, Run Control behaves like a participating simulation model that has to react to events on the global event calendar.

The required information can be described using a special scripting language. The scripting language contains several types of commands. The command used to initialize variables, for example, has the following form:

```
[Component].Variable = NewValue
```

Further, there are special commands interpretable only by the Run Control component. For example the

```
WaitForConnect(Component)
```

is used to notify the Run Control component that a certain component intends to connect to the simulation backbone.

Table 5.1 illustrates an example of a Scenario Object. The implementation of the Scenario Creator, a tool that provides support to create such a scenario, is presented in Section 5.5.1.

Table 5.1 Example of a Scenario Object

	Name	Type		Value
[Scenario]	Name	String	Static	FAMAS.MV2 Demo 8 cranes
	Date	String	Static	07-04-2002
	Version	String	Static	2.1
	Author	String	Static	FAMAS Group
[StackSystem]	Length	Integer	Static	25
	Width	Integer	Static	6
	Height	Integer	Static	4
[QuaySystem]	NrCranes	Integer	Dynamic	8
[AGVSystem]	NrAGVs	Integer	Dynamic	40
	AGVSpeed	Real	Dynamic	3
[Logging]	LogMode	String	Dynamic	LessDetail
[Generic]	ModalSplit	File	Static	C:\Modalsplit.txt
	Weather	String	Dynamic	Sunny
[Initialization Script]	WaitForConnect(BBTM)			
	WaitForConnect(AGVSystem)			
	WaitForConnect(StackSystem)			
	WaitForConnect(QuaySystem)			
	WaitForConnect(Logging)			
[Scenario Script]	0	[Logging].StartLogging		
	3000	[Generic].Weather = Rainy		
	3001	[AGVSystem].NrAGVs = 30		
	3001	[AGVSystem].AGVSpeed = 2		
	5000	[Logging].LogMode = MoreDetail		
	10000	[RunControl].StopSimulation		

5.4.4 Middleware Design

In this section we focus on the design of the middleware that is needed to interface COTS simulation packages to the FAMAS Simulation Backbone Architecture as well as concepts related to it. The currently available COTS simulation packages mainly implement discrete-event simulation methodology (Zeigler, *et al.* 2000). Within this methodology there are various approaches applied for designing and developing simulation models, such as process oriented or object oriented approaches (Law and Kelton 2000). In order to formalize the concepts in such a way that it can be translated to all the COTS simulation packages we base the middleware design on generic abstract concepts instead of specific ones used in different approaches (Boer and Verbraeck 2003). For this reason, in the first part of this section, we introduce some formal description for interfacing simulation models created in COTS simulation packages. Then we extend these notions by analyzing more precisely how two COTS models can interoperate. Finally we discuss some inconsistency problems that might occur during distributed simulation run.

5.4.4.1 Interfacing COTS Simulation Models

This section will give an abstract mathematical treatment of the essential concepts behind data exchange between simulation models.

First let us introduce the following notation:

$M \triangleright P$ means a simulation model *M* is designed and developed in package *P*.

A simulation model manipulates entities, where entities are representation of real world objects (Zeigler, *et al.* 2000, pg. 482). Every time during simulation when a new entity is created it is associated with a unique identifier. In order to formalize this we introduce an abstract set $\mathcal{E}$ of entity identifiers. Now let *M* be a model. At each moment in time *t* the model maintains a collection of entities. We denote by $\tilde{E}_t$, where $\tilde{E}_t$ is a subset of $\mathcal{E}$, the set of entity identifiers corresponding to the entities existing at time *t* in model *M*. To each entity $E \in \tilde{E}_t$, a number *n(E)* of attributes is associated, numbered from 0 through *n(E)-1*.

Definition 1. Let *M* be a model, $\tilde{E}_t$ be the set of identifiers of entities existing at time *t*. Then the collection of all attribute identifiers at time *t* is given by

$$\tilde{A}_t = \left\{ \langle E, i \rangle \mid E \in \tilde{E}_t, 0 \le i \le n(E) - 1 \right\}$$

Relating this notion to Zeigler's DEVS formalism we observe that the set of entity identifiers together with the set of attribute identifiers and their values should typically be included in states as described in (Zeigler, *et al.* 2000). Notice, however, that the Zeigler state in general will include more information, like the system values, such as the simulation time or the contents of the event calendar, while here we only focus on entities and their attributes.

In order to achieve interoperability, these attributes (and their entities) must be accessible from outside the simulation package. Therefore we define an *access function,* as a characteristic function, that specifies whether an attribute is accessible or not.

Definition 2. Let *M* be a model, $\tilde{E}_t$ and $\tilde{A}_t$ the set of entity and attribute identifiers at time *t*. The *access function* $F_t : \tilde{A}_t \to \{0,1\}$ is defined by:

$$F_t(E,i) = \begin{cases} 0, & \textit{if the } i^{th} \textit{ attribute of } E \textit{ is not accessible at time } t \\ 1, & \textit{if the } i^{th} \textit{ attribute of } E \textit{ is accessible at time } t \end{cases}$$

Definition 3. Let *M* be a model, $\tilde{E}_t$ and $\tilde{A}_t$ the set of entity and attribute identifiers at time *t*, and let F_t be the access function. Then the collection of all accessible attributes at time *t* is given by:

$$\hat{A}_t = \{\langle E,i\rangle \mid F_t(E,i) = 1\}, \text{ where } \hat{A}_t \subset \tilde{A}_t$$

A simulation package can be *fully open*, *partly open* or *fully closed*. Let us define these concepts using the previous definitions.

Definition 4. A simulation package *P* is called *fully open* if $\forall M \triangleright P,\ \forall t,\ \hat{A}_t = \tilde{A}_t$.

Definition 5. A simulation package *P* is called *fully closed* if $\forall M \triangleright P,\ \forall t,\ \hat{A}_t = \emptyset$.

Definition 6. A simulation package *P* is called *partly open* if it is neither fully open nor fully closed.

Corollary. If $\exists M \triangleright P,\ \exists t,\ \hat{A}_t \neq \emptyset \ \wedge\ \exists M \triangleright P,\ \exists t,\ \tilde{A}_t \neq \hat{A}_t$ then simulation package *P* is *partly open*.

Notice that we have defined openness only in relation to entities and attributes. We did not discuss openness with respect to system values, such as simulation time or the content of the event calendar, leaving that for further research.

Most of the currently available COTS simulation packages are partly open in this sense. They do not allow direct access to the attributes, but offer interfaces through which some of the model attributes are accessible. These interfaces can be defined as a function for each simulation package.

At each moment in time *t*, each attribute of an existing entity *E* has a value. We denote the value of the i^{th} attribute by $a^i_{E,t}$. This notion is only valid for entity/attribute pairs that exist at time *t*, i.e. only if $\langle E,i\rangle \in \tilde{A}_t$.

We denote the set of all possible values of all possible attributes by *VAL*.

Definition 7. During execution of a model M at each moment in time t, the *interface function* $G_S : \mathcal{E} \times \mathbb{N} \to VAL$ is defined by:

$$G_S(E,i) = a^i_{E,t} \text{ provided } \langle E,i \rangle \in \hat{A}_t$$

The *interface function* and the *access function* are related. That is, if the attribute of an entity instance is not accessible then the interface function for that attribute of the entity is not defined. While the access function indicates whether an attribute is accessible, using the interface function we can access it.

In order to illustrate the previous concepts we take a simple example. To start with, we design and develop a model M in package P. Suppose we use three abstract entities: truck, generator and workstation. The model M at simulation time t has four entity instances (E_0, E_1, E_2 and E_3), namely two trucks (E_0 and E_1), a generator (E_2) and a workstation (E_3). The trucks are moving entities that after being generated move to the workstation. Table 5.2 gives the attribute values of the entities at time t.

The collection of entity identifiers is $\tilde{E}_t = \{E_0, E_1, E_2, E_3\}$.

The collection of attributes identifiers:

$$\tilde{A}_t = \{\langle E_0,0\rangle, \langle E_0,1\rangle, \langle E_0,2\rangle, \langle E_1,0\rangle, \langle E_1,1\rangle, \langle E_1,2\rangle, \langle E_2,0\rangle, \langle E_3,0\rangle\}$$

The attribute values for each entity are

$$\{a^0_{E_0,t}, a^1_{E_0,t}, a^2_{E_0,t}\},\ \{a^0_{E_1,t}, a^1_{E_1,t}, a^2_{E_1,t}\},\ \{a^0_{E_2,t}\}, \text{ and } \{a^0_{E_3,t}\}.$$

Table 5.2 Attribute Values at Simulation Time *t*

Entity Instances	Attribute
E_0 (truck)	$a^0_{E_0,t}$ (velocity of the truck)
	$a^1_{E_0,t}$ (type of the truck)
	$a^2_{E_0,t}$ (info about the shipment)
E_1 (truck)	$a^0_{E_1,t}$ (velocity of the truck)
	$a^1_{E_1,t}$ (type of the truck)
	$a^2_{E_1,t}$ (info about the shipment)
E_2 (generator)	$a^0_{E_2,t}$ (average inter-arrival time)
E_3 (workstation)	$a^0_{E_3,t}$ (processing time)

Suppose only the trucks can be accessed from the outside world. This can be described by specifying the access functions as

$$F_t(E_0,0) = F_t(E_0,1) = F_t(E_0,2) = 1,$$
$$F_t(E_1,0) = F_t(E_1,1) = F_t(E_1,2) = 1,\ F_t(E_2,0) = 0 \text{ and}$$
$$F_t(E_3,0) = 0.$$

Applying this function we obtain

$$\hat{A}_t = \{\langle E_0,0\rangle, \langle E_0,1\rangle, \langle E_0,2\rangle, \langle E_1,0\rangle, \langle E_1,1\rangle, \langle E_1,2\rangle\},$$

i.e. the accessible attributes are: $\langle E_0,i\rangle$ and $\langle E_1,i\rangle$ $(i \in 0,1,2)$.

As we can see $\hat{A}_t \neq \emptyset$ and $\tilde{A}_t \neq \hat{A}_t$ which implies partly openness. The corresponding interface function G_S is given by:

$$G_S(E_0,0) = a^0_{E_0,t},\ G_S(E_0,1) = a^1_{E_0,t},\ G_S(E_0,2) = a^2_{E_0,t}$$

$$G_S(E_1,0) = a^0_{E_1,t},\ G_S(E_1,1) = a^1_{E_1,t},\ G_S(E_1,2) = a^2_{E_1,t}$$

The G_S function might be implemented in the following way

$$G_S(k,l) = getEntity(k).Attribute(l)$$

For example, for all truck entities we obtain

$$\begin{cases} getTruck(k).Velocity = G_S(E_k,0) = a^0_{E_k,t} \\ getTruck(k).Type = G_S(E_k,1) = a^1_{E_k,t} \\ getTruck(k).InfoShipment = G_S(E_k,2) = a^2_{E_k,t} \end{cases}$$

When an entity is created a unique identifier (say a number *k,* in this case $\varepsilon = \mathbb{N}$) is assigned to it. Using this *k* number and the interface function we can access the attribute values of the entity. Other models that want to use these entities must know these unique identifiers. In a distributed simulation study a distributed simulation architecture can provide a mechanism (e.g. publish / subscribe) for informing the other models about updates of relevant information (Kuhl*, et al.* 1999).

In order to make the COTS simulation packages suitable for distributed simulation the vendors should make them as open as possible by enlarging the *collection of all accessible attributes*. Furthermore, for all these accessible attributes a large variety of interface functions should be provided.

5.4.4.2 Interoperability between COTS Simulation Models

Simulation models designed and developed in COTS simulation packages cannot achieve *direct* interoperation with other COTS simulation models. Therefore for packages that are open or partly open so called *middleware* must be developed that makes interoperability with other models possible. The concept of middleware is not new, the software engineering domain already recognized its benefits. Middleware is a general term for any software that serves to glue together or mediate between two separate and often already existing applications (Linthicum 1999). Middleware provides a two way interaction

1. Interaction with its own COTS simulation model implementing the interface function for accessing internal data (e.g. entity instances and their attribute values, simulation time, next event in the event calendar, etc.)
2. Interaction with other models or middleware of these models through a distributed simulation architecture implementing (its part of) the interoperability function.

Most of the COTS simulation packages enable the modeller to define such middleware for the simulation model. Figure 5.8 depicts an architecture where two models are connected through their middleware to a distributed simulation architecture. Interoperability itself between the simulation models is achieved by applying a distributed simulation architecture, like HLA (DMSO 1998a), (DMSO 1998b), (DMSO 1998c) or the FAMAS Simulation Backbone (Boer*, et al.* 2002b), (Veeke*, et al.* 2002). The distributed simulation architecture provides the interoperability functions for the simulation middleware.

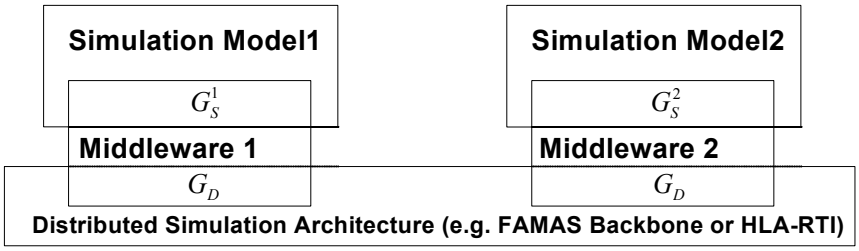


Figure 5.8 Connection of Two COTS Simulation Models

We can formalize the middleware as a set of interface function G_S (one for each model) offered by the COTS simulation package together with an interoperability function G_D offered by the distributed simulation architecture. Since the G_S 's are defined using COTS simulation package concepts, they are simulation package dependent. Similarly, the G_D interoperability function is dependent on the distributed simulation architecture.

We now proceed to give a formal definition of the interoperability function G_D. We want to describe the situation where we have a collection of simulation models operating in parallel, each model having a unique name M_i. To formalize this we introduce the abstract set M of model names, we postulate $M_i \in \mathsf{M}$ for each i, and we define $\mathfrak{M} = \{M_i\}_{i \in I}$, where I is some index set.

At each moment in time *t* each model M_i maintains a set of entities, and associated with each entity E there is a set of attribute identifiers $\langle E, i \rangle$ with corresponding attribute value $a^i_{E,t}$. Furthermore for each model M_i the interface function G^i_S is defined. Globally, i.e. within collection $\mathfrak{M}$, this leads to a set of *global entity identifiers* of the form $\langle M_i, E \rangle$, where E is an entity existing in model M_i at time *t*. This also entails a set of *global attribute identifiers* of the form $\langle M_i, E, j \rangle$, where $\langle E, j \rangle$ is an identifier of an attribute existing in model M_i at time *t* $(M_i \in \mathsf{M}, E \in \mathcal{E})$.

Based on this representation we can now define the interoperability function G_D.

Definition 8. For each collection $\mathfrak{M} = \{M_i\}_{i \in I}$ of model names and for each moment at time *t*, the interoperability function $G_D : \mathsf{M} \times \mathcal{E} \times \mathbb{N} \rightarrow VAL$ is defined by

$$G_D(M_i, E, j) = G^i_S(E, j) = a^j_{E,t}$$

Provided $M_i \in \mathsf{M}$ and $\langle E, j \rangle \in \hat{A}_t$. Here $\hat{A}_t$ and G^i_S are as defined for model M_i.

Thus G_D defines the values of all accessible attributes in all models. The function of the external middleware is to implement this function by rephrasing, so to speak, an interface function in terms of the interoperability function.

If interoperability between submodels is in order, for instance when entities need to be transferred between the submodels, the interoperability function G_D should be applied. Through this function the receiving submodel can obtain the parameters of the entity and use these parameters to instantiate this entity for itself. Thus submodels need to access (relevant parts of) the interoperability function. This can be realized through message passing implementing for instance the publish/subscribe mechanism.

In order to illustrate this let us take a very simple example, where a model transfers an entity instance to another model (e.g., a truck instance is transferred from one model to another one, as shown in Figure 5.9).

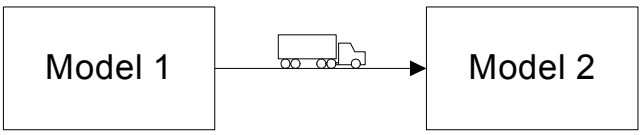


Figure 5.9 Model 1 Transfers a Truck Entity to Model 2

The first model registers the truck entity through the distributed simulation architecture making it visible for the other model. The other model can subscribe to the entity object if

it is interested. When the entity instance must be transferred to the second model, the first model publishes the entity to the second one. This phase is called updating.

The implementation of the G_D function can be done by sending messages like:

$$registerTruckVelocity(k,0)$$

$$registerTruckType(k,1)$$

$$subscribeTruckVelocity(k,0)$$

$$publishTruckVelocity(k,0)$$

$$updateTruckInfoShipment(k,2)$$

Here *k* is the identifier of the truck entity involved, and the second parameter (0, 1 or 2) is the attribute number.

Regarding the implementation of the middleware, for each partly open COTS simulation model we specify two kinds of middleware: *internal* and *external middleware* (Figure 5.10).

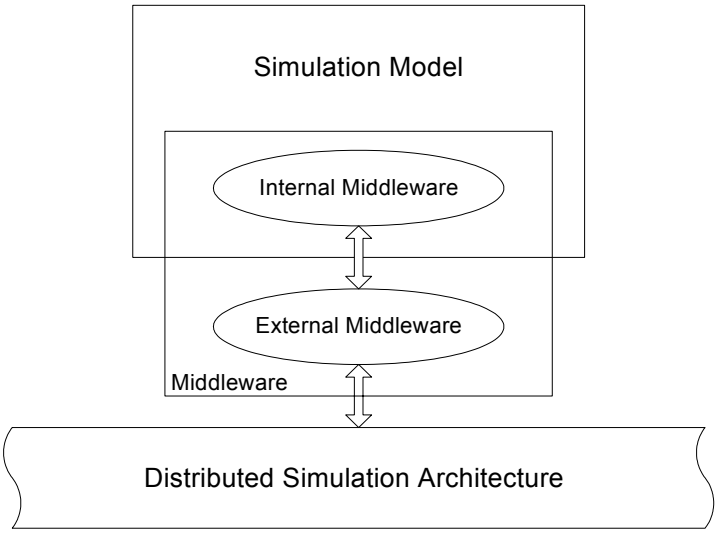


Figure 5.10 Internal and External Middleware

The internal middleware is COTS simulation package specific. It has the responsibility to interoperate with the outside world through the external middleware. The external middleware can be a Dynamic Link Library (DLL) if the package allows such a possibility. Usually the external middleware remains the same for models that are developed in the same package, however for the internal middleware a slight modification is needed depending on the model.

If the simulation model is designed in a programming language generally there is no need for middleware. The current COTS simulation packages do not allow direct access from

the internal middleware to the distributed simulation architecture. This can be achieved only through external middleware. If simulation packages would offer possibilities through which direct connection to the architecture could be specified, the need for an external middleware would be eliminated and a more flexible and easier connection to the distributed simulation backbone could be realized. The internal and external middleware provide support to the *U3*, *U4* and *D3* requirements.

5.4.4.3 Inconsistency Problems during a Distributed Simulation Run

As we stated before, when two simulation models interact they might need to transfer a simulation entity from one model to the other one. If an entity instance is created in a model and is transferred to another one, the receiver model must support the instantiation of the type of the transferred entity. Figure 5.9 depicted two models, where the first model (sender) generates truck entities and transfers them to the second model (receiver). Let us represent the abstract truck entities as $\tilde{E}_{Truck}$. At simulation time *t* the sender creates a truck entity instance $E_i \in \tilde{E}_{Truck}$ that is transferred to the second model. Due to the fact that the truck entity is transferred to the second model, both models should provide the possibility to instantiate the $\tilde{E}_{Truck}$ abstract entity. Unfortunately, in most of the cases this solution is not supported.

If the simulation packages work with a similar set of entities, then both the sender and receiver can instantiate the same type of entity (e.g. a truck entity). However, if the set of entities are different, instantiation is difficult if not impossible. In some of the cases this problem can be solved using syntactical analyzers that check the definitions of the entities. For example, in the sender model the abstract entity of the transferred entity is a truck entity $\tilde{E}_{Truck}$. The receiver model might not have a truck entity, but it might contain an abstract lorry entity $\tilde{E}_{Lorry}$. The truck and lorry entities are basically the same (described by the same attributes), but they are defined by a different name.

The aim of the syntactical analyzers is to find syntactical inconsistencies and to discover a possible matching between different types of entities (e.g. $\tilde{E}_{Truck}$ and $\tilde{E}_{Lorry}$). When the receiver model cannot instantiate any transportation type entity (e.g. $\tilde{E}_{Truck}$ or $\tilde{E}_{Lorry}$) then a transfer of this kind of entities cannot be realized. Basically in this situation the receiver is not allowed to subscribe to a transfer of any transportation entity.

Further, in some of the cases the abstract entity names are the same but they define different attribute sets (e.g. in the second row in Table 5.3). For example, both the first and second models can instantiate an $\tilde{E}_{truck}$ abstract entity, but in the second model the $\tilde{E}_{truck}$ does not have an attribute describing its shipment. Semantic analyzers can be applied in order to tackle this problem.

Table 5.3 Entity and Attributes Relations

Set of Entity	same	different	missing
Set of Attributes	same	different	missing
Type of Attributes	same	different	-

Moreover, if the set of entities and the set of attributes are the same we might still be confronted with the problem that the type of the attributes differs (third row in Table 5.3). The last part of this section describes an approach for solving this situation.

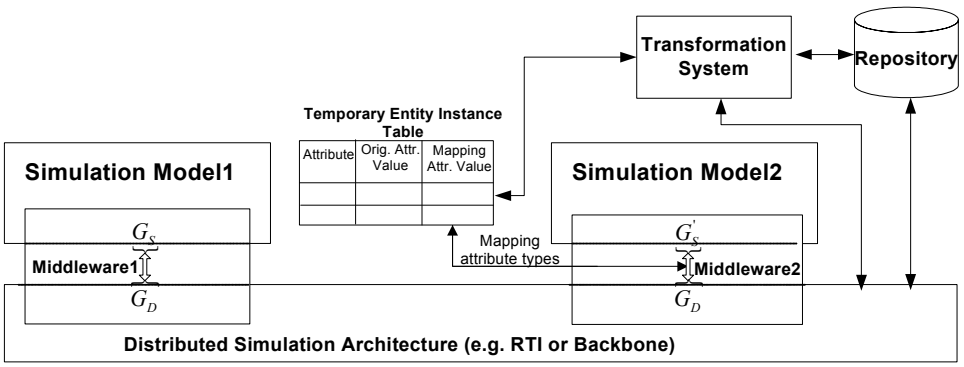


Figure 5.11 Architecture for Attribute Type Inconsistency.

The architecture of Figure 5.11 depicts two simulation models that are connected to each other through a distributed simulation architecture. The first simulation model transfers an entity to the second simulation model. The second model is able to instantiate the same abstract entity but some of the attribute types of the instantiated transferred entity differ. In such a case the middleware of the second model creates a temporary entity instance table and maps the original attributes of the transferred entity on attributes that the second model supports. Let us take a simple example in order to show this mechanism. The example is depicted in Figure 5.12, namely the transfer of a truck entity from the first model (M_1) to the second model (M_2) and then back to the first model (M_1). Suppose that $M_1 \triangleright P_1$ (simulation model M_1 is designed and developed in package P_1), and $M_2 \triangleright P_2$ (simulation model M_2 is designed and developed in package P_2). Simulation package P_1 supports all attribute types, such as string, real, integer, etc. for interoperation (e.g. eM-Plant), but P_2 supports only the type real (e.g. Arena). Both the P_1 and P_2 package can instantiate a truck entity. The set of the attributes of the truck entity are the same (velocity, type, shipment), but some of them differ in their types.

Suppose that the truck entity is created in the first simulation model with attribute values Velocity = 80, Type = 'carrier' and Info Shipment = empty (see Figure 5.13). At a certain point in time the entity leaves the first simulation model and is transferred to the second model. The second model is limited compared to the first one in the sense that it can only interoperate with numbers. However, there is a trick enabling us to represent the type and info shipment using numbers. The middleware of the receiver model creates a temporal entity instance table, where it maps the original attribute values to numbers (e.g. for the second model the attribute value 'carrier' is coded as a '1' and the fact that the truck is 'empty' is coded by giving the attribute info shipment the value '0').

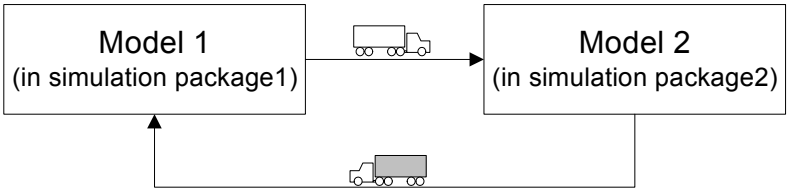


Figure 5.12 Circular Entity Transfer

After the truck entity is instantiated in the second model, it will go through some modifications, for instance the shipment info attribute of the truck is changed. The truck is not empty anymore, it will carry a container. This action is modelled by updating the information about the truck shipment in the entity instance table. For the second model if the truck is filled with a container then the value of the attribute shipment info is updated from '0' to '12'. After some time the loaded truck is transferred back to the first simulation model. The middleware again uses a temporal entity instance table to convert the numbers to their original types. Figure 5.13 gives a picture of this mechanism.

For some COTS simulation packages it is only possible to expose internal (attribute) values to the outside world as values of one type (e.g. integer) but on the other hand inside the simulation model more then a single type can be handled. For example Arena (Kelton, *et al.* 2003) can easily expose integer numbers (through the EVENT block) but inside the package we can use other types as well. When connecting to other packages we can overcome this restriction using temporal entity instance tables.

The next section describes an implementation of middleware for two COTS simulation packages along the lines sketched above.

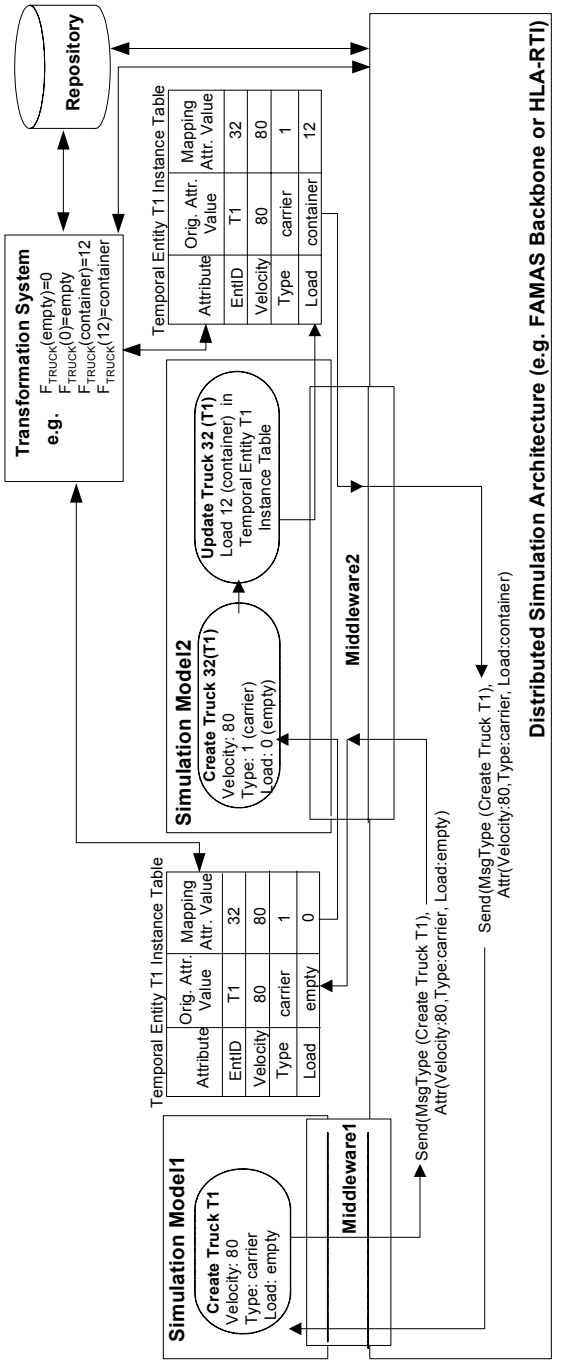


Figure 5.13 Entity Transfer with Inconsistent Attribute Types

5.5 Implementation

In most of the cases the implementation phase of the system development life cycle is the most expensive and time consuming phase of the entire life cycle. It is expensive because so many people are involved in the process and it is time consuming because there is much work to do. The purpose of the implementation phase is first of all to build a properly working system according to the design, then test it and install it within the organization. Implementation includes coding, the creation of user documentation, training users, and preparing a support system to assist users (Hoffer*, et al.* 2002, pg. 568).

Coding is the process whereby the design specifications discussed above has been turned into computer code by the technical team. During the project conducted, once coding had started, the testing process also begun and was conducted in parallel with coding. During the testing phase we first tested the different technical components individually and then, when they were integrated, as part of the larger system. Installation, as described by Hoffer, is the process during which the current system is replaced by the new system. Due to the fact that an earlier backbone did not exist, we did not replace it, instead, we just configured the new one to be able to function. Projects always require deliverables. One kind of deliverable that the FAMAS project required was documentation. The FAMAS project has resulted in two reports, a functional and a technical one (Boer*, et al.* 2002b), (Veeke*, et al.* 2002). Furthermore this research entailed several scientific publications (Boer and Verbraeck 2002), (Boer*, et al.* 2002a), (Boer*, et al.* 2002c),(Boer*, et al.* 2002d) (Veeke*, et al.* 2002a) and (Boer and Verbraeck 2003). As regards training, we devised a training plan which was a strategy for training users so they can quickly learn the new system. The project did not explicitly require a deliverable for this purpose, instead for training reasons one could use the documents mentioned above. The project generated some support systems as well, the main reason of which was to provide help to the user who intended to apply the FAMAS Simulation Backbone Architecture. One of these support systems is the Scenario Creator for creating Scenario Objects for the various experiments which will be presented later in this section.

In this section we focus on the main issues concerning the coding process of the different components of the Simulation Backbone Architecture.

5.5.1 Implementation of the Technical Components

We discussed the design of the technical components in Section 5.4.3. Next we will elaborate briefly, in the same order, on the main coding issues concerning these components.

5.5.1.1 The Run Control Component

The Run Control component is the technical component that, as we indicated discussing the design phase, needs to be started first during the distributed simulation execution. As we described before, it is responsible for several tasks, such as starting, terminating and monitoring the distributed simulation run. After being launched, the Run Control component needs to interpret a Scenario Object which specifies the scenario of the distributed simulation to be executed.

5.5 Implementation

The Run Control technical component is implemented in Java 2 SDK Enterprise Edition[41] and its implementation is realized as a Client/Server Multi-Threaded architecture (Beveridge and Wiener 1997). The multi-threaded structure has been straightforwardly used to implement the communication design presented in Section 5.4.2. As a result, when communication between two components is needed, each component starts a separate thread to communicate with the other one.

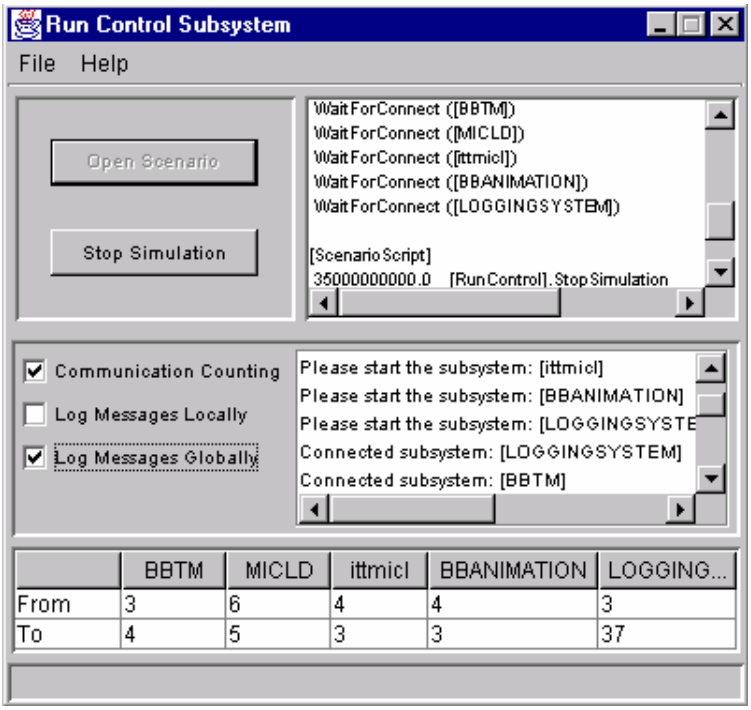


Figure 5.14 GUI of the Run Control Technical Component

In order to provide a user friendly interface, we designed a graphical user interface (GUI) depicted in Figure 5.14, through which users can manage the Run Control component. The interface allows the user to initiate a distributed simulation run by selecting a Scenario Object, and to suspend or terminate a distributed simulation run. Further it provides the possibility to see the contents of the incoming and outgoing messages. Here we profited from the fact that the exchanged messages are readable and interpretable for the end users. Besides the contents, the user can observe the frequency of the exchanged messages as well. Additionally, the end users can determine using this interface whether the messages are stored in local or in global storage.

[41] For more info see http://java.sun.com/

5.5.1.2 The Backbone Time Manager

According to the design plans the second technical component, the Backbone Time Manager (BBTM), is responsible for time synchronization between the integrated simulation models. Two variants of the BBTM exist. One variant is implemented in Borland Delphi at Delft University and another version is implemented in Java 2 SDK Enterprise Edition at Erasmus University. They have the same interface to the simulation packages and use the same messages, however, their internal implementation and graphical user interface differs.

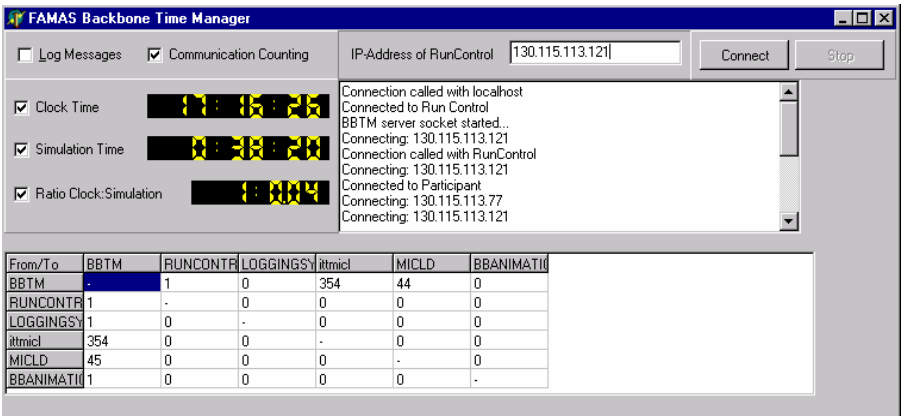


Figure 5.15 GUI of the Delphi Implementation of Backbone Time Manager

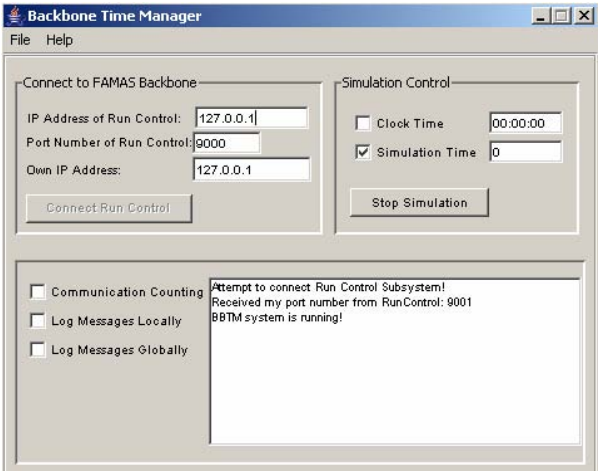


Figure 5.16 GUI of the Java Implementation of Backbone Time Manager

Figure 5.15 depicts the GUI of the Delphi version of the BBTM, while Figure 5.16 depicts the GUI of the Java version. Their user interface contains more or less the same elements. Users interact with the functionality provided by the BBTM through this interface. Using both variants the users can make a connection to the Run Control component, monitor the

activities of the BBTM during a distributed simulation run in a readable form, and additionally, visualize simulation time, wall-clock time (real-time) and the ratio between these two.

Because the distributed simulation architecture is fully component based, in the backbone technical components can be replaced by other ones that have the same role but are implemented in a different way. Replacement can be done, for example, to improve performance. The fact that there exist two implementations of the BBTM shows the extendibility and replaceable feature of the FAMAS Simulation Backbone Architecture. This is a feature of the FAMAS Backbone that High Level Architecture lacks.

5.5.1.3 The Logging Component

The aim of the Logging component, as indicated in the design, is to collect information from distributed models and store it in a central place, which is a relational database. The logging component has been implemented in Java 2 JDK Enterprise Edition, in order to link it to a relational database we used Java Database Connectivity (JDBC), an application program interface (API) for connecting applications developed in Java to many commercially available database systems.

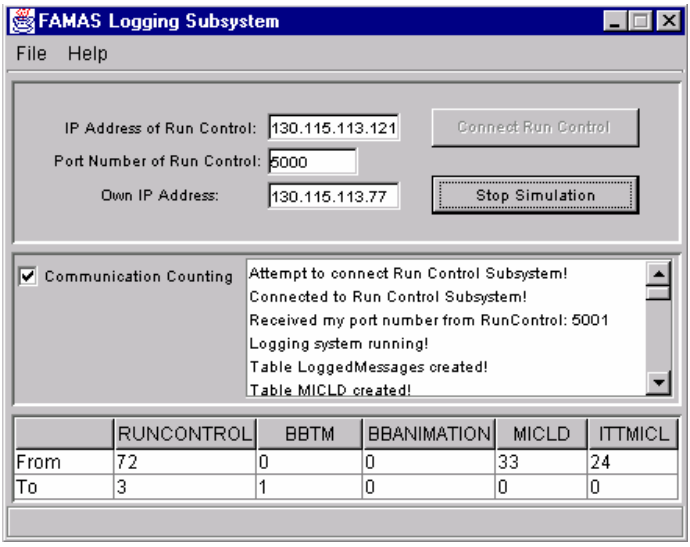


Figure 5.17 GUI of the Logging Component

The Logging component has a graphical user interface as well, through which the users deal with the Logging component (Figure 5.17). Through this interface users can connect to the Run Control component, indicate where the collected data should be stored, and visualize the collected data during the simulation run. Furthermore, through this interface the user can monitor the frequency of incoming and outgoing messages. It might occur that a certain component has several incoming messages in a given time period, and as a consequence this component frequently sends data to global storage. Such a high frequency might reduce the performance of the system. Accordingly, to avoid this situation, as we discussed in the design phase, instead of logging everything, users can choose

through this interface to log only relevant data or to log the data at the end. Supervising the content, respectively the frequency of the messages can help users during the verification process.

In order to provide the possibility for the end users to see the collected data in a structured way, we have additionally designed and developed the FAMAS Logging Viewer component. Figure 5.18 shows the GUI of the viewer application, implemented in Visual Basic, that enables end users to investigate the collected data in a structured way both during and after simulation run.

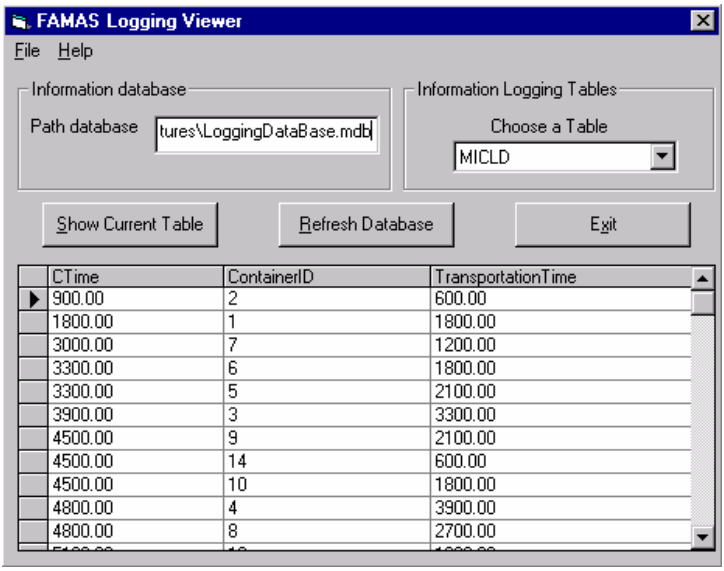


Figure 5.18 GUI of the Logging Viewer

5.5.1.4 The Visualization Component

As described in Section 5.4.3.4, the Visualization component is designed with the aim to animate the activities of the modelled systems during the simulation run. An animation can be an effective way to detect invalid model assumptions and to enhance the credibility of a simulation model (Law and Kelton 2000).

The interface of the Visualization component includes a 3D animation window which behaves like a camera. During simulation the camera can be moved interactively using the buttons at the top of the animation screen. The user can move the camera in three dimensions and also turn it in all three directions. The Visualization component is implemented to be as general as possible, it is intended to wait for commands to show figures in a two or three dimensional world.

A template file specifies which shapes must be used during the simulation run. At the start of a simulation the template file must be loaded and from that moment on all components can request the Visualization component for a figure corresponding to a specific shape.

5.5 Implementation

In order to be able to visualize a figure, three steps must be taken (Boer, *et al.* 2002b):

- Define its shape
- Specify its scale and render mode
- Specify its position, orientation and colour.

Each shape can be stored in a text file and the file can be added to a template file. The Visualization component can show the figure by specifying its position and orientation, by means of a `ShowFigure` message that looks like `ShowFigure/FigureID/X/Y/Z/αX/αY/αZ`

Several Visualization components can be used to animate a simulation model from several angles, as if the user manages multiple video cameras. Figure 5.19 and Figure 5.20 present different views of the map of the port of Rotterdam using two Visualization components, each one presenting the port activities from a different angle.

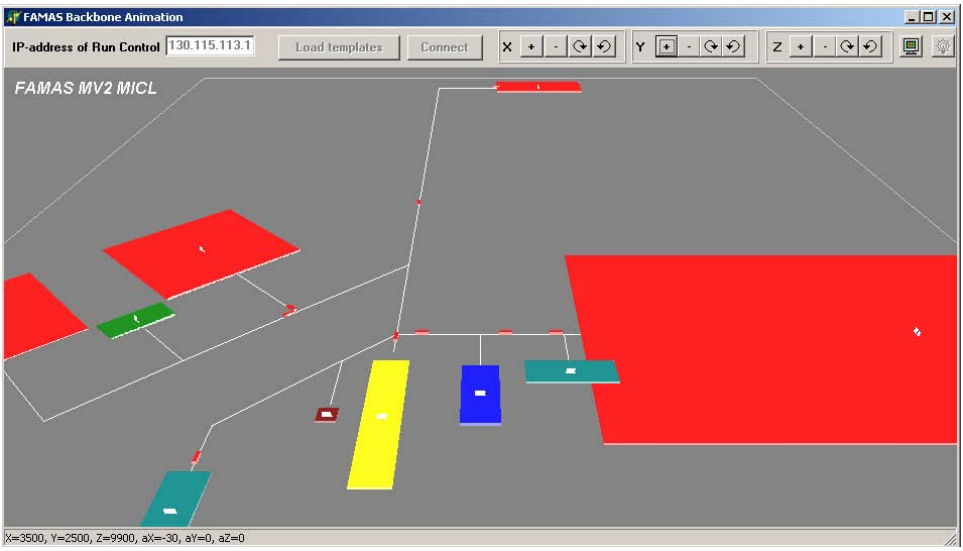


Figure 5.19 Screenshot of a View on the Map of the Port of Rotterdam

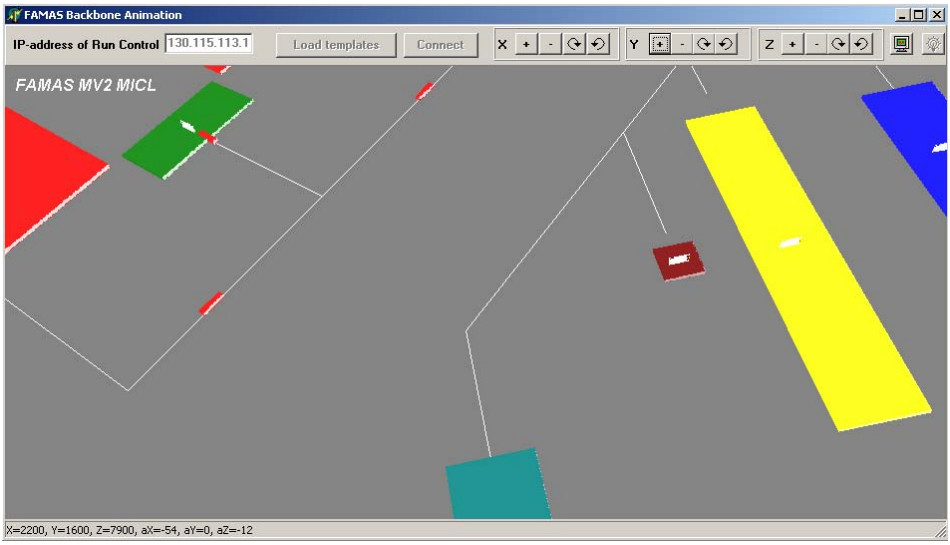


Figure 5.20 Screenshot of an Alternative View on the Map of the Port of Rotterdam

Besides the global animation offered by the Visualization component, there is a possibility to use animations for the individual simulation models as well (Figure 5.21). In this manner we can have a global animation that animates the process of the whole model on a high level and we can have local animations provided by the simulation packages that animate the detailed process of the submodel.

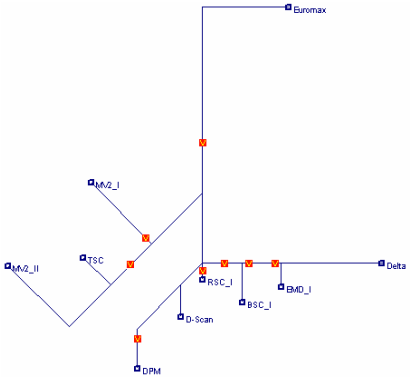


Figure 5.21 2D Animation Offered by eM-Plant

5.5.1.5 The Scenario Creator

As we have specified before, a Scenario Object specifies a distributed simulation run, by having its contents interpreted by the Run Control technical component. An example for a Scenario Object was presented in Table 5.1 in which we could observe three parts:

- Variable Declaration Section
- Initialization Script Section
- Scenario Script Section

Creating a Scenario Object using a text editor increases the chance that one makes mistakes. Therefore, the technical group designed and developed a support application, called the FAMAS Scenario Object Creator, which helps the end users to create consistent Scenario Objects (Figure 5.22).

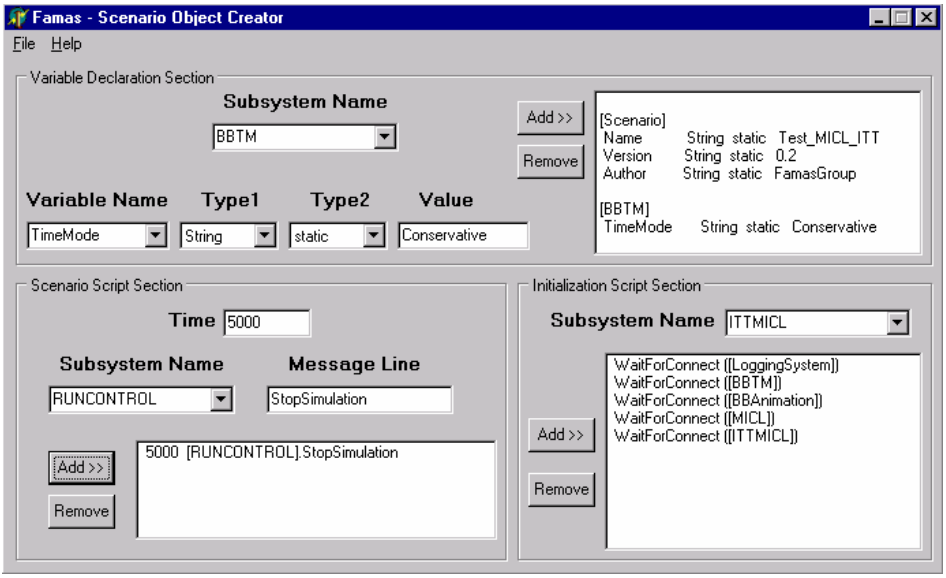


Figure 5.22 GUI of the Scenario Object Creator

The Scenario Object Creator, implemented in Borland Delphi, provides a user friendly interface. The interface is in accordance with the structure of the Scenario Object: end users can open existing scenarios, modify and save them. Further, one can easily create new scenarios with the help of this support application.

5.5.2 The Implementation of the Middleware

As the *U3* requirement (Section 5.3.1) specified, the integration of functional components through the simulation backbone must be accomplished with minimal user intervention. Functional components, as defined in section 5.4.1 can be specific algorithms, simulations, real equipment or controllers. In order to integrate them the components should support the communication protocol defined for the simulation backbone. Accordingly, support of this

protocol should be implemented within the functional components in such a way that minimal user intervention is required.

If the functional components are implemented in a general programming language (e.g. C++, Java, Delphi) the modeller has full control and flexibility to include new concepts within the model, however, general programming languages require the modeller to have appropriate low level programming knowledge in order to implement these concepts. This is remedied in COTS simulation packages that support already existing building blocks (predefined high level components for modelling support) for effective model development. However they are limited in the sense that they do not provide, or even discourage the programmer to consider explicit functionality to interface to external applications. So there is a trade off. Using general programming languages on the one hand gives more freedom, but requires more effort and expertise to implement the models. Using COTS packages, on the other hand, provides less flexibility, but the implementation requires less effort and expertise from the modeller. As it stands now in industry and also in the FAMAS plan, the second stream is the dominant one. For that reason, we have to realize the possibility to implement integrated models from submodels created in COTS simulation packages. As a basis we take the design issues presented in Section 5.4.4. As Section 5.4.4.2 explains it in more detail, in order to access internal information for integrating COTS simulation models through the simulation backbone, we use the middleware approach.

In this section we describe our middleware implementation for two COTS simulation packages, eM-Plant (Tecnomatix 2002) and Arena (Kelton*, et al.* 2003), which are both used in the FAMAS.MV2 project. These two packages are partly open and we build middleware to access internal information. The middleware designed for these two packages behaves like a wrapper around the simulation package: at one side they interface to the simulation package, and exchange information in terms of the simulation package, at the other side they conform to the communication protocol of the backbone. Our implementation follows the idea presented in design phase, namely, for each package we define external and internal middleware. In this sense the middleware provides tools to implement the interoperability function G_D using the publish/subscribe mechanism sketched in Section 5.4.4.2. The internal middleware enables the modeller to extract attribute values from his model (thus in a sense implementing the interface function G_S) and the external middleware transforms these values into the backbone messages described in Section 5.4.4.2. Apart from that the middleware implements time synchronization by reacting on Backbone Time Manager messages in a suitable way.

In order to implement the external middleware we built dynamic link libraries (DLLs) for both COTS simulation packages. The reason behind our choice is that both packages support external communication through DLL files. DLLs are collections of small programs which can be called when needed by another program. The DLLs allow the other program to communicate with the outside world, e.g. a specific device such as a printer or scanner (Rector and Newcomer 1997).

Unfortunately, we were not able to build a common DLL for all existing COTS simulation packages because the COTS packages implement their interface functions in a completely different way. Parts of the DLLs, however, have been implemented in a similar way, because the packages should use the same interoperation functions. In this sense the external middleware is COTS package dependent.

Our internal middleware offers a modelling interface in terms of the modelling concepts (the building blocks) that the COTS package supports. Consequently when users intend to carry out distributed simulation they need to deal only with internal middleware that has the same structure as the other internal components within this COTS environment. In this way, the protocols and internals of the backbone and the external middleware are hidden from the end users, and therefore the FAMAS Simulation Backbone Architecture provides a transparent solution for integrating COTS simulation packages. Consequently, besides the technical components and the communication backbone structure, both internal and external middleware are considered important parts of the FAMAS Simulation Backbone Architecture.

5.5.2.1 Middleware for eM-Plant

According to the concepts of eM-Plant, the internal middleware of this package is implemented as a building block (Figure 5.23). In order to participate in a distributed simulation, the modeller should insert this building block into his model.

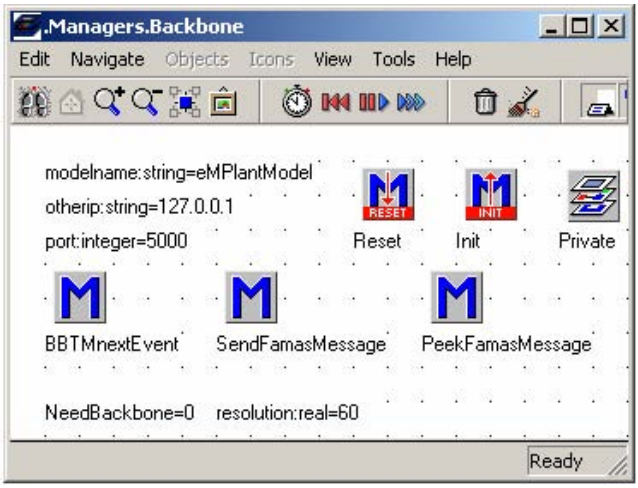


Figure 5.23 Internal eM-Plant Middleware

The building block implementing the middleware makes several methods and parameters available. The `modelname` parameter is used to identify the model. It is used in the scenario and the Run Control component. The `IP address`, named `otherip` and port number named `port` are needed to connect to the Run Control component. The `NeedBackbone` parameter specifies whether the model should be executed in standalone mode or it should participate in a federation. These parameters can easily be changed by the user.

In earlier versions of eM-Plant, that is the versions before eM-Plant 6, it was not possible to access the first event on the event list. So, for those cases, instead of conservative time synchronization, we used a fixed time step approach to synchronize simulation time through the BBTM. This is implemented in the `BBTMnextEvent` method in the following way. The internal middleware generates a fixed number of 'clock tick' events per seconds.

The frequency can be set by the modeller through the variable `resolution` (Figure 5.23), a number indicating how many of such events are to be scheduled pro second.

When a clock tick event is triggered by the simulation engine, execution of the event results in a function being called within the external middleware, the DLL. This function sends a message to the Backbone Time Manager indicating the scheduled time of the next clock tick. After this the function waits, freezing the whole simulation, until the BBTM sends a `NotifyNextEvent` message indicating that the next clock tick can be executed. Only then the DLL function returns, and the local simulation can proceed executing all its events up to and including the next clock tick.

The method `SendFamasMessage` is capable of sending messages to the other components through the external middleware. In order to be able to do that the simulation model needs to find out the address of the other component. As we have discussed earlier the Run Control component provides services for this purposes. The internal middleware offers an Init method which uses the capabilities of `SendFamasMessage` for discovering the address of other components. For instance, the following `TellAddressSubsystem` message is sent to the Run Control component to ask the address of a component named MICL.

```
root.Backbone.sendFamasMessage("RUNCONTROL","TELLADDRESSSUBSYSTEM","MICL");
```

This function call on the COTS simulation package level triggers function within the external middleware that takes care of sending the following message over the backbone:

```
\RUNCONTROL\TELLADDRESSSUBSYSTEM\TellAddressSubsystem\MICL\
```

After receiving an answer to such a message the MICL component is available to be used by the `SendFamasMessage` method.

The `SendFamasMessage` method is called when the model needs to send data to another model. Here is an example call of `SendFamasMessage` sending an `ITT_Arrival` message with certain parameters to the MICL component:

```
root.Backbone.sendFamasMessage("MICL","ITT_ARRIVAL",num_to_str(ContainerNr));
```

When a message is received from the backbone the external middleware stores its contents. The modeller can use the `PeekFAMASMessage` method to inspect whether a message has arrived and to obtain the contents.

The external middleware for eM-Plant is implemented as a DLL written in C++. The functions within the DLL implement the communication message protocol for the FAMAS Simulation Backbone Architecture. As we have seen before, these functions can be directly triggered through the internal middleware.

The eM-Plant function `sendFamasMessage` discussed above triggers the following function within the internal middleware:

```
extern "C" __declspec(dllexport) void SendFamasMessage(UF_Value *ret, UF_Value *arg)
{...
theApp.pFamasMessageHandler->SendFamasMessage(parameters);
theApp.pFamasMessageHandler->checkMessages();
...}
```

The implementation details of the external middleware, including the low level programming details, are hidden from the simulation practitioner. When simulation

practitioners intend to use the backbone they use the external and the internal middleware to integrate simulation models. Due to the fact that they directly deal with the internal middleware which is designed in COTS simulation package concepts, and once the external middleware exists, the integration activity does not require additional low level programming knowledge.

5.5.2.2 Middleware for Arena

The internal middleware for Arena, like the one for eM-Plant, is also implemented as a building block. The building block consists of modules for creating objects, handling events, disposing messages and variables defined by Arena (Figure 5.24).

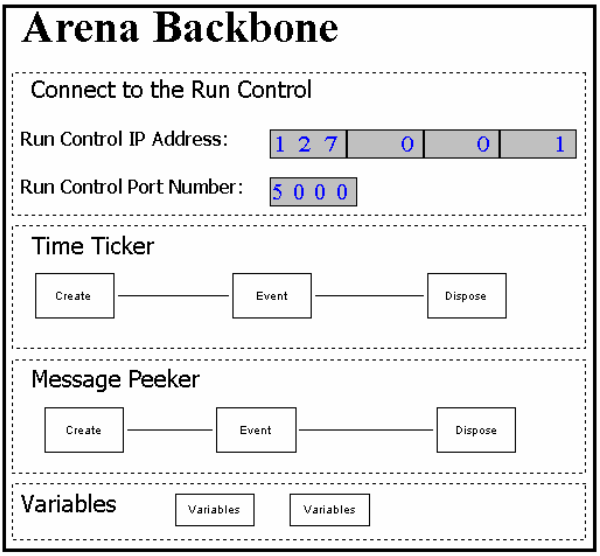


Figure 5.24 Internal Arena Middleware

The external middleware is implemented as a DLL. The DLLs associated with Arena models react when an entity, created in a `CREATE` module, reaches an `EVENT` module. In such a case the `cevent` function in the `DLL` is called with two integer parameters, the identifier of the entity and the identifier of the `EVENT` module. Both integers can be set by the user. If the modeller wants to send a message he has to create an `EVENT` module for each type of message to be sent, and he should create, using the `CREATE` module, an entity with an identifier value that encodes the value of the message parameters. The corresponding code in the cevent function will generate a backbone message based on the identifier of the EVENT module and the entity identifier, and send it over the backbone. The `cevent` function is in fact a kind of interface between the model and the outside world. In contrast to eMPlant, in Arena this interface can communicate only in form of sending/reading integers, which is a constraint entailed by the nature of the Arena package. Therefore, in order to send or receive variables having other types, like string, we need a mapping table for the shared data, as discussed in the design phase (Section 5.4.4).

In Figure 5.24 one sees two `EVENT` modules. The upper one is used by the Time Ticker which implements time synchronization. Due to the fact that the event list cannot be

directly accessed, time synchronization is accomplished in the same way as for the older versions of eM-Plant, namely at fixed regular intervals.

The internal middleware maintains a variable resolution used in a similar way as in eM-Plant: the variable indicates how many times pro second a timer tick entity is created in the `CREATE` module of Figure 5.24, therefore how many times the `Time Ticker EVENT` module is entered, and thus how many times pro second the `cevent` function is called with the identifier of this `EVENT` module as a parameter. On such a call `cevent` behaves in a similar way as we described in Section 5.5.2.1 for the eM-Plant time synchronization.

The internal middleware of Arena uses a special section for peeking messages. Peeking messages is necessary given that the single threaded Arena simulation engine crashes when data is 'pushed' into Arena using a separate thread in the DLL.

The internal middleware employs a special variable `peekres` indicating the frequency according to which arrivals of new messages from other components are to be checked. For this purpose the Message `Peeker` part has been constructed. It operates in a way similar to the `Time Ticker`. The corresponding code in `cevent` checks whether a new message has arrived, analyses this message and takes appropriate action, e.g. by creating a new entity in the model, or by exposing an integer value to the model. However, unlike the Time Ticker, in this case the `cevent` function does return and therefore the `Message Peeker` does not suspend the simulation.

An example of part of the implementation of the interface within the external middleware is depicted here:

```
extern "C" void cdecl cevent (SMINT l, SMINT n)
{ switch (n) {
 case 1:    {            //eventAdvance
             SMREAL currentTime = gettnw();
             theApp.ArenaAdvance(currentTime);
             break; }
 case 2:    {
            theApp.pFamasMessageHandler->sendFamasMessage(…);}
 case 3:    {            //eventPeek:
            MessageType* pMessage = theApp.PeekFamasMessage();
            if (pMessage!=NULL) {
               if ((pMessage->sender == "MICL") && (pMessage->type== "ITT_ARRIVAL")){
                        … //process the message (create truck)}
 }}
```

5.5.2.3 Comparison

Both Arena and eM-Plant implement communication through the backbone using functions such as peek a new message, send a new message, check for a new message, send the next event, notify the next event, etc., which are implemented in the external middleware.

Connecting eM-Plant models to the backbone is more flexible than connecting Arena models because from the eM-Plant environment we can trigger any DLL function directly at any time, there is no need for something like the `EVENT` module. While for eM-Plant the external middleware is quite general and can be applied without adapting, concerning Arena some of the code in the DLL is quite specific for the model which entails adaptation for each new model. Furthermore, in contrast to Arena eM-Plant can transfer not only integers but also other types (e.g. string).

We have presented the implementation of middleware for two major COTS simulation packages: eM-Plant and Arena respectively. These two packages are partly open and were exclusively developed for monolithic simulation. Because not every internal detail is accessible they provide *just* partial solutions for distributed simulation. We tried, however, to exploit the existing opportunities and we created middleware to help the users to interface their simulation models with others ones.

5.6 Summary

In this chapter we presented the simulation backbone, a distributed simulation architecture, the primary aim of which was to support integration of port related simulation models. Additionally, the backbone was designed and developed to be a general solution that can be applied for a large variety of industrial oriented distributed simulation projects.

In the next chapter we evaluate the backbone through two port related case studies in which it was applied. The evaluation aims to determine the appropriateness of the backbone for the required goal and is conducted by analyzing in which measure it satisfies the requirements presented in this chapter. In order to investigate the appropriateness of the backbone as a general distributed simulation architecture in industry we additionally evaluate the backbone with respect to the requirements presented in Section 4.5.1.

6 EVALUATION USING CASE STUDIES

6.1 Introduction

In Chapter 5 we have presented the FAMAS Simulation Backbone project, the objective of which was to provide support for integration of port related distributed simulation models, resulting in the FAMAS Simulation Backbone Architecture. The backbone is designed and developed in a way to satisfy the requirements of several groups of people who develop different simulation models of the new port that will host the container terminals. In order to judge the appropriateness of the FAMAS Simulation Backbone Architecture, it needs to be analyzed to what extent the requirements stated in the analysis phase have been satisfied (Section 5.3). To this end we conducted two case studies for which its suitability has been tested.

The first case presented in Section 6.2 concerns the integration of container terminal models. The second case presented in Section 6.3 is more complex, and concerns the integration of several types of models, among which container terminal models, planning and scheduling models, road traffic models, truck generator models that were built in order to simulate the truck handling process at the container terminal. Both case studies are used to evaluate the backbone requirements presented in Section 5.3.

Besides conducting the FAMAS Simulation Backbone project and the evaluation cases, we have conducted a theoretical research that we presented in Chapter 3 and 4 the primary aim of which was to identify the requirements for an appropriate distributed simulation architecture for industry. Since the FAMAS Simulation Backbone Architecture seems to be an appropriate tool here, in the next chapter we investigate to which extent it fulfils the requirements presented in Section 4.5.1. In the investigation we refer to the two cases again, as they can illustrate the characteristics of the backbone.

6.2 Case Study 1: MICL-ITT Project

6.2.1 Project Initiation and Planning

The case using which we evaluated our backbone was in fact already specified before the design of the FAMAS Simulation Backbone Project started. The case is based on an already existing complex monolithic model from an earlier project developed to simulate a *terminal complex,* referred to as the Maasvlakte container terminal. The monolithic model was developed in the Delphi programming language supported by the TOMAS simulation libraries[42], TOMAS being a collection of simulation libraries developed in the Delphi programming environment (Veeke 2003).

[42] http://www.tomasweb.com/

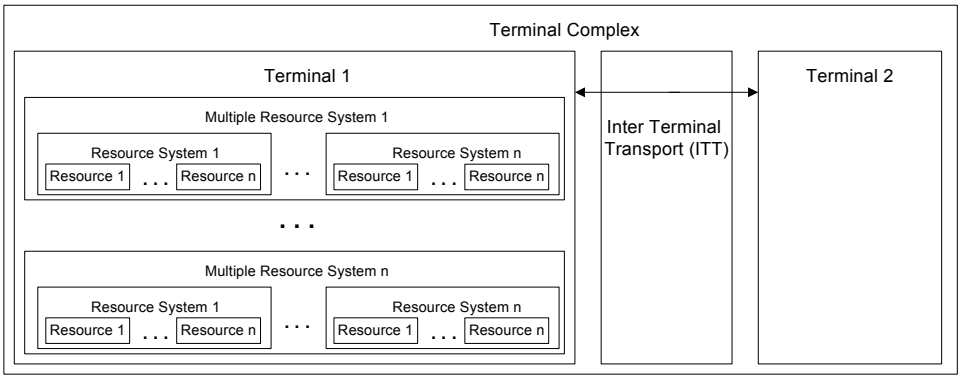


Figure 6.1 Hierarchical Concepts of the Port Processes

Figure 6.1 sketches the hierarchy of the system components for the port project. The components lowest in this hierarchy are the *resources*. Resources refer to personnel, equipment or space. Several instances of the same resource form a so-called *resource system*. If more than one resource is connected, a *resource control* is needed that regulates the collaboration of similar resources. Different resource systems can be combined into *multiple resource systems*, which also contain a coordinating and/or hierarchical control function. A special type of a multiple resource system is the *terminal*, where the system is a geographically bound, autonomous organizational unit with connections to external transport functions. A *terminal complex* is a set of combined terminals. In contrast to single terminals, terminal complexes require the presence of *inter terminal transport (ITT)* functions and usually an overall terminal complex management function. The ITT system connects at least two terminals and is located on the same hierarchical level as the terminals.

6.2.2 Design and Development

The Maasvlakte Integral Container Logistics (MICL) model is a coarse model that simulates the overall functionality of the entire Maasvlakte container port on a high abstraction level including the different terminals and the inter terminal transportation. Our purpose was to break down this complex monolithic model, and try to simulate it using well designed and developed integrated submodels. For integrating the submodels the FAMAS Simulation Backbone Architecture should be applied. Although for the future the backbone is intended to integrate models on different abstraction levels, our first experiment aimed to accomplish integration on the highest abstraction level. For this reason, we disassembled the original model in two parts. The ITT (Inter Terminal Transport) functionality has been taken out of the full MICL model and has been redesigned. In the remainder of the MICL model incoming and outgoing messages have been defined to indicate the need for an ITT vehicle, and the arrival of an ITT vehicle at the destination terminal. Now, the newly designed ITT model had to be coupled to the MICL model through the backbone. Being designed separately, the internal structure and mechanism of an ITT model can be as simple or as complex as the modeller desires. Our intention was to conduct experiments with the same MICL and with different types of ITT models.

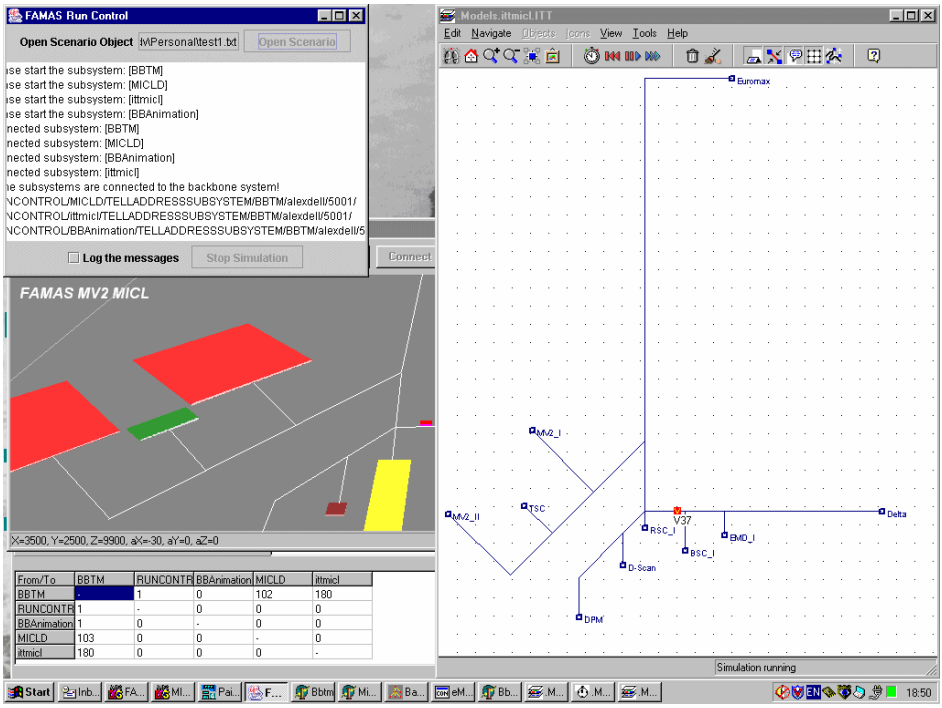


Figure 6.2 Combination of Views, including the ITT Model in eM-Plant, the Run Control Component, the BBTM and the Visualization Component

The MICL was implemented in TOMAS, and several versions of the ITT model have been developed: one in TOMAS – containing the original functionality, as included in the original MICL model, and other ITT models with flexible lay outing in the eM-Plant and Arena COTS simulation environments. As could be expected, the connection of models from similar packages via the backbone, like the TOMAS – TOMAS connection, did not require much effort. We should also mention that the TOMAS models interfaced very easily with all technical components. Since the TOMAS simulation libraries need to be applied within the low level Delphi programming environment, as we discussed before, the end users have full control to implement the required communication protocol within the model. Consequently, the models designed in TOMAS did not require middleware.

The ITT model designed and developed in eM-Plant (Figure 6.2), has been connected to the backbone using the eM-Plant middleware. For eM-Plant the middleware is so powerful, that no special programming in the middleware was needed to implement the messages that are to be exchanged with the MICL model. Our tests showed that the eM-Plant – TOMAS interface worked properly via the backbone, except for one point. Because the future event list could not be read in eM-Plant versions 4 and 5, a truly conservative time mode was not possible. Instead, as we already mentioned in the implementation phase, the timing is based on small time steps that are exchanged with the BBTM.

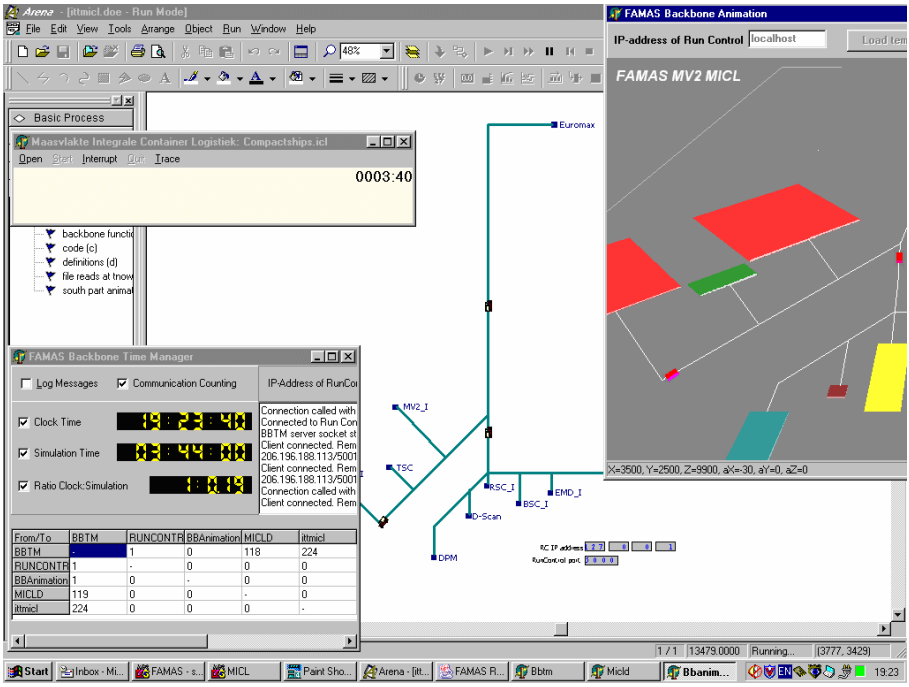


Figure 6.3 Combination of Views, including the ITT Model in Arena, the MICL in TOMAS, the BBTM and the Visualization Component

A simpler implementation of the ITT model has also been developed in Arena (Figure 6.3). Here, the layout of the terminal structure was fixed and cannot be parameterized through initialization files. Furthermore, in order to implement the messages to be exchanged with the MICL model needed we had to insert additional code in the Arena middleware. The Arena middleware also took care of the communication with the technical components. The code for the eM-Plant and the Arena middleware was for 90% the same, only the specific interface with the simulation language differed. The tests conducted showed positive results on the interoperation. In order to fairly compare the two implementations, the timing here is based on small time steps as well, although the Arena event list can be read by the C++ middleware. This is a point for later extension.

For each of the combinations (MICL (TOMAS) – ITT (TOMAS), MICL (TOMAS) – ITT (eM-Plant), and MICL (TOMAS) – ITT (Arena)), simulations were run at an appropriate speed, however the execution speed was noticeably slower (even running on several computers) than running the monolithic version of the model on one platform. Nevertheless, the tests proved to be positive for all aspects, such as reusability of existing models, support for interoperating heterogeneous models, support for collaborative design and development, support for transparent distributed solution, etc. These aspects will be elaborated in the next section when we analyze the fulfilment of the requirements stated in the analysis phase.

A final demonstration test has been done for the steering committee of the FAMAS project and for different companies. The test was presented in the SimLab at Delft University,

where all technical and functional components ran on separate computers. After the presentation of the tests the audience was satisfied and according to them the backbone architecture worked as was expected. Especially for TOMAS and eM-Plant models, the coupling is easy and straightforward. For Arena, a little more effort needs to be done because the C++ Arena middleware needs to be tailored for each project. Probably this can be brought 'closer' to Arena by moving the functionality into the VBA part of Arena.

6.2.3 Evaluation

In our discussion on the analysis phase, in Section 5.3, we presented the requirements for the simulation backbone from three different perspectives viz. that of the user (*U*-requirements), the developer (*D*-requirements) and the administrator (*A*-requirements). Next we present the evaluation of the suitability of the FAMAS Simulation Backbone Architecture for the above case according to these requirements. The discussion concerning the fulfilment of requirements follows the same order as the requirements presented in Section 5.3.

U1. *Quality*: This requirement refers to the fact that the backbone, being a qualitatively satisfactory solution, should support each partner with extra functionality in simulation projects for an acceptable price. The costs that the end users suffer are the costs that are needed to adapt the model to be backbone compliant. Costs are less if straightforward adaptability is supported by the modelling framework and well designed and developed middleware are available. Costs can be, however, more significant, if the framework is closed or partly open and as a consequence designing and developing a middleware involves additional effort. The time dimension of the cost needed to design and develop a middleware depends on how the simulation framework supports interoperability with external applications. Nevertheless, as the previous case illustrated, having designed and developed the middleware for eM-Plant, much less time and effort was required to design the middleware for Arena. In our case the external middleware for the two packages was for about 90% the same. The involved participants, who developed or reused the models, respectively the FAMAS steering committee which finally evaluated the backbone were satisfied with the quality of the backbone and they have found the cost acceptable when integrating simulation models using the backbone.

U2. *Reusability*. Although the case presented above is simple and it serves demonstration purposes only, it still provides an example for reusability of already existing simulation models. In our case, in all three experiments coupling ITT models developed in different frameworks, the MICL simulation model implemented in TOMAS was in fact the same. In this sense the reuse of this MICL model did not entailed a significant cost when we reused it for integrating other ITT models. The reusability of this model when integrating it with another model was as a consequence, implicitly much less than reimplementing it from the scratch. The backbone satisfied thus, for this case the reusability requirement, as stated in the previous chapter.

U3. *Structure transparency*. In order to hide the details behind the distributed approach and integration we applied the concept of external and internal middleware. The internal middleware is exposed to the modeller in terms of high level concepts from

the COTS simulation framework and as a consequence understanding and applying it does not demand additional programming knowledge from the user. The low level details of distributiveness are implemented in the external middleware, which is a bridge between the internal middleware and the simulation backbone. In a full implementation of this idea, the external middleware and the simulation backbone remain hidden from the end user who just needs to use the internal middleware in order to integrate the models participating in a project. In such a case, this solution is transparent towards the end users. Our test case showed that we were able to realize this desired state of affairs in eM-Plant. In Arena we were not able to fully reach this goal in the sense that for each model some extensions had to be added to the external middleware. These extensions were relatively minor and located in a well defined place inside the middleware code.

U4. Support for heterogeneous COTS simulation packages. Although the FAMAS Simulation Backbone Architecture is designed and developed to be as generic as possible, its primary aim is to support the industrial domain, especially the port simulation community. Since industry for most of its simulation projects uses COTS simulation packages and the backbone provides appropriate support to integrate models created in these packages, the FAMAS Simulation Backbone Architecture provides a proper solution for industry. In the case presented above ITT models developed in three different simulation packages (Arena, eM-Plant and TOMAS) have been integrated through the backbone. This case illustrates the possibility to integrate models developed in heterogeneous COTS packages.

U5. Centralized view for conducting distributed simulation. In the backbone the centralized view is solved through several technical components. The table below (Table 6.1) summarizes the technical components and support systems that aim to offer a centralized view for controlling and executing a distributed simulation run. These components have been developed in heterogeneous platforms (e.g. Java, Delphi, and Visual Basic).

Table 6.1 Components Implemented in Different Platforms

Component	Function
Run Control	- start distributed simulation - monitor distributed simulation - terminate distributed simulation
Time Manager	- global time synchronization
Logging	- centralized data collection
Visualization	- centralized visualization
Logging Viewer	- centralized data analysis
Scenario Creator	- distributed scenario creator

U6. Acceptable speed of simulation runs. Concerning the speed of execution of our distributed simulations, our expectation was not too high. One of the essential deficiencies of distributed simulation is the fact that it performs slower than monolithic simulation. This inefficiency arises from the fact that integrated

distributed simulation models must communicate frequently to synchronize their actions. If one applies an appropriate time synchronization mechanism (like the optimistic one), a proper communication structure and of course a large bandwidth network for communication, a backbone might generate efficient results that approximate the monolithic solution. However, for time synchronization we applied only the conservative variant, which in fact is not the best choice when the speed of simulation run is concerned. As we stated in the design of the time management component the implementation of optimistic time synchronization mechanism is very difficult when integrating models designed and developed in COTS simulation packages. Recently an article appeared on this issue where the authors describe theoretically the possibility to apply optimistic time synchronization between integrated COTS simulation models (Wang, *et al.* 2004). In order to reduce computation time and to avoid overloading, we focused not on efficient time synchronization but on designing an appropriate communication structure that, instead of providing a central component for data distribution which would act as a possibly inefficient switching board, would establish peer to peer communication between the components. In this way each component can directly communicate with another one. We performed a test in which all components executed on one single computer. The result of the test was unsatisfactory because the increasing number of components overloaded the computer. Therefore, the final test was conducted using several computers connected by an Intranet network (the SimLab in Delft University). In order to avoid overloading any of the computers, we installed each technical and functional component on a separate computer. The bandwidth of the Intranet network in the SimLab being large enough and the computers being powerful enough we obtained acceptable performance results (in the eyes of the audience) concerning the speed of the distributed simulation run.

U7. *Scalability.* Due to the fact that the case presented above was simple, containing just a few components, we have not done explicit measurements regarding its scalability, and as a consequence, we cannot draw definite conclusions regarding it. However, we have some remarks concerning this issue. The structure of the backbone is designed and developed in such a way that direct peer to peer communication between the components is established. This approach minimizes communication over the backbone compared for example to approaches that send all the information through a technical component that is responsible for data distribution, behaving like a switching board, that can be easily overloaded if the number of integrated components increases. Due to peer to peer communication, our expectation is that the backbone eliminates the possibility to be overloaded. Although the structure of the backbone suggests that the scalability of the architecture is not a problem, the applied conservative time synchronization suggests the contradictory. The fact that during conservative time synchronization only one model is considered as “current” at any moment, the integration of a large number of simulation models will probably entail decrease in performance.

D1. *Service based architecture implemented in modular way.* The FAMAS Simulation Backbone Architecture has a component based structure which consists of technical and functional components. As defined above, the functional components are the simulation models, control algorithms, real equipment, and so on, while the technical components provide distributed simulation services to these functional components. The component based structure of the backbone enables isolation of the functional and technical components. In this way transparency towards the end users can easily be provided, letting them focus only on the implementation of the functional components.

D2. *Efficient communication.* TCP/IP is the basic standard communication protocol for the Internet and enables experimentation with geographically distributed simulation models. Therefore we chose this protocol as the communication protocol for the FAMAS Simulation Backbone Architecture. Furthermore this protocol enables to establish a reliable peer to peer communication. Being based on a peer to peer communication structure, communication over the backbone is minimized providing an efficient solution for communication. In this way the backbone supports direct communication between components, helping to provide a lightweight structure. Applying this protocol one can easily set up a distributed simulation through the Internet.

D3. *Generally applicable solution for interfacing with COTS packages.* In our discussion on the design phase we introduced the concept of internal and external middleware. Based on these concepts we expect to be able to implement middleware for any COTS package. In the case presented in the previous section two COTS simulation packages were involved, eM-Plant which follows an object oriented modelling paradigm and Arena which follows a process flow oriented modelling paradigm. For both packages we have implemented internal and external middleware based on the conceptual design discussed in Section 5.4.4. However, comparing the middleware developed for the two packages, as we have seen the external middleware is package specific and at least for the eM-Plant is hidden from the simulation practitioners, while the internal middleware uses the concepts, building blocks, offered by the COTS package concerned. Accordingly the simulation practitioner deals with notions that he is used to and does not need additional low level programming knowledge for the eM-Plant case.

D4. *Support for debug facilities.* The FAMAS Simulation Backbone Architecture does not provide a specific tool that takes care of debugging, but it provides support for debugging a model. This support is based on the fact that the backbone uses a message protocol on top of TCP/IP, which is readable for the end user. Since the implementation of most of the components offers a monitor window which interactively shows the messages exchanged, and since messages can be saved using the Logging component, the simulation practitioner can debug a distributed simulation execution by studying the order and the content of the messages exchanged.

A1. *Maintainability.* Maintenance of any component, like for instance the time management technical component, does not require reconsideration of the whole architecture of the backbone. In the case presented above we were able to simply replace the time management component with a new one. Although in the new time management component we changed the internal implementation and we chose another platform to implement it, the interface to the other components remained consistent. In the worst case, we need to replace the whole component, like we did with the time management component. The maintenance cost in this case was essentially the cost of redesign and reimplementation of the component concerned, and did not require the adaptation of the backbone.

A2. *Extendibility.* The FAMAS Simulation Backbone Architecture has a component based structure which enables one to easily replace or extend technical components. We proved this concept when we replaced the time management technical component designed in Delphi with an improved one designed in Java or when we extended the backbone with the Logging and Visualization technical components.

By carrying out a small case study using the FAMAS Simulation Backbone Architecture, we observed that the backbone satisfies almost all requirements. However, for some of the requirements more tests are needed to investigate whether they are fulfilled.

6.2.4 Summary

This section presented the first evaluation of the FAMAS Simulation Backbone Architecture. However, as we have stated before, the backbone is designed to be applied in different FAMAS projects. Accordingly, in the next chapter we introduce another FAMAS case in which we applied the backbone. Compared to the previous case the next one will be more complex. Therefore, we expect that backbone requirements can be analyzed to a larger extent. The aim of the next section is to present this case and to analyze again the extent to which the backbone fulfils the requirements.

6.3 Case Study 2: Pre-design Road Container Handling

For the first case study the backbone fulfilled most of the requirements as discussed in the previous section. However, in order to draw more general conclusions regarding the appropriateness of the backbone for integrating the models of the projects within the FAMAS.MV2 research plan, it is necessary to perform further evaluations. For this reason, in this section we present a more intricate case in which the backbone is used again to integrate various components. Being more complex, this case can provide a more comprehensive evaluation of the backbone and expectedly strengthen the appropriateness of this solution. The evaluation is performed in the same way we have done for the first case, that is, we analyze again the extent to which the backbone fulfils the requirements of the research plan presented in Section 6.1.

6.3.1 Project Initiation and Planning

One of the difficulties that container terminals are faced with is the handling of the truck arrivals. The current situation sometimes results in large number of trucks waiting in excessively long queues, as they arrive more frequently than they can be served. This situation especially occurs during peak hours. Due to the limited number of serving cranes and the limited capacity of parking places these trucks are confronted with delays which is a costly situation (Connekt 2001). One of the goals of the FAMAS.MV2 project is to remedy this. The case study described in this chapter focuses on this problem.

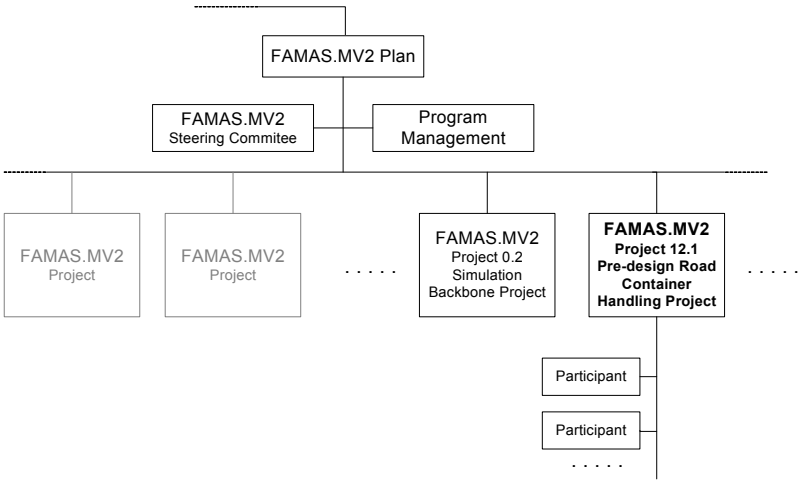


Figure 6.4 Organization of FAMAS.MV2 (Miller and Melis 2001, pg. 6)

The FAMAS.MV2 TA Project 12.1, also called FAMAS.MV2 Pre-design Road Container Handling project (see Figure 6.4) aims "to develop economically realistic and socially acceptable logistic concepts for the handling of road containers on the future 2nd Maasvlakte and to define the corresponding criteria for the design and the technical feasibility thereof, together with the business community concerned, knowledge institutions and the authorities in a chain perspective. Based upon the results of 'International state-of-the-art in container logistics and performance requirements for mega hubs' (Connekt 2001), it will strive to design logistic concepts with which 90% of all trucks will be handled within 30 minutes."(Miller and Melis 2001, pg. 4). This project involved several organizations. The list of participating organizations and their representatives can be found in Appendix F.

Concerning the planning of this project, Miller and Melis describe five important tasks that have to be performed (Miller and Melis 2001):

1. Survey of market and demarcation of area of investigation;
2. Design (simulation) logistics models;
3. Develop (simulation) logistics models;
4. Show case;
5. Present the results in an end report.

Concerning the first task, in (Connekt 2003) we can find comprehensive information regarding data collection, analysis of different terminal concepts, definition of cost models, and so on. We were not involved in this first task, only in the second, third, fourth and partially in the fifth one in which the applicability of the simulation backbone is relevant. Since our main goal with this case is to evaluate the FAMAS Simulation Backbone Architecture, in this chapter we focus only on these issues. Accordingly, in Section 6.3.2 we present briefly the design and development of the constituting (simulation) logistic models, and in Section 6.3.3 we demonstrate the case and evaluate the appropriateness of the backbone against the requirements presented in Section 5.3.

6.3.2 Design and Development

In this section we aim to provide insight in the design and development phase of the FAMAS.MV2 Pre-design Road Container Handling project. To start with, we present an overall view on the simulation study and we discuss why it is beneficial to design and develop the model in a distributed way instead of a monolithic way. Then we present the conceptual design of the distributed model which uses the FAMAS Simulation Backbone Architecture. Finally, all functional components which are part of this distributed model are presented briefly one by one.

6.3.2.1 General Overview

There are several processes that need to be carried out when delivering or picking up a container at a terminal. First of all truck companies that intend to deliver or pick up their containers at the terminal have to contact the terminal operator in order to make an appointment. The truck companies usually specify a request that contains the desired arrival time at the terminal. After obtaining a time slot that describes the arrival time at the container terminal which is adequate for both parties, the truck can drive to the container terminal. Finally, the truck arrives at the terminal where it delivers or picks up the containers.

Based on the previous description in our simulation study we distinguish four processes (Figure 6.5): the Time Slot Request and Truck Generation Processes, Planning and Scheduling, the Driving Process and Container Handling.

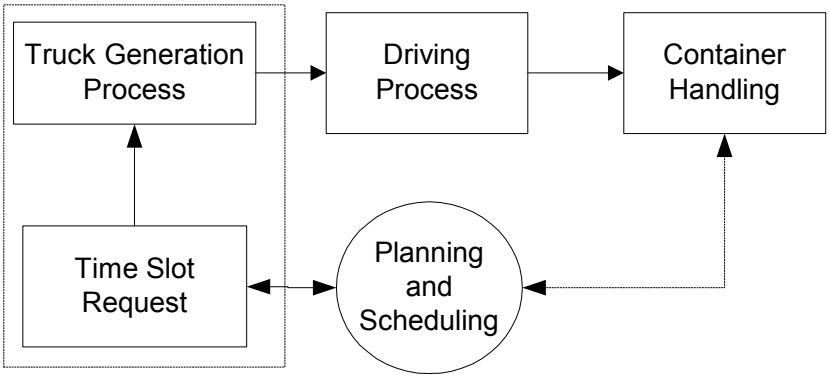


Figure 6.5 The Main Interoperability Processes

In the first process, Time Slot Request and Truck Generation, models are designed and developed that generate requests, negotiate with the intermediary, and creates truck instances to be sent to the container terminal using the road traffic model. The request includes the specification of the desired arrival time, the type of container, etc. Because of the fact that the container terminal has limited capacity (cranes, parking places, etc.) it might happen that too many truck companies request the same time slot. Reservation of a timeslot is solved by a multi agent negotiation system. Agents negotiate on reservations on behalf of the truck companies and the terminal operator. The negotiation process is influenced by several constraints, such as the trucks currently available, the currently available drivers, the time limits within which the trucks are driving (e.g. they do not drive at night), that the negotiator agents need to take into account. These tasks are performed by a Planning and Scheduling system that attempts to provide the best time slot for each truck. One of the participating organizations was responsible to design and develop this system using high level negotiation mechanisms applying recent technologies, such as intelligent agent based technology, web services, XML, etc.

Unfortunately, the driving time cannot be predicted precisely due to traffic delays that might occur. Therefore we introduce a Driving Process that takes into account the delays that might occur. For this reason a model is designed that simulates the driving process of the trucks that go from the destination to the port, with or without container(s). Different types of driving models exist: micro and macro traffic simulations, differing in the level of detail they take into account. Highly detailed models lead to more accurate results but require very detailed input (Flinsenberg 2004).

The last process that needs to be modelled is Container Handling. This operation is carried out by a container terminal simulation model. The model of the container terminal is highly detailed, which is essential in order to give a realistic prediction, as the efficiency of the truck handling is very dependent on the other processes in the terminal, and there are many shared resources. The Container Handling model did not need to be designed and developed anew since an existing model could be reused, that did not incorporate a truck arrival planning and scheduling system (Van Til 2003).

The design and development of the whole complex model can be done *monolithically* (one big model using one package) or in a *distributed way* (well distinguished models designed and developed separately in the same or different packages). Analyzing the possibilities we concluded that it would be extremely difficult to implement the agent based negotiation system in a simulation language – which would also result in a waste of resources, as this system has already been built in the case we are studying. The monolithic approach, that most simulation environments support as the only choice, always poses such difficulties in complex modelling tasks where different model parts from different background disciplines need to be integrated. In the cases where other types of systems have to be included as well, the problem is even more aggravated. Considering the characteristics of distributed simulation projects in industry discussed in Section 4.2, we observed that we extensively meet these characteristics in this second project. Namely:

- The simulation model of the container terminal existed and there is a need to reuse it;
- The whole model entails various concepts that need to be represented in heterogeneous environments, such as the agent based planning and scheduling

system in the Java environment or the container terminal simulation model in eM-Plant;

- The way the internal algorithms of some models are implemented, like the negation mechanism of the agent based planning and scheduling model is not publicly available, not even for the participating organizations.

Therefore instead of a monolithic approach we propose a distributed one in which the possibility to interface with real planning and scheduling software is possible, and the existing developed simulation models and algorithms can be put to use.

By applying the distributed modelling approach the various participating organizations can develop their own model without sharing their business logic. The truck companies, for example have to focus only on the truck generation process, and the terminal operators just on the container handling, while they share only information that is relevant for the negotiation process. The planning and scheduling system can be designed separately as well, not being considered as part of a certain simulation model, but being in contact with the relevant participant models and requiring only internal information that is necessary for the negotiation process. A big advantage of this distributed approach, as illustrated here, is the possibility to hide internal information and the increased efficiency of the individual work as different modellers from different organizations can work in parallel (Taylor*, et al.* 2003b).

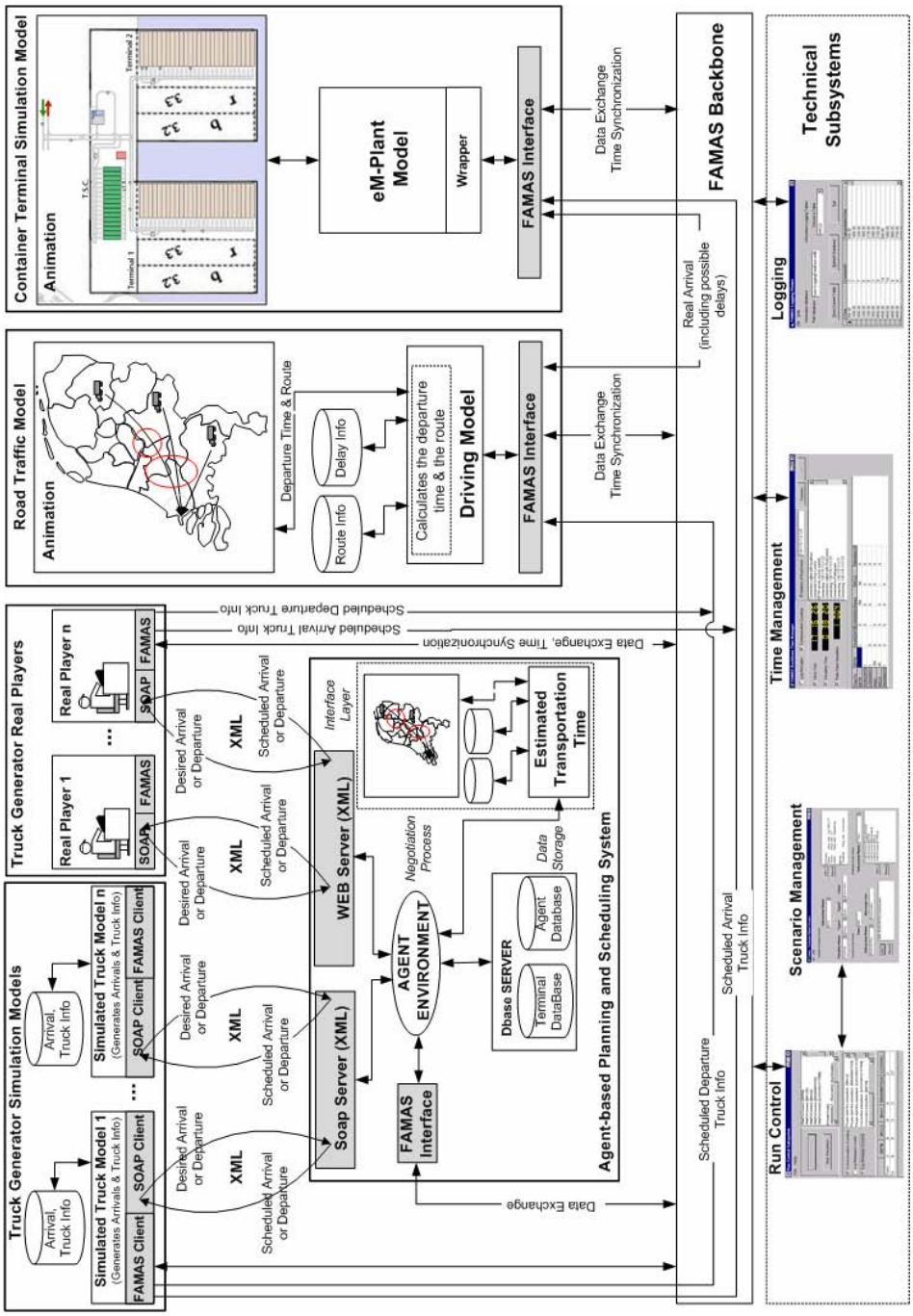


Figure 6.6 The Conceptual Distributed Model

6.3.2.2 The Conceptual Distributed Model

In Chapter 5 we presented the FAMAS Simulation Backbone Architecture, designed with the aim to serve future FAMAS.MV2 projects which require integration support. Consequently, for our purpose we chose the FAMAS Simulation Backbone Architecture to achieve the interoperability between the various models of this project.

When applying the FAMAS Simulation Backbone Architecture, the models can use the FAMAS communication protocol for time synchronization and data exchange, however other communication protocols can be applied as well. It is more convenient for the planning and scheduling tool, for example, to communicate with the truck generator model through SOAP (Simple Object Access Protocol) by using the World Wide Web's Hypertext Transfer Protocol (HTTP) and its Extensible Markup Language (XML) as the mechanisms for information exchange (Schmelzer*, et al.* 2002). Since in this way the truck generator model and the planning and scheduling model can exchange request and confirmation messages at any time and without putting a burden on the FAMAS Simulation Backbone Architecture, we decided to apply the above mentioned protocol. This way of communication in parallel with the backbone reduces the amount of messages to be processed by the technical components. Figure 6.6 depicts the conceptual distributed model of the whole system including the interoperability between different models. This picture includes the truck generator simulation models, the planning and scheduling system, the road traffic system and the container simulation models introduced in the previous section. In the next subsections we elaborate on the design and development of each model, treating them on by one.

6.3.2.3 The Truck Generator Model

In the Truck Generator model truck instances are generated based on requests for picking up and delivering containers. The requests can be either generated automatically using truck simulation models or in real-time by users using an interactive application (see Figure 6.6). Requests are generated in the form of timeslots that refer either to a *desired departure* time (when the truck starts driving to the port) or a *desired arrival* time (when the truck should arrive at the port). It is not guaranteed, however, that a truck can start the driving process at the desired departure time, because of the limited capacity of the container terminal, that given the limited parking places, cranes, etc., can accept only a certain amount of trucks at a given time. As there can be more truck organizations that request the same timeslot for arrival time, the container terminal might not be able to handle all of them at the desired time. In order to avoid congestion and time wasted by waiting at the terminal for handling, a negotiation process is taking place between the truck companies and the port authorities. The negotiation process results in a *scheduled* time slot for the departure and arrival time of the trucks at the container terminal. This is implemented in the scheduling and planning model described in Section 6.3.2.6.

The planning and scheduling model which performs the negotiation uses a model of a road traffic system in order to estimate the transportation time. This is needed in order to find out the approximated arriving time if the departure time is given or vice versa. Having obtained the scheduled arrival and departure time, the truck generator model sends a message over the backbone containing the scheduled departure time to the road traffic

model and a message containing the approximated scheduled arrival is sent to the container terminal model. The road traffic model is described in the next subsection.

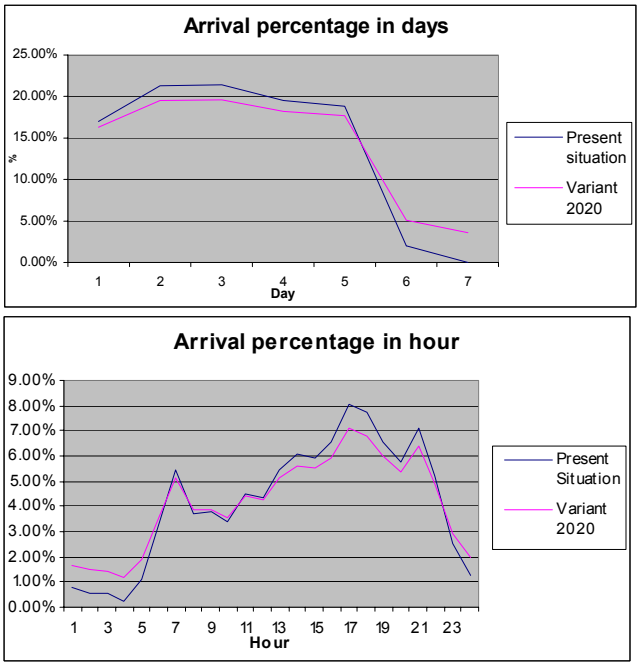


Figure 6.7 Arrival Percentages in Days and Hours

The simulation model of the truck generator stochastically generates requests and creates truck instances based on earlier observed historical data (Figure 6.7). The Truck Generator uses special mechanisms to simulate the reservation of the desired arrival or departure time requested by truck companies. It allows the truck companies to make a reservation for a timeslot for a certain time period in advance. In our case we fixed this to be one week. The trucks that make a reservation for a time slot earlier in time get the desired time slot more easily compared to the companies that register late.

We implemented a general Truck Generator that can be instantiated in various ways by the different truck companies. The companies should use a configuration file, where they can specify company specific data, such as the number of trucks available, driving time, etc. Using several configuration instances we can run various scenarios for the truck companies.

The simulation model is designed and developed in the Java environment by Erasmus and Delft University (Figure 6.8). The interoperability between the Truck Generator and the other models is achieved by the FAMAS Backbone and Simple Object Access Protocol (SOAP). The negotiation process between the Truck Generator and the Planning and Scheduling model is achieved through XML. Therefore the Truck Generator model is developed as a SOAP client for the planning and scheduling model, and as a FAMAS client for the backbone. Being a simulation model it is scheduled through the backbone (Figure 6.8).

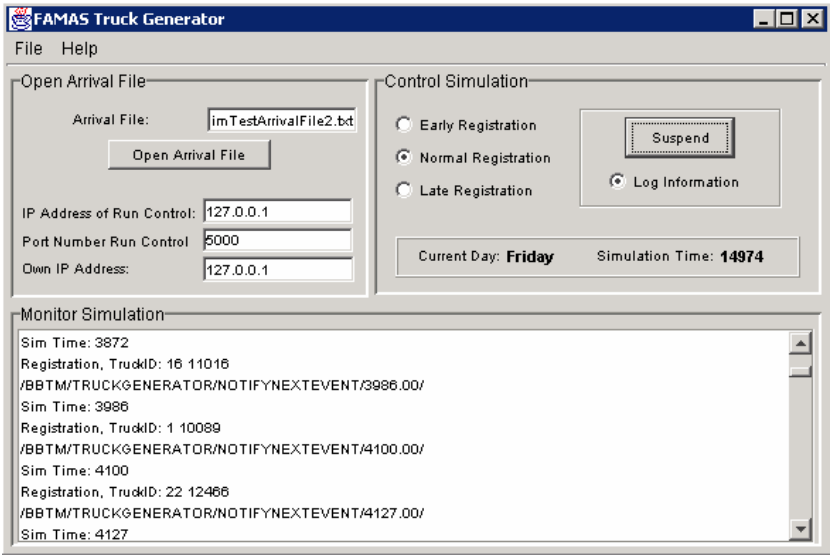


Figure 6.8 Graphical User Interface of FAMAS Truck Generator Simulation Model

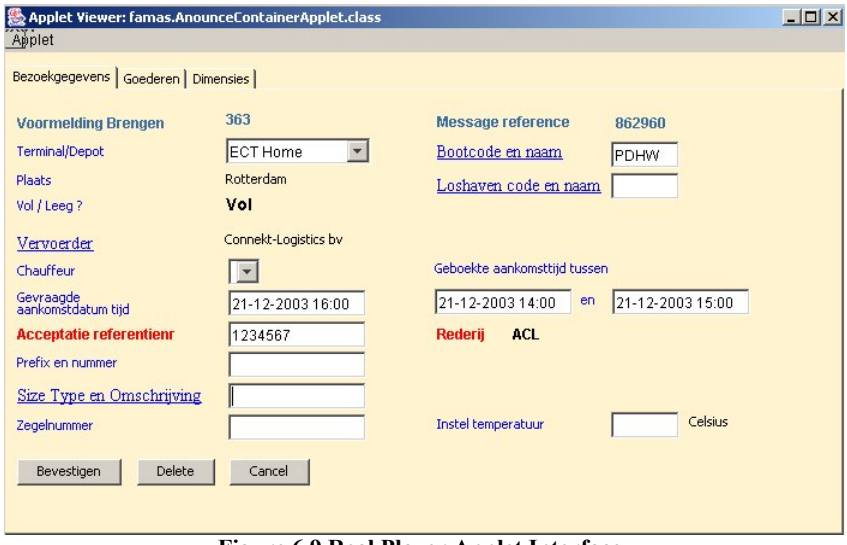


Figure 6.9 Real Player Applet Interface

As we mentioned before, besides being implemented as a simulation model, the Truck Generator can be a real-time model controlled by users in real-time. In order to show the functioning in a simulated "real" environment, a real player interface (Figure 6.9) has been developed by ILLYAN company to submit time slot requests manually (Connekt 2003, pg. 62-65). In this case each truck company can make the reservation using the facilities provided by Internet. This option is included for demonstration purposes only, and has not taken part in the full simulation.

6.3.2.4 The Road Traffic Model

The aim of the road traffic model is to represent the driving phase of the trucks from the companies to the port based on a given starting point and time provided by the truck generator, and external route and delay information provided by input data. Due to the fact that we are working in a distributed environment, we can use existing models of road traffic systems, the only issue to be solved is interfacing with the other systems.

In the full distributed model two road traffic models exist. The first one is used to simulate how trucks drive to the port. The second one is applied by the planning and scheduling model to estimate transportation time in advance when not the desired arrival time but the desired departure time is provided by the truck generator. Both types stochastically define the driving time, which highly depends on particular days, hours and routes.

The input for the Road Traffic model contains route and delay information stored in a database. Depending on the level of detail, external data might provide further information on the distance between two points (e.g. two cross points, two cities, etc.), the name of the road (e.g. A1), the maximum speed limit on that road, earlier measured delays on this distance at different days and hours, etc.

The Road Traffic model might include visualization by means of animation. The animation represents a map (e.g. a map of The Netherlands or Europe) and visualizes the driving process of the trucks. In our final distributed model this has not been included, only textual information has been generated (Figure 6.10).

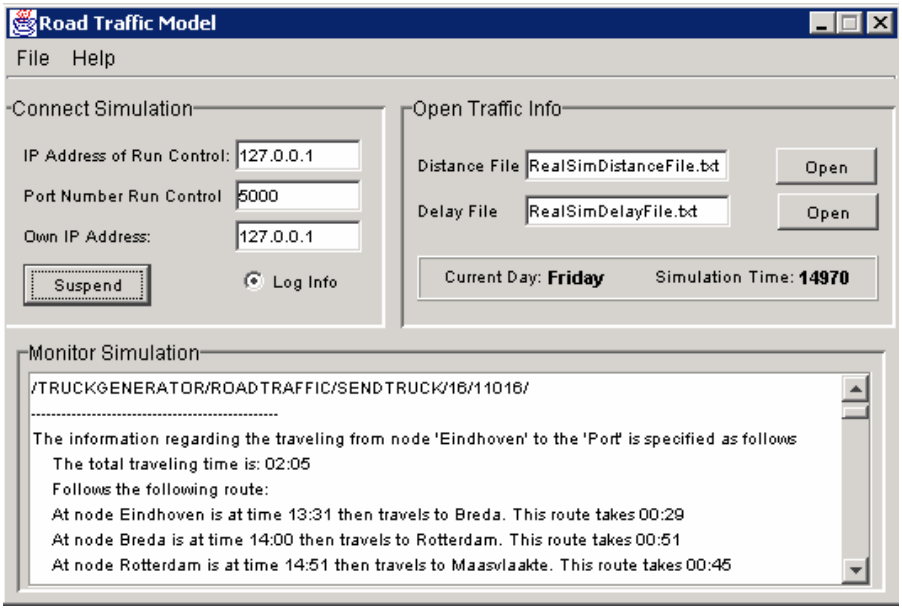


Figure 6.10 Graphical User Interface of the FAMAS Road Traffic Simulation Model

Data exchange and time synchronization with other models is solved through the FAMAS Backbone. Being a simulation model, the model is scheduled through the backbone. The

Truck Generator model sends a message containing a scheduled departure time through the backbone to the road traffic model, indicating the time at which a truck starts driving to the port. As we mentioned earlier, this process is not deterministic, as we can count on unexpected delays or accidents, and therefore the earlier provided scheduled arrival time to the port might differ from the actual arrival. Therefore, although the container terminal is informed about an approximately scheduled time indicating when the trucks should arrive to the port, the driving model is responsible for sending a pre arrival message, informing the terminal about the actual arrival of the truck. In this way the Automated Stacking Cranes (ASC) are able to start preparing the requested container(s) for the truck if needed. This message is also needed to schedule truck arrival in the terminal simulation.

The Road Traffic model was developed in the Java environment as well by Erasmus and Delft University (Figure 6.10). The way it is designed is satisfactory for demonstration purposes, however, due to the fact that not all routes are covered by our model, in order to better suits its purpose, in addition to the basic functionality described above route finder web service applications can be applied. The route finder web services also generate driving directions between two locations. However in contrast to the simulation model they do not take into consideration the possible delays that might happen due to traffic jam. The returned results are displayed as a route map and textual directions. Examples are ArcWebService (http://arcweb.esri.com/), MapPoint Web Service by Microsoft (http://www.microsoft.com/mappoint/net/).

6.3.2.5 The Container Terminal Model

A complex model of the container terminal already existed before the project was initiated. This model did not depend on a planning and scheduling system and had its own simple truck generator. However, the project is aimed to improve the handling process of trucks at new container terminals and a planning and scheduling system needs to be applied (30 minutes 90% reliable). In an earlier FAMAS.MV2 project five container terminal concepts (designs) were identified concerning the truck handling process (Connekt 2003). Two of these have been chosen to be investigated in detail by means of simulation modelling (Van Til 2003).:

1. Compact Terminals (Concept 1);
2. Compact Terminals with a Central Gate and Truck Service Centre (Concept 5).

One aspect that these two concepts have in common is that a truck generator model is required, which generates trucks according to some predefined distribution as indicated in Section 6.3.2.3. Both concepts have been designed and developed on a detailed level, which is essential to provide a realistic prediction.

Concept 1: Compact Terminals

The schematic overview of the truck processes using the Compact Terminals concept is depicted in Figure 6.11.

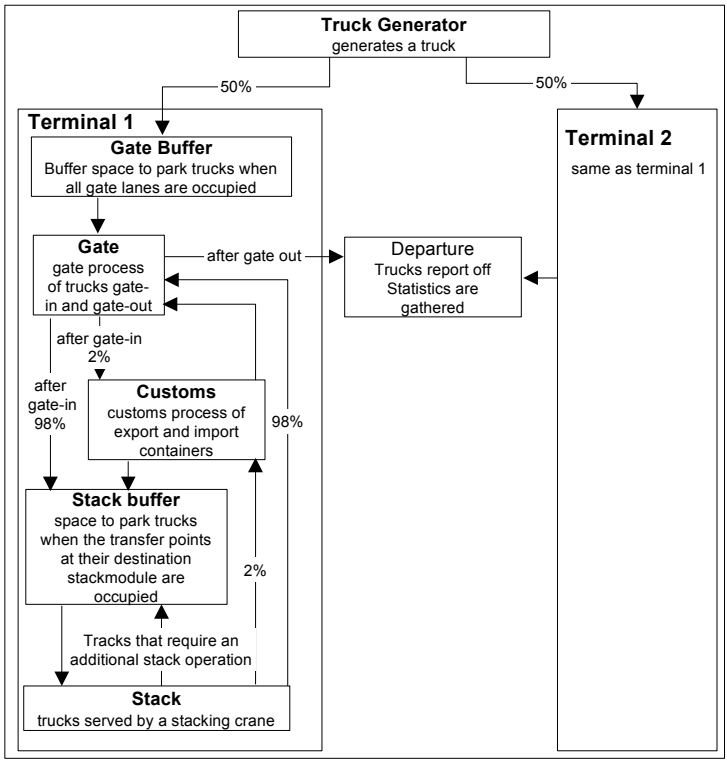


Figure 6.11 Schematic Overview of Concept 1

According to the schema, when a truck arrives at the terminal, the truck is assigned to a gate and is positioned in the buffer in front of the assigned gate. When one of the gate lanes becomes available, the truck continues its trace. After the treatment that is required at the gate lane (e.g. identification, registration, etc.), the truck proceeds towards the customs. At customs, 2% of the trucks are checked randomly. The selected trucks are scanned after which they drive towards the substack (place for container storage) of their destination. If no transfer points are available at the substack, the truck is positioned in the stack buffer, where it waits until one of the transfer points become available.

Trucks that are not checked at customs go directly to the substack of their destination, after passing the gate, where they are loaded or unloaded by the ASC (Automated Stacking Crane). If a truck still has orders after the stack treatment, it will preferably stay in the same substack for the remaining orders. If this is not possible (e.g., because it has to pick up a container from another substack) the truck will go to this other substack, if necessary via the buffer. If all orders are carried out, the truck drives back to the gate to sign out.

The simulation model of this concept is implemented in eM-Plant by TBA Nederland. Figure 6.12 depicts the design of the compact terminal concept.

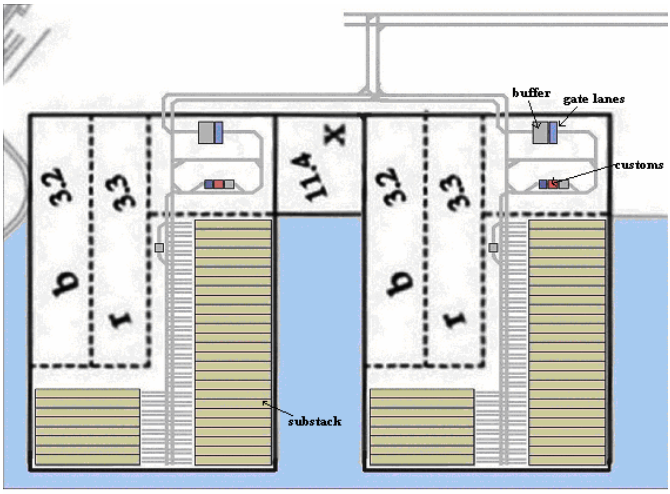


Figure 6.12 Layout Terminals for Concept 1

Concept 5: Compact Terminals with a Central Gate and a Truck Service Centre

The second concept investigated is a centralized variant of the first one. The schematic overview of the truck processes using this concept is depicted in Figure 6.13. In contrast to the first concept, this one has one single central gate instead of more gates and one single central custom instead of more customs. Furthermore, Concept 5 introduces two new notions that play an important role in the management of the containers: the Truck Service Centre (TSC) and Inter Terminal Transportation (ITT).

As depicted in Figure 6.13, the truck generator generates trucks that all pass the same gate to sign in and out. After leaving the gate, some of the trucks have to go to the TSC. This applies to the trucks that have to be checked by customs (1% in the case of deliveries, 2% in the case of pick ups) and trucks with off-standard containers (5% - 10%). The TSC consists of a stack where all off-standard containers are stored, as well as export containers that have to be scanned by customs and import containers that are already checked by customs. Special off-standard substacks are available for the off-standard containers and normal substacks for the regular containers. A truck that is sent to the TSC delivers or picks up the containers. The other trucks drive straight to the terminal stack. In this case, none of the trucks need to pass customs, as transport to customs is executed by the Inter Terminal Transport system (ITT).

Just like in the compact terminal version, buffers are available to park trucks when the transfer points at the destination substack are occupied. If necessary, trucks visit several substacks in order to handle all the orders. When all orders are carried out trucks drive back to the gate to sign out.

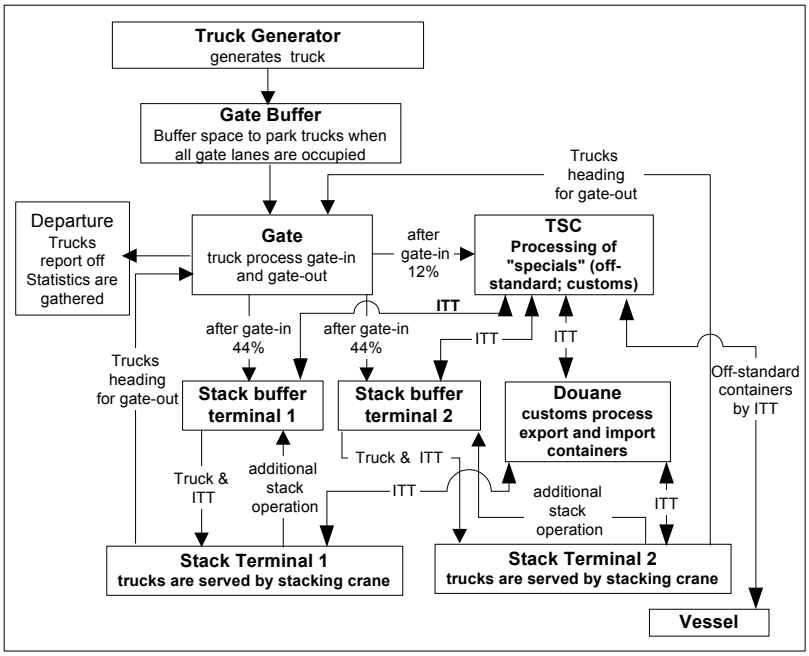


Figure 6.13 Schematic Overview of Concept 5

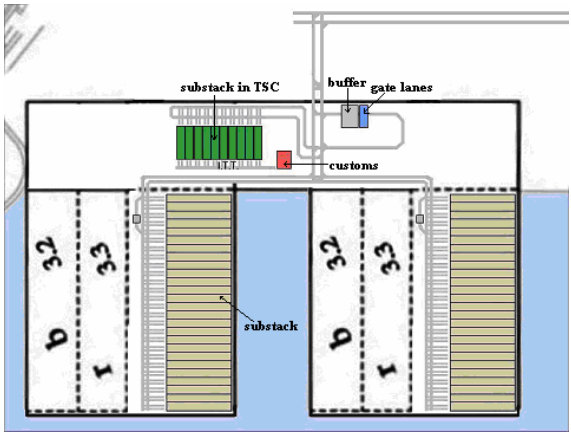


Figure 6.14 Layout Terminals for Concept 5

Like the first concept, this one has also been implemented in eM-Plant by TBA Nederland. Figure 6.14 shows the design layout of this concept. Transport between the TSC and the customs is carried out by ITT-transport.

Interfacing the Container Terminal Models to the FAMAS Simulation Backbone

eM-Plant, the package in which these models are developed, supports the design and development of monolithic simulation models, however, due to the fact that it also supports Dynamic Link Libraries (DLLs), the package can be accessed by external applications. This is necessary in order to achieve time synchronization and data exchange with other models through the FAMAS Simulation Backbone. Being a simulation, the container terminal model is scheduled through the backbone. Through messages received through the backbone internal events are scheduled, e.g. arrival of new trucks. In order to interface to the backbone we applied the external and internal middleware we used in the previous project presented in Chapter 5, Section 5.5.2. We did not need to apply any changes neither to the external nor to the internal middleware presented in the previous chapter. The only change needed in the original model was to make use of the SendFAMASMessage, respectively PeekFAMASMessage, for sending and receiving messages to and from other components. Using this mechanism instead of the original simple truck generator a module was designed that interprets and forwards the received messages from the external truck generator models to the respective internal eM-Plant components.

6.3.2.6 The Planning and Scheduling Model

As mentioned before, for this project a planning and scheduling system had to be designed in order to schedule the truck arrivals at the container terminal. The requirements for this system were: automated multi-channel communications, support for planning and scheduling, facilitation automated negotiations and eventually strategies that can be personalized. Taking into account these requirements the timeslot negotiations at the container terminal have been implemented as an agent based system. Agent based technologies to be used for designing and developing this system are discussed in (Leenaarts and Kentrop 2003).

In the agent based system, an own negotiating agent takes care of the requests and possibilities of every single participating party, which will be either a truck company or a terminal operator. A request with a desired arrival time from a truck company is sent to the system and is picked up by the representing truck operator agent. The request is passed to the terminal operator agent, who checks the availability of resources in the requested time slot. If there are sufficient resources in the requested slot, the terminal agent confirms the booking to the truck agent and updates the terminal database. If the terminal is fully booked at the requested time slot, the terminal agent will propose another slot, taking into account the bandwidth for negotiation that was sent with the request, the estimated transportation time based on the truck's departure information and information about the deep sea vessel on which the container has to be loaded or from which the container has to be unloaded.

In order to illustrate how the system functions an interface has been developed. Through this interface it can be observed whether a request for a specific timeslot has been rejected and a new timeslot has been proposed by the terminal, the truck company accepts the proposed time slot or tries to get another slot which is better suited.

The system is based on the ILLYAN Agent Framework which is built on top of Java (Arnold, *et al.* 2000). It uses a SOAP (Simple Object Access Protocol) Server and a Web Server for the communication with the other models and the interfaces. The messages are based on Extensible Markup Language (XML) for flexibility and interoperability considerations. The selected database management system (DBMS) is taken from the open source Firebird project[43]. Because the standard JDBC 2.0 protocol is used to communicate with the databases, the system is effectively independent of the DBMS chosen. For visualisation purposes, a Java Swing GUI has been developed to display an overview of the requested and assigned timeslots. An example is shown in Figure 6.15.

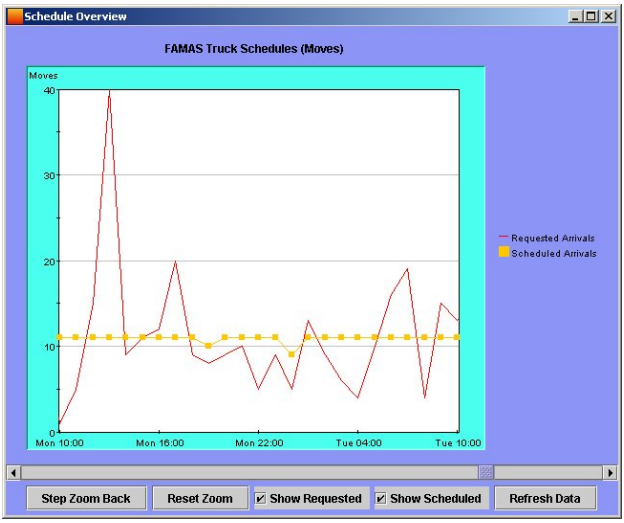


Figure 6.15 Visualization of Assigned Timeslots

Communication with other submodels can be realized through the SOAP server. This is how requests and confirmations for the Truck Generators are handled. As the Planning and Scheduling Model is not a simulation there is no need for it to be scheduled through the backbone. Message passing over the backbone might be feasible thought. Through this mechanism the database describing the state of affairs at the container terminal model might be synchronized for instance. In our final experiment this option has not been implemented (see Figure 6.16).

[43] http://firebird.sourceforge.net/

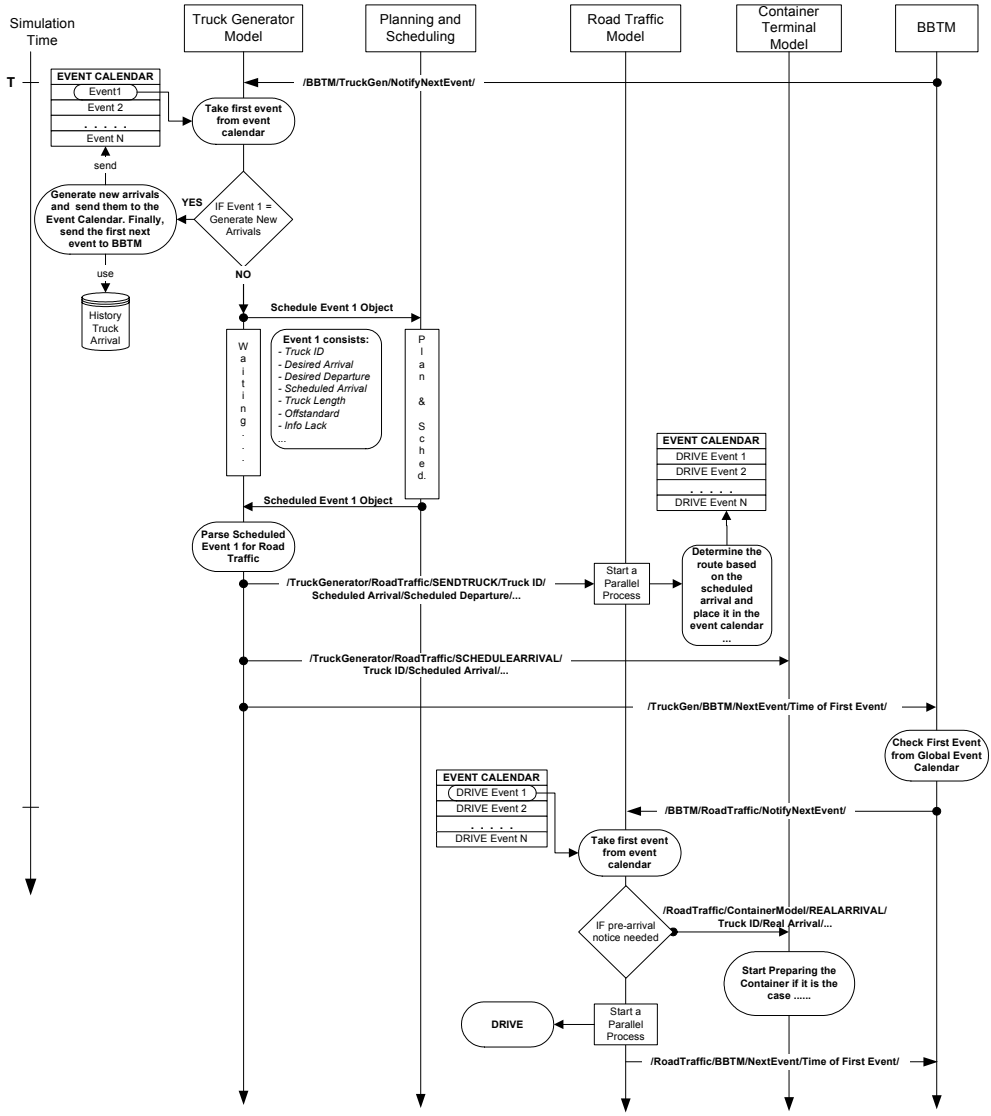


Figure 6.16 Communication Flow between Distributed Components

Figure 6.16 depicts the communication flow between five components during simulation run. As soon as the Truck Generator Model receives a notification from the time management model (BBTM), it processes the first event from the event calendar. This event can be either a request for a truck arrival to the container terminal or an event which generates further requests. If the event is a request for a truck arrival to the container terminal, this request is sent to the Planning and Scheduling Model that checks the availability of requested time. If the requested time can be scheduled, the Truck Generator Model informs the Road Traffic Model about the scheduled departure time and scheduled arrival time, and the Container Terminal Model about the scheduled arrival time. After obtaining the scheduled departure time, the Road Traffic Model simulates and animates the driving process of the trucks. As soon as the truck approaches the container terminal it sends a pre-arrival notice to the container terminal concerning the real arrival time. Possessing this information, the container terminal can generate a copy of the truck with similar parameters at the concerned time.

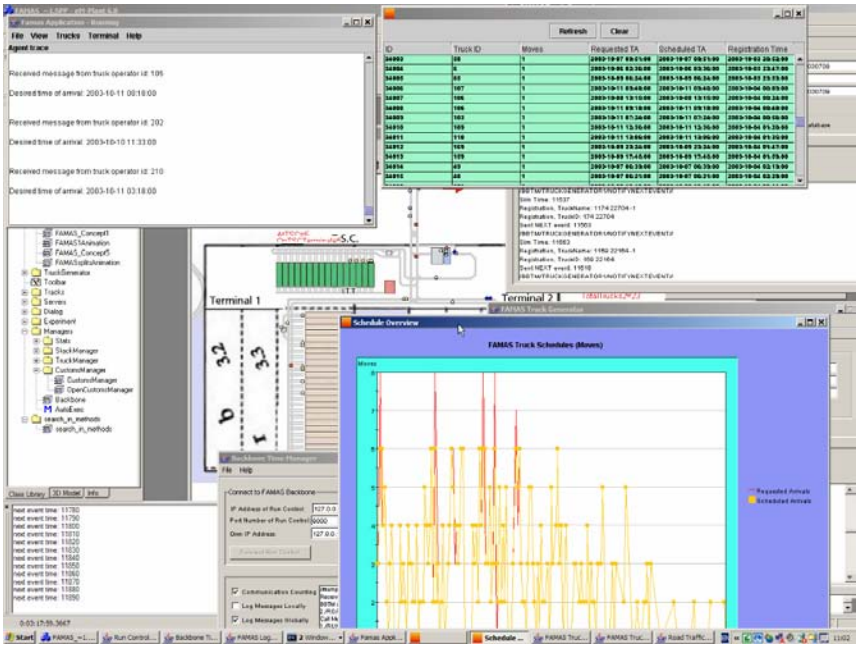


Figure 6.17 Snapshot During Distributed Simulation Run

In this section we have presented the structure and ingredients of the distributed model designed for our project. In the next section we will evaluate how the FAMAS Simulation Backbone Architecture performed on this case. To do this, we first present some issues concerning the experiments we conducted, and then we analyze to what degree is the requirement list presented in the previous section satisfied.

6.3.3 Evaluation

In order to evaluate the distributed model presented in the previous sections we conducted several experiments. Although the main goal of these experiments was to investigate the functioning of the organizational structure of the container terminal, our primary aim with the project described in this chapter was to focus on evaluating the integration process. Therefore, first we just briefly present some of the findings related to the changes needed to the container terminal model and then we focus more elaborately on the model integration aspects.

6.3.3.1 Experimentation with Distributed Simulation Model

The experimentation results of this study have been presented in an end report which was generated for Connekt containing all relevant information (Connekt 2003). We mention some noticeable results.

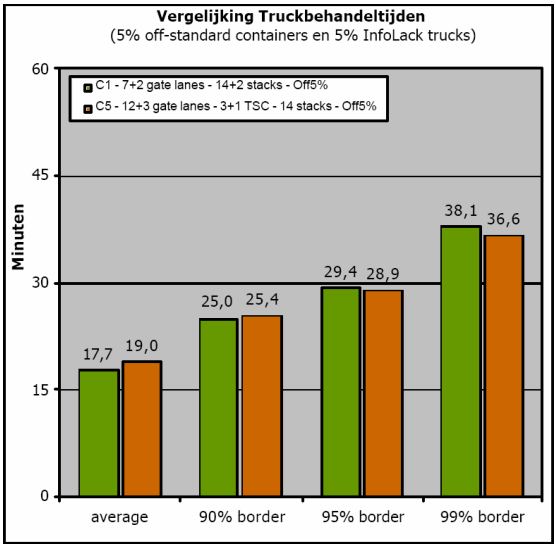


Figure 6.18 Comparison of Handling Times (Connekt 2003, pg. 58)

First of all, regarding the handling time, we found that there is no noticeable difference between the two different terminal concepts mentioned above (Figure 6.18), with the exception that Concept 5 is somewhat more reliable. This concept seems to be better suitable when applying the time window concept for planning and scheduling the arrival of the trucks (Connekt 2003). Further, both terminal concepts are designed to serve 95% of the visiting trucks in 30 minutes. The average handling time of trucks in Concept 1 is well over a minute shorter than in Concept 5, however, the variation in handling times is smaller in Concept 5. Furthermore, 99% of the trucks in Concept 5 are treated within 36.6 minutes, whereas in Concept 1, 99% is treated within 38.1 minutes.

From experiments we concluded that for example, in Concept 5 an additional standard stack module was required, and that an extra off-standard stack module was needed in Concept 1. The additional stack module for off-standard containers resulted in a

considerably lower efficiency of the off-standard stack modules already available, therefore three extra gate lanes needed to be added as well (Connekt 2003).

6.3.3.2 Evaluation of Simulation Model Integration

Since in this thesis we deal with integration of simulation models we focus on the aspects concerning simulation model integration related to the experiments conducted with the distributed model.

Figure 6.19 Final Collaborative Experimentation (Connekt 2003, pg. 76)

One of the most remarkable activities in integrating was carried out in Delft where the representatives of the organizations involved came together to couple the various components (Figure 6.19). In order to investigate the integration of various models we used several computers interconnected through a large bandwidth Intranet network (Figure 6.20).

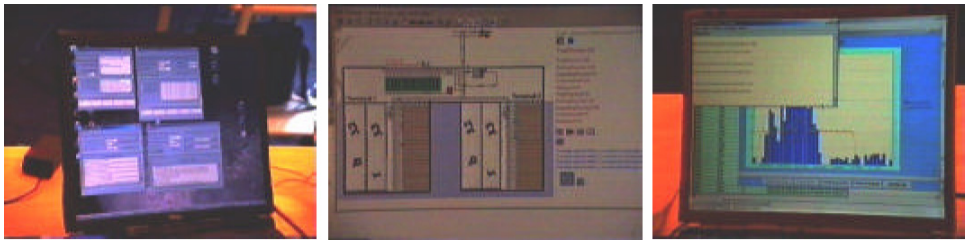


Figure 6.20 Snapshots During Experimentations on Several Computers: Backbone Technical Components and Road Traffic Simulator (a), Simulation of the Container Terminal (b) and Agent Based Planning Component (c) (Connekt 2003, pg. 76)

As we discussed before, in order to integrate the separately designed and developed models for this project, we used the FAMAS Simulation Backbone Architecture. Therefore our primary aim is to evaluate the backbone with respect to this project. The evaluation is based on the FAMAS Backbone requirements presented in Section 5.3 in the previous chapter.

In this case most of the technical components of the backbone were already available from earlier projects. This case did not require modification of the structure of the FAMAS Simulation Backbone Architecture, and further, it did not require modifying or adapting the existing technical components. So, in this project the FAMAS Backbone Architecture has been applied without change.

Accordingly, in this case, the backbone needs to be considered from the user perspective only, that is as it is applied to integrate the separately designed and developed models. As stated above, because there was no need for modification or adaptation of the structure of the backbone itself or any of the technical components, the requirements from the developer's and the administrator's perspective are not relevant to evaluate. Accordingly, we evaluate the backbone only according to the user requirements (*U* requirements) presented in Section 5.3.1.

U1. *Quality*. The quality of the backbone was proven to be adequate for this case. The partners involved in this project could easily apply the FAMAS Backbone Architecture to integrate their separately designed and developed models. For example the organization that focused on planning and scheduling (ILLYAN) could easily integrate its tool to the backbone. Similarly, the organization that reused the container terminal model (TBA Nederland) could straightforwardly eliminate its initial internal simple truck generator, and instead include internal eM-Plant middleware that provides support to couple an external truck generator through the backbone. Since the container terminal model already existed the only task was to interface to the backbone. This interfacing again involved minimal effort because the terminal model was implemented in eM-Plant and for this package we already had internal and external middleware. The only change that was needed was a slight modification in the model to create the relation with internal middleware. The other models were designed and developed from scratch and during design and development we took into consideration the interfaces to the backbone. For interfacing we followed the reports presented for the FAMAS Simulation Backbone Project (Boer*, et al.* 2002b), (Veeke*, et al.* 2002). So, using the FAMAS Backbone Architecture the integration either for the case where we reused a model or where we developed from the scratch, incurred minimal additional work and an acceptable cost.

U2. *Reusability*. As discussed above several submodels existed beforehand and have been reused during the project, like the simulation models of the container terminals (both concepts). The internal parts of these models did not have to be modified for this reason. The only change that has been made was to replace the simple truck generator module with external modules that generate the truck arrivals, and this was needed for experimental reasons. This change again entailed an acceptable cost which was much less than reimplementing the model from the scratch.

U3. *Structure transparency*. The backbone did not pose barriers to the participants when they integrated their models. Furthermore the backbone did not require additional technical knowledge to integrate the models. For those partners that reused models created in COTS simulation packages the previously existing internal and external middleware provided great support for hiding the notions behind distributed simulation. In order to integrate the eM-Plant container terminal models we have used the internal and external eM-Plant middleware developed for the previous project. Using the middleware described in Chapter 5, it was easy to connect the detailed eM-Plant terminal models to the FAMAS backbone. As we have seen the middleware, both external and internal, did not require any modifications. Further, partners who created their models in general programming languages used the FAMAS message protocol presented in Appendix E to interface to the backbone.

An example of such a model is the ILLYAN agent based framework for planning and scheduling the timeslots which was implemented as a Java application. Interfacing tests showed that all information exchange and synchronisation took place as indicated (Boer*, et al.* 2003).

U4. *Support for heterogeneous COTS simulation packages.* The previous case already proved that the backbone supports models created in heterogeneous COTS simulation environments. The second case strengthens this conclusion by showing that, besides container terminal simulation models created in eM-Plant, the backbone supports integration of planning and scheduling tools, road traffic models and simulated truck generator models implemented in the Java programming environment.

U5. *Centralized view for conducting distributed simulation.* See evaluation for Case Study 1.

U6. *Acceptable speed of simulation runs.* Regarding the speed of simulation we have the same observation as discussed in the first case. Due to the fact that the time synchronization is conservative the distributed simulation run is slower than the slowest model in the federate. As we suggested before one solution to avoid this is to apply optimistic time synchronization in order to obtain parallelism, however, as it stands now it would be impossible to apply this synchronization mechanism for models implemented in COTS simulation packages. On the other hand the backbone is open for such a solution.

U7. *Scalability.* We did not conduct measurements concerning the scalability of the backbone, so we can only draw the same conclusion we did in the first case. Our expectation is not too much because of the applied conservative time synchronization mechanism. When we apply conservative time synchronization there is only one model active at a certain simulation time. Therefore, increasing the number of components decreases simulation speed. The only solution to avoid this problem is to apply an optimistic time synchronization mechanism.

A noticeable lesson we have learned from this project is that even if a reliable distributed simulation architecture exists, regular conventional meetings for collaborative distributed simulation studies are always needed. During meetings in the design and development phase the participants agreed on the interfaces between the models and on the interoperation that should take place between the models. These models were tested beforehand either in a standalone way or in a simple distributed testing environment. Despite of these tests and agreements of interfaces, at the final integration test (see Figure 6.19) the well tested models seemed to be “unworkable” at first sight as far as interoperation with other models concerned. Regarding the backbone everything seemed to be working correctly. As a consequence, some final adjustments concerning the interoperability were needed to be implemented by the designers and developers of the models in order to let the integrated models communicate. This problem was a surprise for all of us and would probably never have occurred if we would have had more frequent regular conventional meetings beforehand. As a conclusion we can state that in general the FAMAS Simulation Backbone Architecture facilitated collaboration between the different parties but regular conventional meetings are indispensable.

The distributed modelling approach allows the participants to design and to develop their models in their preferred environment and to keep the internal details of their models confidential. Furthermore it encourages reusability of already existing models. As we have seen before, reusability, heterogeneity, collaborative design and development, and information hiding played an essential role in this project. The FAMAS Simulation Backbone Architecture provided good support to achieve this and this would probably not have been possible if we would have chosen a monolithic approach.

6.3.4 Summary

We can conclude from the case presented above that the project has profited from the FAMAS Simulation Backbone Architecture described in Chapter 5. Evaluating the distributed model depicted in Figure 6.6 we can further conclude that FAMAS Simulation Backbone Architecture in general provided good support for this case. It supported reuse of existing models, integration of models designed and developed in heterogeneous environments, design and development of models in a collaborative way and allowed to hide confidential information inside the model. Although the two cases presented in this thesis are from the domain of container logistics, their evaluation (and the submodels from various areas) suggests that the backbone shall be an appropriate tool for integrating other types of industrial oriented simulation models as well.

Independently of the FAMAS Simulation Backbone Project and the evaluation cases, we have conducted research that we presented in Chapter 3 and 4 the primary aim of which was to identify the requirements for an appropriate distributed simulation architecture for industry. Since the FAMAS Simulation Backbone Architecture seems to be an appropriate tool here, in the next chapter we will investigate to which extent it fulfils the requirements presented in Section 4.5.1. In the investigation we will use the two cases discussed above as well.

7 FINAL EVALUATION AND CONCLUSIONS

7.1 Final evaluation

This thesis started with the observation that, although distributed simulation is accepted and widely applied in defence oriented simulation projects, so far industry rarely applies this way of modelling. We have perceived the absence of an appropriate and acceptable solution for simulation model integration in industry as one of the main reasons behind this phenomenon. During the survey that we conducted, the experts agreed with our observation, suggesting that good technical solutions could create a market for distributed simulation in industry. This reasoning motivated us to formulate our research objective:

> *Provide an architecture for coupling simulation models and test its appropriateness in industry.*

In order to pursue the research objective, we raised three research questions. These research questions were mainly answered in the previous chapters. The aim of this final chapter is to briefly review and summarize the answers to the research questions, and to discuss to which degree we have managed to achieve the proposed objective. We complete the thesis with concluding remarks and suggestions for further research in this area.

7.1.1 First research question

The first research question was aimed at identifying the requirements for an appropriate architecture for coupling simulation models in industry. Accordingly, the first research question was formulated as follows:

> *What are the requirements for an appropriate architecture for coupling simulation models in industry?*

In order to answer this question, we started with a literature survey to investigate what has been done and what kinds of architectures exist so far. We presented and discussed several architectures in Chapter 2 and we concluded that at the moment HLA, a solution developed and used by the defence, is the most appropriate one on the market. This finding suggested that we select and accept HLA as the most appropriate architecture for coupling simulation models and use it for industry. However, next to literature study, from personal communications with experts we additionally perceived that HLA is hardly applied in industry. This observation made us suspicious regarding whether the application of HLA could really lead to achieve our research objective. Therefore, prior to formulating any requirements we decided to test whether our observation holds and if so, whether this observation could make us to think that HLA is not the appropriate solution for which we are looking. In accordance to this reasoning we have formulated the observation as a hypothesis and we tested it by confronting it to a number of experts from the field of distributed simulation. As presented in Chapter 3, for this reason we applied the Delphi methodology, which involved a questionnaire and an interview survey. In order to get a more accurate and unbiased answer, the experts were chosen from various domains related to HLA and simulation: experts who designed and developed the HLA, experts from

defence who apply HLA for daily use, experts from industrial domain having knowledge of distributed simulation and/or HLA, researchers involved in distributed simulation and/or HLA, and finally, COTS simulation package vendors. We asked the experts whether they agree with the hypotheses, we requested them to motivate their answers concerning their opinion, and we asked them to indicate alternative architectures to get an answer to our research questions. The collected data provided us with significant information. By analyzing the data collected during the literature-, questionnaire-, and interview survey we deduced several requirements for an appropriate architecture for coupling simulation models in industry. We identified 25 requirements and structured them in four main groups, as presented in Section 4.5.1. The appropriateness of the requirements was validated by the experts who participated in the survey, as well as by experts who did not directly participate (Table 3.14.).

This list of requirements is not an exhaustive one, it is just an initiative that can be extended and further improved on. Additional requirements that are not in our list, could very probably crop up during the design and development phases of an architecture for distributed simulation, or even while applying the architecture for specific simulation integration problems. Although the set of requirements suggested in this thesis is just an initiative, we believe that it provides a beneficial starting point for stimulating a technical push of distributed simulation for industry.

We certainly do not expect that all the requirements will be fulfilled at once, but we rather expect the design of such an architecture to be evolutionary: first taking into account only the most relevant requirements and skip the less important ones. However, finally one should strive to satisfy all of the requirements in order to encourage the industrial community to recognize the advantages of such a distributed simulation architecture.

7.1.2 Second research question

The second research question was aimed at finding, adapting or designing an appropriate distributed simulation architecture for coupling simulation models in industry and to test the suitability of this architecture with respect to the requirements identified as an answer to the first research question. Accordingly, the second research question was formulated as follows:

> *Can an architecture be found, adapted or designed that satisfies these requirements?*

As a possible solution we have considered a lightweight distributed simulation architecture, called FAMAS Simulation Backbone Architecture, that we designed and developed in order to couple port related distributed simulation models. We presented this architecture in Chapter 5 and we evaluated the implemented architecture through two case study projects in Chapter 6. We concluded that the backbone solution presented was appropriate enough within the research plan for which it was required.

Being appropriate enough for a certain task in a given industrial domain does not entail, however, that an architecture is appropriate enough for a broader industrial area. In this sense, the validation of the proposed solution conducted so far, does not answer the second research question yet, that concerns the suitability of the proposed solution, in this case the FAMAS Simulation Backbone Architecture, in a broader sense, that is with respect to the

requirements identified for the industry in general. Therefore, in order to be able to answer this second question, and point out the benefits industry could gain from such an architecture, next we take one by one all the identified requirements and evaluate to what extent satisfies the backbone them.

Requirements related to the characteristics favouring distributed simulation projects

1. *The architecture should enable reusability of already existing simulation models.*

   Reusability of already existing models is a central benefit of general distributed application projects because, as McIllroy's law states, it reduces cycle time and increases productivity and quality of the projects (McIlroy 1969). As discussed in Section 4.2.1, one of the advantages of distributed simulation against the monolithic approach is that simulation practitioners can reuse already existing models in combination with other ones. By reusing models is meant that no effort or not too much effort is required to modify the existing models in order to integrate them with other distributed models. The FAMAS Simulation Backbone Architecture enabled the participants to reuse existing models by providing well defined messages and middleware discussed in Sections 5.4.2 and 5.4.4. Reuse of existing models was demonstrated in both case studies: in the first case the MICL model was reused, while in the second case the container terminal model.

2. *The architecture should support integration of heterogeneous simulation models.*

   By heterogeneous simulation models we mean simulation models designed and developed in different simulation packages (see Section 2.3.2 and 4.2.2). Simulation model integration as discussed in Section 2.5.1 involves: data exchange and time synchronization. Consequently, the architecture designed to integrate heterogeneous simulation models should support data exchange and time synchronization between simulation models designed and developed in different simulation packages. As illustrated by the case studies and discussed during the evaluation of the *U4* requirement (see Sections 6.2.3 and 6.3.3), heterogeneous simulation models being developed in eM-Plant, Arena, TOMAS, and Java, were integrated through the backbone. There, data exchange was based on the message protocol presented in Section 5.4.2 and Appendix A, and time synchronization was solved by the Backbone Time Manager discussed in Section 5.4.3 and 5.5.1.

3. *The architecture should facilitate collaborative simulation design and development.*

   Collaboration, as Keen defines, is "a joint commitment to a target output, with team members sharing authority and responsibility as needed, at different stages and for different tasks" (Keen 1991, pg. 110). During collaboration "two or more individuals with complementary skills interact to create a shared understanding that none had previously possessed or could have come to on their own" (Schrage 1995, pg. 33). Participants collaborated in the second case study for the development of which several functional components had to be designed, developed, reused and integrated. Each involved organization was responsible for a certain functional component, such as the simulation model of the container terminal, the model of planning and scheduling system, and the model of road traffic system. As a consequence,

collaborative meetings were needed to agree on the data exchanged and the way the data has to be exchanged, that is the form of interfaces between the different models. Middleware and well defined message protocols provided by the backbone supported the participants from various domains to "talk a common language", avoiding the need for too frequent communications, and saving time. The backbone facilitated, in this way, straightforward collaboration between different participants.

4. *The architecture should allow for information hiding between simulation models.*

   Information hiding within simulation models, as discussed in Section 4.2.4, refers to protecting two issues: intellectual property (e.g., special algorithms) or sensitive internal data (e.g., company or client information). Information hiding played an important role in the second case, since the organization that has built the planning and scheduling model was not willing to share the internal code of the algorithms. Indeed, using the backbone we managed to obtain a workable solution by exchanging only non sensitive data.

Requirements related to the technical design of the architecture

Service Oriented Structure

5. *The overall functionality should be organized as a set of services.*

   The execution of a distributed simulation model in contrast to a monolithic one requires additional functionality, such as data exchange or time synchronization between models. Since the currently available COTS simulation packages neglect such functionality, in order to connect two COTS based simulation models and to allow distributed execution, required functionality should be organized and externally provided as a set of distributed simulation services. The FAMAS Simulation Backbone Architecture, as discussed in Section 5.4.1, defines technical components which provide services to the distributed simulation models. The distributed service oriented component-based structure of the backbone is depicted by Figure 5.1. In this sense the backbone is organized as a set of services and the different services are well defined and separated from each other. For example, the Run Control component provides services for starting, terminating, monitoring a simulation run, the Backbone Time Manager for time synchronization, the Logging component for data collection and the Visualization component for animation.

6. *The services provided should be COTS package independent.*

   Since the services mentioned above should be widely applied for a large number of COTS simulation packages, they should not be defined in terms of concepts implemented in a specific COTS simulation package. The backbone satisfies this requirement, since the message oriented structure that it uses to invoke and deliver services is independent of any COTS simulation package. The implemented technical components are independent of any COTS simulation package and programming environment, providing services for all the functional components without reference to the package in which they were developed. Illustratively, in both case studies we used several models designed and developed in different COTS simulation packages, such as Arena, eM-Plant and TOMAS.

7.1 Final evaluation

7. *The services should be separated from the simulation models.*

   The structure of the FAMAS Simulation Backbone Architecture, as discussed in Section 5.4.1 and depicted by Figure 5.1, is designed in such a way to separate the functional components developed by end users from the technical components that provide services for the functional ones. In this sense distributed simulation services are completely separated from the simulation models.

8. *The architecture should provide at least three basic groups of services. That is:*
   - *Services for starting, stopping and periodically checking the execution of the collection of distributed simulation models.*
   - *Services for data representation and exchange.*
   - *Services for time synchronization*

   Based on the interview with the experts we identified the above minimal set of services, which currently should be sufficient for the industry. The architecture presented provides all of them. For starting, stopping and periodically checking the execution of the collection of distributed simulation models the FAMAS Simulation Backbone uses the Run Control component, which is discussed in Section 5.4.3.1 and 5.5.1.1. For data representation it uses the Scenario Object discussed in Section 5.4.3.5, while for data exchange it applies the message protocol presented in Section 5.4.2. Finally, for time synchronization the Backbone Time Manager technical component is used as described in Section 5.4.3.2 and 5.5.1.2.

Flexibility

9. *The distributed simulation architecture should be extendible in the sense that it should allow simulation vendors and practitioners to provide additional services that are specific for their tasks.*

   Experts agreed that a distributed simulation solution should be open to incorporate new services in order to be flexible and applicable for a large scale of problems. As discussed in Section 5.4.1, one of the advanced features of the backbone is that we can extend its set of technical components with new ones. We have initially defined a minimal set of technical components for providing only the set of services mentioned in the 8th requirement above. However, when the case studies needed, additional technical components have been designed and developed, such as the Logging and Visualization components.

10. *The distributed simulation architecture should allow for replaceable elements (e.g., it should allow simulation vendors and practitioners to replace already existing functionality with more efficient ones).*

    As discussed in Section 4.5.1, elements should be replaceable in order to allow for further improvements of the functioning of the architecture. In this context the replaceable feature means that the specification of the services does not change. The backbone satisfies this requirement, this feature being illustrated in the first project in which we replaced the time manager with another one, implemented by a different group, in a programming environment that was different from the original solution (see Section 5.5.1.2).

Transparency

11. *The architecture should hide low level implementation details from the simulation practitioners.*

    The backbone is intended both for novice and experts in distributed simulation. The extendible and replaceable features discussed above are mainly meant for experts who aim to add new services or to improve existing ones. However, the backbone aims to support users, who just want to apply the available technical components without having an interest in their internal implementation. This situation was clearly shown in the second case study.

    In Section 5.4.4.2 we presented internal and external middleware, through which the end user can approach the services provided by the backbone. The internal middleware exposes an interface to the modeller in terms of high level concepts from the COTS simulation framework. As a consequence, understanding and applying it does not demand additional programming knowledge for the user. The low level issues regarding distributiveness are implemented in the external middleware which is a bridge between the internal middleware and the backbone. In a full implementation of this idea the external middleware and the simulation backbone will remain hidden from the end user who just needs to use the internal middleware in order to integrate the models participating in a project. Our test case showed that we were able to realize this desired state of affairs in eM-Plant. In Arena we were not able to fully reach this goal in the sense that the internal and external middleware sometimes had to be adapted to integrate new models. However, the adaptations were relatively minor and located in a well defined place inside the middleware code.

Data representation and exchange/communication

12. *Communication used by the architecture should be based on one of the standard communication protocols supported by most COTS simulation packages.*

    The backbone as discussed in Section 5.4.2 is built on the TCP/IP standard communication protocol. As Table 3.5 presents, this protocol is also supported by 8 COTS simulation packages out of 18. For packages that do not support TCP/IP protocol, middleware (e.g., a DLL) can be used, which implements the connection to the TCP/IP protocol (see Section 5.4.4). This latter solution is illustrated in Section 5.5.2. On top of the TCP/IP we defined the message protocol for coupling models via the backbone, which is presented in Appendix E.

13. *Representation of the data and object model and data exchange should be based on standards or quasi standards which are defined by the industrial community. This industrial community should be represented by simulation practitioners, simulation researchers and COTS simulation vendors.*

    Although there is a recently initiated product development group under SISO that is aimed to develop standard representation for data exchange and exchange mechanism, when designing and developing the FAMAS Simulation Backbone Architecture there have been no standards or quasi standards available for interoperation between industrial simulation models. So, we could not meet this requirement.

7.1 Final evaluation

Performance

14. *The architecture should perform efficiently when simulation models interoperate with each other.*

    During the evaluation of the backbone through the case studies we have already referred to performance issues (see evaluation of *U6* requirement in Sections 6.2.3 and 6.3.3). Concerning the speed of simulation runs, we observed earlier that, we can obtain efficient performance only if the simulation models run in parallel. In order to run them in parallel we need to use optimistic time synchronization mechanism. As it stands now, an implementation of an optimistic time synchronization mechanism when integrating simulation models developed in COTS simulation packages is a big challenge and there is ongoing research in this direction (Wang*, et al.* 2004). For our purpose we have chosen conservative time synchronization which could be easier implemented for the simulation models created in COTS simulation packages. Using this time synchronization mechanism we obtained acceptable results during the demonstrations, in the sense that the audience offered positive feedback concerning the speed of simulation run. Of course, the backbone can accommodate technical component implementing optimistic time management.

Scalability

15. *The architecture should be scalable in the sense that it should handle hundreds of coupled simulation models in a relative efficient way.*

    We cannot draw definite conclusion concerning the scalability of the backbone, because we have not done explicit measurements. However, we have some remarks concerning this issue. The structure of the backbone is designed and developed in such a way that direct peer to peer communication between the components is established. This approach minimizes communication over the backbone compared for example to approaches that send all the information through a technical component that is responsible for data distribution, behaving like a switching board, that can be easily overloaded if the number of integrated components increases. Due to peer to peer communication, our expectation is that the backbone eliminates this possibility to be overloaded. Although the structure of the backbone suggests that the scalability of the architecture is not a problem, the applied conservative time synchronization suggests the contradictory. As discussed in the 14th requirement, the fact that during conservative time synchronization only one model is considered as "current" at any moment, the integration of hundreds of simulation models will probably entail decrease in performance.

16. *Besides coupling simulation models, the architecture should enable coupling of external applications or real equipment.*

    Next to supporting the integration of heterogeneous simulation models, as discussed under the second requirement, the integration of real equipments for advanced experimental purposes and the integration of applications designed and developed in general programming languages that provide specific algorithms could be often required by distributed projects. As illustrated by the case studies, the backbone satisfies this requirement. In both case studies the backbone supported the integration

of models designed and developed in different simulation packages, such as Arena, eM-Plant and TOMAS. The support for external applications is illustrated in the second case, in which we integrated a planning and scheduling application implemented in Java. Furthermore, in (Boer, *et al.* 2002d) we have presented the possibility to apply the FAMAS Simulation Backbone to integrate real equipment. An application integrating simulation models and real equipment through the backbone was presented at the Logistica 2003 exhibition[44] by TBA Nederland. They have integrated a miniature real Automated Stacking Crane which was controlled by an eM-Plant container terminal simulation model through the FAMAS Simulation Backbone Architecture.

Miscellaneous

17. *The architecture should be based on TCP/IP as the main protocol for Internet.*

    The Internet is defined as a network of many networks that interconnect worldwide and use the TCP/IP protocol. The FAMAS Simulation Backbone Architecture is based on TCP/IP, which is the basic communication protocol for the Internet. Using Internet connection both case studies were tested in the SimLab[45] with positive results.

18. *The architecture should incorporate fault tolerance mechanisms.*

    As it stands now, the FAMAS Simulation Backbone Architecture does not incorporate any mechanisms to ensure fault tolerance. Consequently, if "somebody cuts the network wire" the distributed simulation stops and it needs to be restarted. However, if the Logging component was used, data collected during the simulation run before the fault can be retrieved from the database.

Requirements connected with the relation to COTS simulation packages

19. *The architecture should support integration of simulation models designed and developed in a large variety of COTS simulation packages.*

    Supporting a large variety of simulation packages is necessary in order make the architecture to be accepted by the industrial community using several packages for different problems and areas. In the above cases we have shown the suitability of the backbone for supporting the interoperation of models written in three simulation packages, namely Arena, eM-Plant and TOMAS. For other packages integration can be solved in similar way, either directly using FAMAS messages or through middleware.

20. *The services offered by the distributed simulation architecture should be naturally expressible in terms of concepts used in COTS simulation packages.*

    For each applied COTS simulation package, as we discussed in the evaluation of the 11th requirement, we used an external and internal middleware, through which the end user can approach the services provided by the backbone (see Section 5.4.4.2). In

[44] http://www.logistica-online.nl/

[45] http://www.simlab.tbm.tudelft.nl/

order to couple their models to the backbone, modellers have to use the internal middleware which exposes an interface to them in terms of high level concepts from the COTS simulation package. By using this interface, in form of building boxes provided by the COTS simulation packages (see Figure 5.23 and 5.24 in Section 5.5.2), modellers can achieve thus, time synchronization and data exchange with other models.

Requirements related to the efforts needed from the simulation practitioners

21. *Building a COTS simulation model which is aimed to be used as a submodel in a distributed simulation should not require more effort and knowledge than building it as a standalone model.*

    As discussed in the 11th requirement, through suitable internal and external middleware the low level details of distributed simulation will remain hidden from the end user. Nevertheless, having an internal middleware which is expressed in high level COTS simulation package terms, the user still has to know what data he can send to, and expect from other models. The backbone does not and cannot completely satisfy this requirement because developing a model that will participate in a distributed simulation demands little more effort, even having proper internal and external middleware, than developing a monolithic model. As far it stands now we do not see any solution that can achieve this requirement. What we can do however, is to try to decrease the additional effort.

22. *The effort from the simulation practitioner needed to setup and manage an execution run of a distributed simulation model should be minimal.*

    The effort needed to setup and manage an execution run of a distributed simulation model should be minimal as compared to the effort needed to execute a similar monolithic simulation. To run distributed simulation using the backbone the user has to start the Run Control specifying a scenario file that describes a Scenario Object. Next, the user can start all technical and functional components in any order. When all components are connected to the Run Control component, the distributed simulation starts running. As we can see, setup of a distributed simulation run involves additional work, however, this is a direct consequence of the fact that the system to be run contains a number of submodels, the existence of which the user is aware of.

23. *The architecture should provide a graphical user interface (GUI) facility that allows the simulation practitioner to configure and to monitor a simulation run.*

    This requirement refers to an easy configuration of a simulation run. As discussed in Section 5.5.1, for all technical components we implemented a graphical user interface that enables the simulation practitioners to configure and to monitor a simulation run. Thus, the backbone satisfies this requirement.

24. *The architecture should allow or even facilitate the possibility to debug the simulation.*

    Concerning this issue, the only facility the FAMAS Simulation Backbone provides is that exchanged messages can be monitored. There are two ways to do this. One possibility is that a component can visualize the exchanged messages through its

graphical user interface, the other is that it can send messages to the Logging component.

25. *The user should be able to access all simulation results in a consistent format in order to examine them after completion of a simulation run.*

    The backbone satisfies this requirement through the Logging component. The Logging component, as discussed in Section 5.4.3.3 and 5.5.1.3, enables the user to collect data from all components in a centralized database that can be accessed and examined any time.

Although there are some requirements, that are not completely or appropriately satisfied by the backbone (e.g., requirements 14 and 15) or requirements that are neglected by it (e.g., requirement 18) or even requirements that currently cannot be satisfied at all (e.g., requirement 13 and 21 – due to the inexistence of a standard or quasi standard), we can state that most of the requirements are satisfied by the backbone. Reviewing the requirements identified and stated in Section 4.5.1, and analyzing to which degree satisfies the presented architecture them, we can conclude that the FAMAS Simulation Backbone is a lightweight distributed simulation architecture, originally designed for port related simulation models, that can be, however, considered as a possible solution for a broader industrial area.

We do not state that the FAMAS Simulation Backbone Architecture is better than HLA. We should always keep in mind that HLA is an accepted standard for distributed simulation and it is widely applied in defence. However, as we observed, it is neglected for some reasons by industry, the domain on which we are focusing. We investigated, identified and analyzed reasons why HLA is disregarded in industry, and we built the set of requirements for the industry based on these reasons. We believe that alternative approaches can be developed in order fulfil this set of requirements.

FAMAS Simulation Backbone is such an alternative approach proposed and presented in this thesis. The backbone is a more simple solution for integrating distributed models than HLA, its simplicity follows, however, from the fact that it was originally intended for less complex problems than HLA. HLA is a good approach for defence community and is proposed for complex defence-oriented distributed simulation. Industrial projects, however, are mainly characterized as small or middle size projects and these might not benefit from HLA, be it only on cost consideration. Industrial projects mainly deal with less complex solutions, and, as a consequence, practitioners from the industrial area are not used to create models on the level on which they are required by HLA. Instead, they apply high level COTS simulation packages. The FAMAS Simulation Backbone was mainly focusing on these aspects, and tried to provide a possible solution for industrial applications. From this point of view we believe that a lightweight approach like FAMAS Simulation Backbone is more appropriate for industry at the moment. Consequently, taking into account that the backbone satisfied the requirements to a respectable degree, for our purpose, we accept the FAMAS Simulation Backbone Architecture as a possible candidate that is appropriate to integrate industrial oriented distributed simulation models.

7.1.3 Third research question

The third research question was aimed at testing the practical applicability of the architecture as answer to the second research question. Accordingly, the third research question was formulated as follows:

Is it possible to successfully apply this architecture in practice?

The FAMAS Simulation Backbone Architecture, as presented in Chapter 6 was applied in two industrial projects. Although the first project was mainly aimed to evaluate the backbone, the second project applied the backbone as a 'solution' to couple heterogeneous simulation models developed (and reused) by different organizations. Since both the collaborative organizations and the problem owner were satisfied with the end results, which was underlined during the presentation[46] of the end product in 2003 October at Gemeentelijk Havenbedrijf Rotterdam, see (Connekt 2003), our answer to the third research question is affirmative.

7.2 Conclusions

In this section we aim to discuss to what degree we managed to achieve the stated research objective. Analyzing the three research questions and the answers provided we can state that with this research we have accomplished the followings:

- We have identified a list of requirements that an appropriate architecture for coupling simulation models for industry should satisfy (Chapter 3 and 4).
- We have designed and implemented a lightweight architecture, called FAMAS Simulation Backbone Architecture, for coupling simulation models for industry (Chapter 5).
- We have tested the applicability of the FAMAS Simulation Backbone Architecture through two industrial projects, and we have found together with the involved participants that the backbone was suitable for the purpose (Chapter 6).
- We have evaluated the appropriateness of the proposed architecture for coupling simulation models for industry, that is FAMAS Simulation Backbone Architecture, with respect to the list of the identified requirements. We have found that almost all the requirements are fulfilled by the backbone (Chapter 7).

Based on the above conclusions we can state that we have achieved our research objective.

In order to pursue our research objective we followed an evolutionary learning phase, in the sense that we expanded our scientific knowledge by adapting it based on new observations. The question we would like to analyze next is: what have we learned from this research?

[46] The author of this thesis has also been participated at this final presentation and he had personal discussion with the participants concerning the appropriateness of the backbone in this project.

First of all, we managed to obtain a deep insight into the application of distributed simulation in industry. We perceived that the reasons behind the unpopularity of distributed simulation in industry, compared to defence, are primarily market related. The market reason has to do with the cost-benefit ratio. We believe that currently there is disequilibrium between the cost and the sensed benefit of applying distributed simulation in industry. We analyzed and discussed this ratio in Chapter 4. What distributed simulation, additionally requires, compared to monolithic simulation, is design and development of the models with distribution in mind and an architecture to couple these models. We analyzed the extra costs for using distributed simulation solutions instead of monolithic ones along three dimensions: time, monetary costs and quality.

Concerning time we observed that distributed computing in general is too complex for the simulation practitioners who are not used to it, and therefore, it requires a lot of effort to learn to deal with it. Time spent to learn to deal with this complexity entails unforeseen costs that might be too high related to the expected benefits. In order to reduce the complexity we designed and developed middleware through which we hide low level details of distributed computing from simulation practitioners.

Time spent to design and develop models entails monetary costs. Additionally, there are direct monetary costs involved, i.e. the purchase price of an existing architecture to couple models or the monetary costs of developing such an architecture. We believe that purchase price of such an architecture might be too high for some organizations and it certainly adds extra cost to the cost-benefit ratio, however, as it stands now, we see that there are more important issues and other type of costs than the price of the architecture. In spite of the fact that the DMSO RTI is a free architecture that exists since 1996 almost none of the COTS simulation package vendors applies it as an additional feature for their customers. This indicates that the purchase price is not the main reason for not applying distributed simulation. Instead the effort needed for integrating the concepts of a specific distributed solution into a COTS simulation package, and further, hiding the low level implementation details of these concepts from the simulation practitioners represents the higher costs.

Next to the monetary costs and the costs related to time, the quality of the available solutions might hold people back to apply distributed solutions. Although distributed simulation, as we presented in Chapter 4, can provide added value for some projects comparing to the traditional monolithic solution, the end solution can be often qualitatively deficient in some aspects. One of these qualitative aspects is performance. It is true that distributed simulation theory has already brought about efficient algorithms, some of them being discussed in (Fujimoto 2000), however they still need to be implemented. As we discussed, the FAMAS Simulation Backbone Architecture provides conservative and time stepped synchronization mechanisms. Since we wanted to support the coupling of simulation models developed in COTS simulation packages we were forced to support these time synchronization mechanisms. The implementation of an optimistic time synchronization mechanism, which would perform much better than the supported ones, makes in most cases no sense because of the constraints imposed by the COTS simulation packages, within which this sort of time synchronization cannot be applied at the moment. So, it is always a trade-off needed, however, in this way developers of distributed solutions loose quality when choosing a solution based on the features of the focused COTS packages. One approach to avoid this problem is to open somehow the COTS simulation packages. Currently, as we discussed, these packages are closed in the sense that they do

not allow access to internal variables which are essential to conduct distributed simulation. Such a variable is, for example, the event calendar which is needed during time scheduling. Unfortunately, as from interview survey described in Chapter 3 results, COTS package vendors seem to maintain a customer lock situation by developing their own package to provide more functionality, instead of integrating it with other packages. This was explicitly mentioned by simulation experts as well, who stated that COTS simulation package vendors do not want to take too much initiative because they are just trying to avoid that somebody uses other systems than theirs.

In this thesis we analyzed how two models implemented in different COTS packages can interoperate, and we discussed some inconsistency problems that might occur during distributed simulation run. We also mentioned the semantic inconsistency problem but we did not go in too much detail, as semantic inconsistency is a big challenge not only for simulation community but for the whole software engineering community. The unsolved problem of semantic interoperability when aligning data models is in fact, another important reason that hinders developers to apply distributed simulation. Fujimoto argued that semantic interoperability is a big challenge, a really hard problem, probably much harder than the problem for which HLA was designed and developed. In Chapter 4 we refer to this problem outside of simulation community, namely the web community. This community is much larger than the simulation community and still facing this problem. Future tools for semantic interoperability could help to eliminate the limitations of keyword based matching techniques. Although this is a big challenge for distributed simulation community, we expect that this problem will be solved by general software engineering communities. In this sense the FAMAS Simulation Backbone, in the same way like HLA, does not provide any support for semantic interoperability.

HLA played an important role in this research. Although, it seems that HLA is the most appropriate architecture for distributed simulation, and even an accepted standard, we observed that it is hardly applied in industry. This thesis provided a comprehensive evaluation of this observation stated in form of a hypothesis that was confronted with a number of experts in this field. The experts accepted the hypothesis motivating their opinion. Motivations in this way led us to a number of reasons that can explain why industry does not apply the HLA standard (see Chapter 4). The first reason is that HLA standard is too complex for industry. Complexity especially refers to understanding the concepts behind this standard and using it for distributed projects. The second reason is that HLA standard is not transparent enough for the industry, transparency being related to the way distributiveness is hidden from the user. The third reason is that the current implementation of the HLA standard, especially the DMSO RTI, is too inefficient for industry.

We designed and developed the FAMAS Simulation Backbone Architecture with these reasons in mind and we strived to avoid these deficiencies. By designing and implementing the backbone we achieved a less complex, or as we called lightweight, architecture, and with the middleware approach within it we managed to make the backbone more transparent for the end users. However, the performance inefficiency is still a challenging task due to the limited possibility to interface COTS simulation packages.

Although HLA is an accepted standard for distributed simulation and frequently applied in defence, it seems that more architectures or standards might deserve a place. We do not

really believe in a universal standard which can solve all problems, instead it is important to design and develop the most appropriate tool for a certain set of problems. As we observed, large numbers of industrial projects are characterized as small or middle size projects and these might not benefit from HLA, be it only on cost consideration. We think that, as it stands now, industrial applications require less complex solutions and by keeping the complex HLA implementation as a basis of the architecture, inherent complexity will remain. In this sense we believe that a more lightweight distributed simulation architecture, like the FAMAS Simulation Backbone Architecture, that will eliminate some of the complexities and would be more suitable for COTS simulation world (industrial world) would have its place besides the implementation of the HLA standard.

The approach applied for designing and implementing the backbone differs from the approach applied for designing the HLA. The concepts and approaches that are additionally used in the backbone compared to the HLA are the component-based structure, peer-to-peer communication, possibility for scenario definition, and middleware for the COTS packages. The component-based structure allows to easily replace, exchange, and extend technical components that provide common tasks for simulation models. The direct peer to peer communication between the components minimizes communication over the backbone compared, for example, to approaches that send all the information through a technical component that is responsible for data distribution, behaving like a switching board that can be easily overloaded if the number of integrated components increases. Further, various distributed simulation runs can be specified in the backbone with the help of the Scenario Object. The simulation practitioner can use a simple script language for describing a scenario for each distributed simulation run. In this way he can trigger some events that should occur in a certain simulation time (e.g., the weather is changing and decrease the number of vehicles within the container terminal). Finally, the various middleware created to the backbone allow coupling simulation models created in different COTS packages, such as Arena, eM-Plant or TOMAS.

An important conclusion of this research is that lightweight solutions for coupling models are possible and, as the results of the two cases proved, the participants and the problem owners were pleased with such a solution. We need, thus, lightweight architectures like the FAMAS Simulation Backbone Architecture, as we see them as a possible way to escape from the chicken-egg scenario described in Section 4.4, according to which: on the one hand simulation practitioners do not see the benefits of distributed simulation because they do not have a tool to experiment with, and therefore they do not request tool vendors for a tool that supports distributed simulation; and on the other hand, tool vendors do not provide a tool because end users do not request it. Nevertheless, this situation seems to change. Experts from industry believe that there is room for distributed simulation and for standards that implement the concepts for coupling industrial simulation models. We expect that by providing lightweight architectures that satisfy the identified requirements to a high degree would create a market for distributed simulation in industry.

7.3 Further Research

Although the primary aim of this thesis was to answer the questions that we posed in the beginning, it also points to additional research challenges and implies further questions that should be answered in the future. Several challenges cropped up, in the first place, during the theoretical research presented in Chapter 3 and 4 as a result of discussing with experts in domain and by analyzing the collected data. Further, the practical experience and the cases we carried out suggested new research opportunities for the future. In the rest of this thesis we would like to elaborate on these challenges and questions recommending them for future research.

One of the research challenges that we recommend is to provide an implementation based on the requirements presented in Chapter 4 on top of an existing HLA RTI. This way of implementation is an alternative solution proposed to achieve the objective, and it was already mentioned and briefly discussed in Section 4.5.2. Having an HLA based implementation, next to the lightweight approach discussed in this thesis, the two solutions could be compared and would provide a more comprehensive picture for the communities involved. For a comparison both solutions should be involved in a number of cases. However, we conducted only two case studies with the FAMAS Simulation Backbone Architecture in only one branch of industry. Hereby we recommend, thus, further experiments with both architectures in other branches of the industry as well. Being a standard, IEEE 1516, an HLA "tailored" for the industrial domain could increase the chance to be accepted and used by the industrial simulation community. There have been and are attempts made in this direction. Different HLA adaptors have been built by different institutes and organizations (Straßburger 2001), (Rabe*, et al.* 2001), (McLean and Riddick 2000) and (Revetria*, et al.* 2003) in order to connect COTS simulation packages through HLA RTI by hiding the complexity of HLA. Unfortunately, however, as we have discussed before, these adaptors are still prototypes still requiring specific knowledge to apply them. The adaptation of these solutions should be continued in the view of the needs of the industrial community, suggested in form of an initiative list of requirements in this thesis. The improved adaptors should be fit naturally within a COTS simulation package environment and should hide details of the distributed solution from the end user. Furthermore, in order to encourage reusability and to avoid inconsistency problems during data exchange, standard reference object models would be needed that could be used by the adaptors. The adaptors could be designed and developed by COTS vendors or third party adaptor vendors, while the standard reference object models could be maintained by non-profit organizations.

The definition of reference object models required to solve the interoperation of simulation models created in COTS simulation packages is a second important research challenge that we would like to emphasize. An initiative in this direction has started recently, in October 2004, under Simulation Interoperability Standards Organizations (SISO)[47] when a product nomination for creating these sorts of reference models has been approved. A research group, called COTS Simulation Package Interoperability Product Development Group (SAC-CSPI-PDG) has been started within the SISO with the objective to identify and build the required reference models. A proposal for the product nomination can be found at

[47] http://www.sisostds.org

http://www.cspif.com. We believe that the research we have conducted can provide support for the identification of these reference object models.

Another recommendation for future research is to provide a mechanism for fault tolerance. This was explicitly mentioned by several experts, who would like to see in the next version of HLA or other standards such a solution which would remedy this problem.

Further, an interesting topic for future research is the design and development of a multi-level hierarchical distributed simulation architecture which involves more than one coupled distributed simulation architectures. This issue was already addressed in (Cai*, et al.* 2001), in which coupled HLA-RTI's were proposed to tackle information hiding between groups of simulation components in a simulation federation. In (Boer and Verbraeck 2002) we have also proposed such a multi-level distributed simulation architecture with the aim to couple HLA compliant models with FAMAS backbone compliant simulation models. In order to enable to couple HLA models to FAMAS Backbone Architecture we have to combine these two architectures. Although we did not implement it, we have proposed an HLA-FAMAS bridge, for coupling these two distributed architectures. The implementation and the test of this bridge would be an interesting research in this area to carry out in the future.

From the questionnaire survey we have concluded that COTS simulation package vendors are looking into the future to support HLA or other distributed simulation architecture in their packages, however, currently there does not seem to be a big drive into this direction. We have observed that, while a couple of organizations intends to accommodate distributed simulation by creating "homespun" architectures based on low level technical solutions, other organizations do not have any intention at all to support connection with other packages. We can understand that the willingness to support the HLA standard, other distributed simulation architectures, or to open up the COTS simulation package, relates to the perceived cost-benefit ratio, we still believe, however, that the willingness from the side of COTS vendors to adapt their packages for supporting distributed simulation would advance to a great extent the distributed simulation in industry. A suggestion to modify COTS simulation packages in order to provide the necessary functions and interoperability required within the package is discussed in (Ryde and Taylor 2004). It would be an interesting future research topic to take one or two COTS simulation packages and investigate in collaboration with the vendors, how distributed simulation concepts can be incorporated within these packages, e.g., supporting optimistic time synchronization between these packages, or supporting the reference models proposed by the CSPIF group. This effort might lead to a next generation of COTS simulation packages.

SUMMARY

Distributed simulation is an application of distributed systems technology that enables models to be coupled over computer networks so that they interoperate during a simulation run. Distributed simulation is widely applied in the military domain, and recently there has been a growing interest to apply it in industry as well. In spite of this, the application of distributed simulation in industry is still in its infancy. We have perceived the absence of an appropriate and acceptable architecture for coupling simulation models in industry as the main reason behind this phenomenon. As a consequence, we expect that the existence of such an architecture could provide a technical push which could pull the distributed simulation into the industrial market. This line of reasoning motivated the research objective of this thesis, namely "*provide an architecture for coupling simulation models and test its appropriateness in industry*".

In order to pursue the research objective, we raised three research questions. The first research question was aimed at identifying the requirements for an appropriate architecture for coupling simulation models in industry. The second research question was aimed at finding, adapting or designing a distributed simulation architecture that satisfies these requirements. Finally, the third research question was aimed at testing the practical applicability of the architecture as answer to the second research question.

In order to answer the first research question, we started with a literature survey to investigate what has been done and what kinds of distributed simulation architectures exist so far. We presented and discussed several architectures in Chapter 2, and we concluded that at the moment High Level Architecture (HLA), a solution developed and used by the defence, is the most appropriate one on the market. This finding suggested that we select and accept HLA as the most appropriate architecture for coupling simulation models and use it for industry. However, from a literature study, and from personal communication with experts we additionally perceived that HLA is hardly applied in industry. This observation made us suspicious regarding whether the application of HLA could really be the way to achieve our research objective. Therefore, prior to formulating any requirements we decided to test whether our observation holds and if so, whether this observation could bring us to the conclusion that HLA is not the appropriate architecture which we are looking for.

In order to test whether our observation is correct we have formulated the observation as a hypothesis and we confronted it to a number of experts from the field of distributed simulation through a questionnaire and an interview survey. Besides asking the experts whether they agree with the hypotheses, we requested them to motivate their answers concerning their opinion, and we asked them to indicate alternative solutions to our research questions. In order to get an accurate and unbiased answer, the experts were chosen from various domains related to HLA and simulation: experts who designed and developed the HLA, experts from defence who apply HLA for daily use, experts from the industrial domain having knowledge of distributed simulation and/or HLA, researchers involved in distributed simulation and/or HLA, and finally, COTS simulation package vendors. The data collected from the experts is presented in Chapter 3.

We analyzed the collected data in Chapter 4. During the analysis firstly we identified the advantages and drawbacks of applying distributed simulation in industry, secondly, we

discussed the appropriateness of existing approaches for industry, and finally we proposed a design approach for distributed simulation architecture for industry. The proposed approach was expressed in the form of a list of requirements, containing 25 items, that needs to be satisfied when one intends to design and develop distributed simulation for the industrial domain. The appropriateness of the requirements was validated by the experts who participated in the survey, as well as by experts who did not directly participate. The resulting set of requirements provided the answer to our first research question. Although the set of requirements suggested in this thesis is just an initiative, we believe that it provides a beneficial starting point for stimulating a technical push of distributed simulation for industry.

As an answer to the second research question we presented in Chapter 5 a lightweight distributed simulation architecture, called the FAMAS Simulation Backbone Architecture. We designed and developed the backbone in order to couple port related distributed simulation models created especially in COTS simulation packages, which are frequently applied by industry. The backbone was designed in a way to support reuse of existing models, and to hide low level implementation details from the simulation practitioners making it easier for them to attach their models to it. Before answering the second research question completely, we first evaluated the implemented architecture within the research plan for which it was required. In addition to the fact that the backbone proved to be appropriate enough for the two case study projects presented in Chapter 6, for which it was intended, we could also answer affirmatively the third research question. In order to provide a complete answer to the second research question, however, the FAMAS Simulation Backbone Architecture had to be evaluated in a broader sense, that is with respect to the requirements identified for the industry in general.

In Chapter 7, therefore, we have evaluated the appropriateness of the proposed architecture for coupling simulation models for industry, that is, the FAMAS Simulation Backbone Architecture, with respect to the list of the identified requirements, proposed in Chapter 4. We have found that almost all the requirements are fulfilled by the backbone. In this sense we succeeded to answer our second research question as well.

Having provided answers to all our research questions we can state that we have achieved our research objective. We have provided a lightweight architecture for coupling simulation models, which is aimed at the industrial domain in the first place, and which can be used easier than HLA by the industrial community that applies COTS simulation packages. We observed that distributed computing in general is too complex for simulation practitioners who are not used to it, and therefore, it requires a lot of effort to learn to deal with it. Time spent to learn to deal with this complexity entails unforeseen costs that might be too high related to the expected benefits. The approach we suggest for reducing the costs of learning is to reduce complexity by hiding low level details of distributed computing from simulation practitioners. In this research we showed that a lightweight architecture for coupling simulation models is possible, with which, as we observed in two case studies, the participants and the problem owners were pleased. We observed that experts from industry believe that there is room for distributed simulation and for standards that implement the concepts for coupling industrial simulation models. We expect that providing lightweight architectures, that satisfy the identified requirements to a high degree, would create a market for distributed simulation in industry.

Appendix A Questionnaire

(one per simulation package)

Aim of the questionnaire:

To get an overview regarding the (willingness for the) application of High Level Architecture (HLA) for interoperability purposes within commercial-off-the-shelf (COTS) simulation packages.

Researcher: Csaba Attila Boer
Faculty of Economics, Department of Computer Science
Erasmus University, Rotterdam, The Netherlands
Telephone: +31-10-4081316,
Fax: +31-10-4089167
e-mail: acboer@few.eur.nl

Overview

Thank you very much for completing this questionnaire. The questionnaire aims to help me to find out whether and how existing distributed approaches, especially HLA standard, are applied and supported by COTS simulation package vendors. The outcome of the survey will be published in my Ph.D. thesis. I will be pleased to share the results of the survey with you and your organization; therefore I will send you a report after the evaluation.

My research deals with identifying and designing effective approaches for integrating distributed simulation models designed and developed in COTS simulation packages. For this reason I consider several existing approaches, like HLA, but besides I try to design and develop distributed approaches that can more effectively support interoperability.

The first part of the questionnaire requires information about your company and your function, while the rest concerns the view of your company regarding interoperability and the HLA standard. Please complete an apart questionnaire for each package that your organization provides.

Section 1: Simulation Vendor Information
Name: _____
Function: _____
Name of company: _____
Name of simulation package: _____
E-mail address: _____

Section 2: Support for Interoperability within the Simulation Package

Have, to your knowledge, any projects been carried out successfully that link two or more separate simulation models created in your package?

☐ **Yes** ☐ **No**

If Yes, could you please name the most important projects and the involved organizations?

_____

_____

_____

Section 3: Support for Interoperability with other Simulation Packages

Have, to your knowledge, any projects been carried out successfully that link two or more separate simulation models created in your package with models created in *other* simulation packages?

☐ **Yes** ☐ **No**

If Yes, could you please name the most important projects and the involved organizations?

_____

_____

_____

Section 4: Support for Interoperability with External Applications

Does your simulation package support interoperability with external applications (e.g., data bases, spreadsheets, optimization software, etc.)?

☐ **Yes** ☐ **No**

If Yes could you please specify the supported applications:

_____

_____

_____

_____

Section 5: Protocols/Middleware used for Interoperability

If your package supports interoperability, which protocol(s)/middleware do you use to realize it?

☐ WinSock ☐ CORBA ☐ COM

☐ ALSP interface ☐ DIS interface ☐ HLA interface

Other (please state):

_____

_____

Section 6: Support for HLA

Have, to your knowledge, any projects been carried out successfully in which simulation models created in your package are integrated using the HLA standard?

☐ **Yes** ☐ **No**

If No, please move to section 8.

Section 7: Support for HLA (2)

Could you please name the most important projects and the involved organizations that integrate models designed in your simulation package using HLA?

_____

_____

How much effort did the developers, to the best of your knowledge, take to make the simulation model HLA compliant?

_____ % - as a percentage of the overall time of the simulation project

☐ I do not know, I was not involved

If making the simulation model HLA compliant took significant effort, could you please indicate what were the most relevant difficulties encountered?

_____

_____

Have your organization ever reused HLA compliant simulation models created in your package in another project?

☐ **Yes** ☐ **No**

Section 8: Future Plans for Interoperability and HLA standard

Does your company make efforts to support HLA as an additional feature in your package?

☐ **Yes** ☐ **No** ☐ **It supports already**

Does your company make efforts to support other standards than HLA for distributed simulation?

☐ **Yes** ☐ **No**

If Yes, could you please elaborate on the other standards?

_____

_____

If you fill in the questionnaire in a printed form, please fax it to: +31-10-4089167

Thank you for taking the time to complete this questionnaire.

Appendix B COTS packages[48]

	COTS Simulation Package	Vendor	Typical Applications of the software	Primary Markets for which the software is applied	Contacted	Willing to participate
1	Analytica	Lumina Decision Systems, Inc.	Business modelling, decision and risk analysis, widely used as a more flexible alternative to spreadsheets	Industry, government, and education: telecommunications, healthcare, pharmaceuticals, energy, environment, consumer products, manufacturing, automotive, transportation, defence	YES	NO
2	AnyLogic 5.0	XJ Technologies	Business and system dynamics, performance, cost, and risk analysis, optimization, planning, decision support, agent-based, discrete/continuous	Strategic management, manufacturing, service, logistics, supply chain, material handling, healthcare, transportation, IT management, telecom, scientific	YES	YES
3	Arena	Rockwell Software	Manufacturing, supply chain, customer management, business process, healthcare, military, warehousing and logistics improvement	Manufacturing, supply chain/logistics, business process, military, healthcare	YES	YES
4	AutoMod	Brooks Automation	Material handling and movement systems, warehousing, baggage handling and manufacturing	Automotive, aerospace, airport operations, manufacturing, warehousing and distribution	YES	YES
5	Crystal Ball	Decisioneering, Inc.	Business planning and analysis, cost/benefit analysis, risk management, petroleum exploration, portfolio optimization, project management	Financial services, financial planning, oil and gas, pharmaceuticals, telecom, manufacturing, energy, utilities, insurance, government, aerospace	YES	YES
6	DecisionPro	Vanguard Software Corporation	Business financial modelling, process optimization, decision-making	Financial services, Management consulting, oil and gas, manufacturing, legal	YES	NO Reaction

[48] This table is deduced from the survey collected by James Swain except for the last two columns.

Appendix B

7	eM-Plant	Tecnomatix Technologies Inc.	Manufacturing, material handling, business process simulation, logistics, distribution, scheduling, line balancing, process verification, supply chain	Discrete manufacturing (automotive, electronics, shipyard, machining, line builder, etc), logistic, distribution, consulting, healthcare, banking	YES	YES
8	Enterprise Dynamics 5.0	Incontrol Enterprise Dynamics	Simulation, emulation, capacity analysis, staffing, material handling, facilities layout, line balancing, resource utilization, supply chain	Dedicated simulation suites for manufacturing, material handling, warehousing, airports, research and education, steel industry, ports	YES	YES
9	ExpertFit	Averill M. Law & Associates	Automatically and accurately fits probability distributions to data; helps choose distributions with absence of data	Manufacturing, defence, communications, transportation, healthcare, process reengineering, call centres, services, data analysis in general	YES	NO Reaction
10	Extend	Imagine That, Inc.	Model continuous, discrete-event, or discrete rate processes, plus get a relational data management system	Large scale and rate-based systems, manufacturing, logistics, packaging lines, transportation, business, call centres, engineering, scientific	YES	YES
11	Factory Explorer	Wright Williams & Kelly	Gross margin optimization, cycle time reduction, ramp up and ramp down planning, bottleneck analysis	Semiconductors, flat panel displays, thin film record heads, solar panels, automotive, discrete manufacturing and assembly	NO	N/A
12	FirstSTEP Designer	Interfacing Technologies	FirstSTEP Designer is a comprehensive business process management tool that gives decision makers the power	Enterprises focused on establishing operational standards or obligation to implement highly regulated processes & environments	YES	NO Reaction
13	Flexsim	Flexsim Software Products, Inc	Manufacturing, material handling, warehousing, distribution, real-time monitoring/analysis , supply chain, container shipping, storage access systems	Manufacturing, material handling, warehousing, distribution, real-time monitoring/analysis , supply chain, container shipping, storage access systems	YES	YES
14	GAUSS	Aptech Systems, Inc.	GAUSS is a general purpose scientific and statistical programming language with multiple tools	Economics, finance, social sciences, engineering, or any field that does data analysis or model fitting	NO	N/A

15	GoldSim	GoldSim Technology Group	Strategic planning, risk analysis and management, business dynamics, engineered systems modelling, portfolio management, environmental modelling	Manufacturing, mining, water resources, insurance, power, government (hazardous and radioactive waste management)	YES	YES
16	GPSS World for Windows	Minuteman Software	Modelling of manufacturing, telecommunications, computer networks, queuing networks, modelling of discrete-event systems	Operations research departments, industrial engineering professionals doing simulation modelling	YES	YES
17	HighMAST™	Highpoint Software Systems, LLC	.Net object-oriented framework for development of simulation services & components for master, slave or peer-level integration with larger systems.	Various - currently in use in batch manufacturing and schedule/plan validation, under consideration for others.	YES	YES
19	Micro Saint	Micro Analysis & Design, Inc.	Has been used primarily to model applications in defence, human factors, manufacturing, and service industries	Defence industry, human factors	YES	YES
20	mystrategy	Global Strategy Dynamics Ltd.	Business strategy and planning, business architecture modelling. Uses Strategy Dynamics approach (see Web)	Any business system; audience is strategy planners and general management wishing to understand business performance	NO	N/A
21	NAG C Library	Numerical Algorithms Group	Forecasting, logistics, portfolio/product, risk management, scheduling, linear/integer programming, optimization, non-linear programming, data mining	Chemical/process industries, finance, military, manufacturing, healthcare, oil/natural gas, environment, biotech, telecommunications, transportation	YES	NO Reaction
22	PASION Simulation System	Stanislaw Raczynski	Supports discrete-events, queuing, continuous ODE models, bond graphs, signal flow graphs, rigid body dynamics	Education, universities, simulation consultants and companies	YES	NO
23	PIMSS	MJC2 Limited	Modelling, optimisation and planning of manufacturing operations	Food, beverages, petroleum, cement, timber, bulk chemicals, pharmaceuticals, automotive, textiles, footwear, furniture, construction, plastics, telecommunications	YES	NO Reaction

Appendix B

24	ProcessModel	ProcessModel, Inc.	Business Process Analysis (BPA) process improvement, Six Sigma, ISO certification, requirements definition and application development	Can analyze and improve discrete-event processes in all markets	NO	N/A
25	ProModel	ProModel Solutions	Lean; Six Sigma; project & portfolio planning; capacity, cost analysis; process, cycle time improvement; supply chain	Manufacturing & logistics, pharmaceutical	YES	YES
26	Proplanner Manufacturing Process Management Software	Proplanner	Process planning and engineering solution to create, document, manage and access your manufacturing process information	Manufactures discrete product that involves product complexity greater than 25 components and variations of product	YES	NO Reaction
28	Resource Manager	User Solutions, Inc.	Manufacturing planning and scheduling models with inventory 'what-ifs' and alternate routing analysis	Manufacturing and operations management with people, machines, and material considerations\ conststraints	YES	NO Reaction
29	SAIL	CMS Research Inc.	Daily scheduling and performance monitoring of FMS	Metalworking, machining, automated CNC, machine cells, machine parts supplier	YES	NO Reaction
30	SansGUI Modelling and Simulation Environment	ProtoDesign Inc.	Scientific and engineering model building, simulation program development and deployment, dynamic charting and visualization	General systems, scientific research, engineering, and educational software development	YES	NO Reaction
31	ShowFlow	Webb Systems Limited	New facility planning; throughput time reduction; bottleneck analysis; staffing level analysis	Manufacturing, logistics, financial services, retail operations	YES	YES
32	SIGMA	Custom Simulations	General discrete-event systems, supporting all world views with an event relationship graphical interface	General manufacturing and service systems including bioproduction, semiconductor, health care, and banking enterprises	YES	NO Reaction
33	SILK	Thread Tec.			YES	COULD NOT CONTACT
34	SimCAD Pro	CreateASoft, Inc.	Manufacturing, assembly line, robotics, lab automation, factory layout, workflow re-engineering, conveyor simulation, mail order and fulfilment	Manufacturing, assembly line, robotics, lab automation, factory layout, workflow re-engineering, conveyor simulation, mail order and fulfilment	YES	NO

35	SIMPROCESS	CACI Products Company	SIMPROCESS is a key component in the business process and reengineering improvement life cycle	SIMPROCESS is designed for organizations that need to analyze varied scenarios and to mitigate risk	YES	NO Reaction
36	SIMUL8	SIMUL8 Corporation	Work flow management, throughput analysis, de-bottlenecking, new product/process development, capacity analysis, continuous improvement, what-if scenarios	Business process, call centres, manufacturing, supply chain, logistics, healthcare, financial, pharmaceutical, customer service, Six Sigma	YES	YES
37	SLIM	MJC2 Limited	Strategic modelling, optimisation and analysis of distribution networks, transport operations and supply chain networks	Logistics, distribution, petroleum, retail, 3PL, express delivery, transport, bulk chemicals, construction, container shipping, food, beverages	YES	NO Reaction
38	SLX	Wolverine Software			YES	YES (Added to Swain List)
39	Supply Chain Builder	Simulation Dynamics	SCB is used to gain understanding of complex supply and/or distribution chains and evaluate	Automotive, consumer goods, food and beverage, transportation, pharmaceutical	YES	NO Reaction
40	VisSim	Visual Solutions	Nonlinear dynamic systems, control design, embedded systems, logistics, economics, motion control, real-time simulation of virtual plants	Automotive, aerospace, HVAC, power, process, industrial control, logistic scheduling	YES	YES
41	Visual Simulation Environment	Orca Computer, Inc.	The Visual Simulation Environment software product is an integrated development and execution environment for discrete-event, general-purpose	VSE is a general-purpose simulation product that can be used for discrete-event modelling and simulation	YES	COULD NOT CONTACT
42	WebGPSS (micro-GPSS)	FLUX Software Engineering	General purpose discrete-events simulation	Educational (business and engineering)	YES	YES
43	Witness	Lanner Group			YES	YES (Added to Swain List)

Appendix C Interview survey for experts

PART 1. INTRODUCTION

1.1 Personal information

Name: _____ *Function:* _____ *Name of organization:* _____ *E-mail address:* _____ *Date of interview:* _____ Start and end time of the interview: _____	

1.2 Confidentiality

Do you agree to record this interview?	☐ YES	☐ NO

PART 2. EXPERIENCE WITH THE HLA STANDARD, COTS SIMULATION PACKAGES AND GENERAL PROGRAMMING LANGUAGES

2.1 Experience with the HLA standard

Could you please tell me about your knowledge and experience with HLA standard? _____ _____ _____ _____ _____ _____	• Are you familiar with HLA? • How long have you been working with it? • Did you ever use the RTI? • Did you ever use the RTI in projects? • Are you involved in research? • Have you ever been involved in accepting HLA as a standard? • Have you been involved in designing and developing the HLA standard?

2.2 Experience with COTS Simulation Packages

Could you please tell me about your knowledge and experience with Commercial-off-the-Shelf (COTS) simulation packages?
_____
_____
_____
_____
_____
_____
_____
	• Which ones are you familiar with?
	• How long have you been using them?
	• Are you still using some? Which ones?
	• Have you used them in projects?
	  - In how many projects?
	  - Were they big projects?
	  - What was your role?
	• Have you ever been involved in designing and developing any of the packages?
	• Are you researching them?

2.3 Experience with general programming languages

Could you please sketch your knowledge and experience with general programming languages (e.g., C++, Java, COBOL, Fortran, etc.)? _____ _____ _____ _____ _____	• Are you familiar with any programming languages? • Which ones? • How long have you been using them? • Are you still using some? • Have you ever taking part in projects where you had to program yourself? - In how many projects? - Were they big projects?

PART 3. EXPERIENCE WITH DISTRIBUTED SIMULATION PROJECTS

3.1 General issues

Have you been involved in distributed simulation projects? _____	☐ YES ☐ NO
In how many projects? _____ _____	☐ 0 ☐ <5 ☐ <5,10> ☐ >10
How did you couple the simulation models? What kind of architecture did you use? _____ _____ _____	☐ DIS ☐ HLA ☐ CORBA ☐ COM ☐ WinSock ☐ Other
What was your role in these projects? _____ _____ _____	☐ Proj. Manager ☐ Developer ☐ Tester ☐ Referee ☐ Other
Did you use COTS simulation packages in these projects? *Which ones?* _____ _____ _____ _____ _____	☐ YES ☐ NO If YES, ▪ In how many projects? ▪ Which COTS simulation packages?
Were these projects industrial oriented, military oriented or both? _____ _____	☐ Military ☐ Industrial

3.2 Differences between the industrial and defence oriented simulation communities.

In my opinion there is a difference between the industrial and defence oriented simulation communities. Defence simulation communities mainly use general purpose languages (e.g., Java, C++, Fortran, etc.) for creating simulation models, while the industrial simulation communities use COTS simulation packages. *Do you agree?* _____ _____ _____ _____ _____ _____ _____ _____	☐ YES ☐ NO If YES: *Why do you think it is so?* If NO *Why not?*

3.3 Other differences between the industrial and defence oriented simulation communities.

Have you observed any other differences between the defence and industrial oriented simulation communities? _____ _____ _____	

PART 4. EXPERIENCE WITH HLA PROJECTS

4.1 Experience applying the HLA standard

Which HLA RTI versions have you been using?	☐ DMSO ☐ IEEE-1516 ☐ pitch-RTI ☐ MAK-RTI
Besides interoperability, reusability is also an important feature of the HLA standard. *Have you reused or tried to reuse any parts of the model in another project?*	☐ YES ☐ NO If YES - Did you have any problems? - Did you have to change the model? - How much effort was needed to reuse? - Would have been easier to completely reimplement the reused model? If NO - Why not? - Was not an issue?
Besides reusability another issue is the information hiding, especially when different organizations are involved in a project. *Was information hiding within the model in your project an issue?*	☐ YES ☐ NO If YES - Have you managed to hide all the information? - What kind of approach/technique did you use?
Have you been involved in projects where simulation models developed in COTS simulation packages were coupled through HLA-RTI?	☐ YES ☐ NO

4.2 Experience with integrating simulation models developed in COTS simulation packages through HLA-RTI –advantages and difficulties of HLA standard

In my research I would like to investigate the benefits and difficulties of the applicability of HLA standard for integrating simulation models created in COTS simulation packages. • *What was the benefit of using HLA?* • *What kind of problems did HLA solve for you?* • *Could you please tell me about one of your successful projects that you had in which you applied HLA?* • *What was difficult in using HLA?* • *Did you have any problem?* • *Could you please talk about one of your nightmare projects you had in which you applied HLA?*	

4.3 Support COTS simulation packages by HLA vendors (ONLY FOR COMMERCIAL HLA-RTI VENDORS)

Does your company have any clients from the industrial domain that use your HLA-RTI product for coupling industrial oriented simulation models?	☐ YES ☐ NO If YES • Could you please name the involved organizations? • Do these organizations use COTS simulation packages or general purpose programming languages for designing and developing simulation models? • If they use COTS simulation packages: - What kind of COTS simulation packages? - Is your company providing any HLA interface for these COTS simulation packages or is the simulation modeller responsible to develop them?

PART 5. EXPERT'S FUTURE VISION REGARDING INTEROPERABILITY AND THE HLA STANDARD

5.1 Future plans for interoperability and HLA standard

Requirements for "an ideal" distributed simulation architecture for industry

Which features DO YOU LIKE and DO NOT LIKE in the HLA standard and other architectures for distributed simulation? _____ _____ _____ _____ _____	
My observation is that: ***HLA is hardly applied in industry for integrating simulation models designed and developed in COTS simulation packages.*** *Do you agree?* _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____	☐ YES ☐ NO If YES • Can you please state some reasons? • *How would you imagine "an idealized" distributed simulation architecture for industry?* • What functionalities would the architecture offer? • What is the minimum set of functionalities that the architecture should offer to achieve distributed simulation? • How should the interfaces look like?
Are there any additional comments you want to make regarding the future of the interoperability? _____ _____	

Appendix C

PART 6. CLOSING SECTION

6.1 Further references

Could you please provide me some groups, names of distributed simulation experts, who have been involved in the integration of COTS simulation models through HLA?	

6.2 Additional Information

If I will have additional questions, may I contact you by e-mail?	

6.3 Evaluation Interview

Could you give me some feedbacks on this interview? Did I forget something?	

Appendix D Participants in the FAMAS.MV2 Simulation Backbone Project

- **ECT (European Container Terminals)**[49]
  - Ruud van der Ham
  - Anko Nagel
- **Port Authority of Rotterdam**[50]
  - Maurits van Schuylenburg
- **TBA Nederland**[51]
  - William Rengelink
  - Yvo Saanen
  - Shing Wong
- **TRAIL Research School**[52]
  - Csaba Attila Boer (Erasmus University Rotterdam)
  - Mark Duinkerken (Delft University of Technology)
  - Jaap Ottjes (Delft University of Technology)
  - Hans Veeke (Delft University of Technology)
  - Alexander Verbraeck (Delft University of Technology)
  - Corné Versteegt (Delft University of Technology)

[49] http://www.ect.nl/

[50] http://www.portofrotterdam.com/

[51] http://www.tbanederland.nl/

[52] http://www.trail.tudelft.nl/

Appendix E Standard Messages in the Backbone

- **AskJoinFederation*(string IP-address)***

  Sender: Any component
  Receiver: Run Control
  Reply Message: AskJoinFederation*(int PortNumber)*
  Description: If a component wants to join a distributed simulation execution (federation), it first needs to send an AskJoinFederation message to the Run Control component. The parameter of this message contains the location (IP address) of the component concerned. The Run Control component replies with the same message, in which parameter list contains the port number which will be used to set up a listening socket. If the port number is not acceptable for the component, it will try to join the federation again.

- **ConfirmJoinFederation**

  Sender: Any component
  Receiver: Run Control
  Reply Message: N/A
  Description: After obtaining a port number from the Run Control component (see AskJoinFederation), the component concerned sets up a thread for a server socket to listen to the other participants. As soon as the component is listening it sends a ConfirmJoinFederation to the Run Control. On receipt of this message, the Run Control registers the component concerned, so that if other components intend to contact this component the Run Control component can provide information concerning the address and the port number at which the component is listening.

- **TellAddressSubsystem*(string SubsystemName)***

  Sender: Any component
  Receiver: Run Control
  Reply Message: TellAddressSubsystem*(string SubsystemName, string IP-address, int PortNumber)*
  Description: Any component that joined the federation can obtain information concerning the other components. The only information that needs to be known is the name of the component. A component can request information regarding the location of a specific component by sending a TellAddressSubsystem to the Run Control component, specifying the name of the requested component. If the Run Control component succeeds in finding the requested component, it sends back the same message with the parameter list containing information about the component demanded. An empty list of parameters means that the component is not available.

- **TellVariable *(string VariableName)***

  Sender: Any component
  Receiver: Run Control
  Reply Message: TellVariable*(string VariableName, string VariableValue, boolean OK)*
  Description: Any component that joined the federation can obtain variables defined in the Scenario Object. Variables requested can be either belong to the concerned component or a publicly available variable of another component. To the request the Run Control replies with the same message. The parameter list of this message contains the value of the variable, and additionally, a logical variable which identifies whether the component has rights to access the demanded variable.

- **TellAllVariables**

  Sender: Any component
  Receiver: Run Control
  Reply Message: TellAllVariables*(string VariableName, string VariableValue,...)*
  Description: This message is similar to the TellVariable message, with the difference that it returns all the variable values which are publicly available for the requesting component.

- **SubscribeVariableUpdate *(string VariableName)***

  Sender: Any component
  Receiver: Run Control
  Reply Message: SubscribeVariableUpdate*(boolean OK)*
  Description: Although there is a possibility to exchange messages between two components using the peer to peer communication mechanism, we can apply the well know publish/subscribe mechanism as well through the Run Control component. A component that joined the simulation execution can send a SubscribeVariableUpdate message to the Run Control component in order to get noticed whenever the value of this variable changes. If any change occurs in the value of the variable, the Run Control component immediately sends a NotifyVariableUpdate message including the updated value. The Run Control returns the same message with the logical parameter OK set to true to let the requester know whether the subscription was successful, that is whether the requested variable is public or belongs only to the component.

- **SubcribeAllVariableUpdate**

  Sender: Any component
  Receiver: Run Control
  Reply Message: N/A
  Description: This message is similar to the SubscribeVariableUpdate, with the difference that the sender component intends to subscribe for all the publicly available variables defined in the Scenario Object.

- **NotifyVariableUpdate *(string VariableName, VariableValue)***

  Sender: Run Control
  Receiver: components subscribed to the corresponding variable
  Reply Message: N/A
  Description: Whenever the value of a variable changes, the Run Control component sends this message to notify the components that subscribed to this variable about the change, specifying the new value.

- **TellAllParticipants**

  Sender: Any component
  Receiver: Run Control
  Reply Message: TellAllParticipants*(list of participating components)*
  Description: Components can request through this message the list of all components containing their names and addresses. The Run Control component replies with the same message as a parameter the list of all participating components that already joined the federation including their names and addresses.

- **BBTMCanStartSimulation**

  Sender: Run Control
  Receiver: BBTM
  Reply Message: N/A
  Description: Run Control informs the BBTM that all components are ready so that the simulation can be started.

- **NextEvent *(int EventTime)***

  Sender: Any component
  Receiver: BBTM
  Reply Message: N/A
  Description: This message aims to solve time synchronization. The sender provides the BBTM with the time stamp of the next message from its event calendar. The BBTM uses conservative time synchronization. According to the algorithm of this synchronization mechanism the BBTM will choose the smallest time stamp and it notifies the concerned component by sending a NotifyNextEvent message.

- **NotifyNextEvent**

  Sender: BBTM
  Receiver: Any component
  Reply Message: N/A
  Description: The BBTM lets a specific component know through this message that it can perform its next event.

- **StopSimulation**

  Sender: Any component
  Receiver: Run Control
  Reply Message: N/A
  Description: The component informs the Run Control component about its intention to stop the simulation execution.

- **StopSimulationNow**

  Sender: Run Control, BBTM t
  Receiver: Any component
  Reply Message: N/A
  Description: The Run Control or the BBTM informs the other components that they *must* stop their simulation.

- **SimulationStopped**

  Sender: Any component
  Receiver: Run Control or BBTM
  Reply Message: N/A
  Description: As a confirmation to the StopSimulationNow message, the participating components inform the Run Control or BBTM that they have stopped the simulation and they have closed down.

- **Pause**

  Sender: Any component
  Receiver: BBTM
  Reply Message: N/A
  Description: Suspends the simulation execution until the sender component sends a Resume message.

- **Resume**

  Sender: Any component
  Receiver: BBTM
  Reply Message: N/A
  Description: Receiving a resume message, the BBTM will continue the paused simulation by sending the next notification.

- **AddToBackGround *(objecttype, real coordinates, color, boolean Fill)***

  Sender: Any component
  Receiver: Visualization Component
  Reply Message: N/A
  Description: Adds an object to the background with the specific information defined by the parameter list .

- **NewFigure *(string ShapeName, real ScalingFactors, Color, RenderMode)***

  Sender: Any component
  Receiver: Visualization Component
  Reply Message: NewFigure*(string FigureID)*
  Description: Defines new figures dynamically during the simulation run.

- **ShowFigure** ***(string FigureID, real Position(X,Y,Z), real OrientationChange(αX,αY,αZ))***

  Sender: Any component
  Receiver: Visualization Component
  Reply Message: N/A
  Description: Shows a figure based on the information given in the parameter list.

- **CreateTable** ***(string TableName, string FieldName1, string FieldType1, string FieldName2, string FieldType2,…)***

  Sender: Any component
  Receiver: Logging Component
  Reply Message: N/A
  Description: Creates a table called TableName with the specified field names and field types given as parameters.

- **AddRecord** ***(string TableName, string FieldName1, FieldValue1, string FieldName2, FieldValue2, …)***

  Sender: Any component
  Receiver: Logging Component
  Reply Message: N/A
  Description: Sends information to the Logging component. This data will be inserted into the table called TableName.

- **CloseTable (string TableName)**

  Sender: Any component
  Receiver: Logging Component
  Reply Message: N/A
  Description: Closes the specified table.

Appendix F Participants in the FAMAS.MV2 Pre-design Road Container Handling Project

- M. van Schuylenburg (Port Authority of Rotterdam)
- I. Miller (Port Authority of Rotterdam – Initi8)
- M. Melis (Port Authority of Rotterdam – Initi8)
- B. van de Rakt (Port Authority of Rotterdam – Initi8)
- G. Driebeek (3BK Consultancy Group)
- M. van Nederpelt (LogicaCMG)
- L. Verspui (LogicaCMG)
- R. Schoo (Districon)
- R. Karreman (Belastingdienst/douane)
- J. Molenaar (ECT Home)
- R. Geurtsen (Euromax Terminal)
- F. van den Boom (Groenenboom Transport)
- B. van Eck (Illyan)
- J. Seager (Illyan)
- R. Nieuwveld (Mitsui O.S.K. Lines Europe)
- K. van Til (TBA Nederland)
- A. de Waal (TBA Nederland)
- D. Henstra (TNO Inro)
- A. van der Ham (TNO Inro)
- J. Vleugel (TRAIL Research School)
- A. Verbraeck (TRAIL Research School)
- C.A. Boer (TRAIL Research School)
- P. Dijkshoorn (Transport & Logistiek Nederland)

BIBLIOGRAPHY

Ackoff, R. L. (1974), Redesign the Future. A System Approach to Societal Problems, New York, USA: John Wiley and Sons.

Ackoff, R. L., and Sasieni, M. W. (1968), Fundamentals of Operational Research, New York, USA: John Wiley and Sons.

Albergo, E. J., and Thomen, D. (2000), "Simulation Based Acquisition, The Way Ahead" in Simulation Interoperability Workshop, Spring, SISO, 00S-SIW-120.

Arnold, K., Gosling, J., and Holmes, D. (2000), The Java(TM) Programming Language (3 ed.), Addison-Wesley.

Auinger, F., Vorderwinkler, M., and Buchtela, G. (1999), "Interface Driven Domain-Independent Modeling Architecture for "Soft-Commissioning" and "Reality in the Loop"" in Winter Simulation Conference, eds. P. A. Farrington, H. B. Nembhard, D. T. Sturrock and G. W. Evans, Phoenix, USA: Association for Computing Machinery Press, pp. 798-805.

Babeliowsky, M. (1997), "Designing Interorganizational Logistic Networks", Ph.D. Thesis, Delft University, Delft, The Netherlands.

Balci, O. (1997), "Principles of Simulation Model Validation, Verification, and Testing", Transactions of the Society for Computer Simulation International, 14(1), pp. 3-12.

Balci, O. (1998), "Verification, Validation and Testing" in Handbook of Simulation. Principles, Methodology, Advances, Application and, Practice, ed. J. Banks, New York, USA: John Wiley and Sons, pp. 335-393.

Balci, O. (2003), "Verification, Validation, and Certification of Modeling and Simulation Applications" in Winter Simulation Conference, eds. S. Chick, P. J. Sanchez, D. Ferrin and D. J. Morrice, New Orleans, Louisiana, USA: Association for Computing Machinery Press, pp. 150-158.

Banks, J., Hugan, J. C., Lendermann, P., McLean, C., Page, E. H., Pegden, C. D., O. Ulgen, and Wilson, J. R. (2003), "The Future of the Simulation Industry" in Winter Simulation Conference, eds. D. Ferrin, S. Chick, P. J. Sanchez and D. J. Morrice, New Orleans, Louisiana, USA: Association for Computing Machinery Press, pp. 2033-2043.

Best, S. J., and Krueger, B. S. (2004), Internet Data Collection., ed. M. S. Lewis-Beck, Thousand Oaks, California, USA: Sage Publications, Inc.

Beveridge, J., and Wiener, R. (1997), Multithreading Applications in Win32: The Complete Guide to Threads, Boston, USA: Addison-Wesley Developers Press.

Bluemel, E., and Novitsky, L. (2000), Simulation and Information System Design: Application in Latvian Ports, The Society for Computer Simulation International.

Boer, C. A., Saanen, Y. A., Veeke, H. P. M., and Verbraeck, A. (2002b), "Final Report Simulation Backbone FAMAS.MV2 Project 0.2 Technical Design", Technical Report, Research Report Connekt, Delft, The Netherlands.

Boer, C. A., and Verbraeck, A. (2002), "Connecting High Level Distributed Simulation Architectures: An Approach for a FAMAS-HLA Bridge" in European Simulation Symposium, eds. A. Verbraeck and W. Krug, Dresden, Germany: The Society for Computer Simulation International, pp. 398-405.

Boer, C. A., and Verbraeck, A. (2003), "Distributed Simulation With COTS Simulation Packages" in Winter Simulation Conference, eds. S. Chick, P. J. Sanchez, D. Ferrin and D. J. Morrice, New Orleans, Louisiana, USA: Association for Computing Machinery Press, pp. 829-837.

Boer, C. A., Verbraeck, A., De Waal, A., Van Eck, B., and Seager, J. (2003), "Distributed e-Services for Road Container Transport Simulation" in 15th European Simulation Symposium, eds. A. Verbraeck and V. Hlupic, Delft, The Netherlands: The Society for Computer Simulation International, pp. 541-550.

Boer, C. A., Verbraeck, A., Saanen, Y., and Veeke, H. P. M. (2002c), "A Virtual Design Environment for the Port of the Future. The FAMAS Simulation Backbone Architecture" in 7th TRAIL Congress, ed. P. H. L. Bovy, Rotterdam, The Netherlands: The Netherlands Research School for Transport, Infrastructure and Logistics, pp. 1-13.

Boer, C. A., Verbraeck, A., and Veeke, H. P. M. (2002a), "Distributed Simulation of Complex Systems: Application in Container Handling" in European Simulation Interoperability Workshop, Harrow, Middlesex, UK: SISO, pp. 134-142.

Boer, C. A., Verbraeck, A., and Veeke, H. P. M. (2002d), "The Possible Role of a Backbone Architecture in Real-Time Control and Emulation" in Winter Simulation Conference, eds. E. Yücesan, C. H. Chen, J. L. Snowdon and J. M. Charnes, San Diego, California, USA: Association for Computing Machinery Press, pp. 1675-1682.

Boyson, S., Corsi, T., and Verbraeck, A. (2003), "The e-Supply Chain Portal: a Core Business Model", Transportation Research Part E, 39, pp. 175-192.

Brassé, M., Huiskamp, W., and Stroosma, O. (1999), "A Component Architecture for Simulator Development" in Simulation Interoperability Workshop, Fall, Orlando, Florida, USA: SISO, pp. 168-174.

Bunge, M. A. (1979), Treatise on Basic Philosophy. A World of Systems (Vol. 4), Dordrecht, The Netherlands: Reidel Publishing Company.

Burks, T., Alexander, T., Lessmann, K., and LeSueur, K. G. (2002), "Latency Performance of Various HLA RTI Implementations", Technical Report, White paper, http://www.mak.com/latency.pdf.

Buss, A., and Jackson, L. (1998), "Distributed Simulation Modeling: A Comparision of HLA, CORBA and RMI" in Winter Simulation Conference, eds. D. J. Medeiros, E. F. Watson, J. S. Carson and M. S. Mantvannan, Washington DC, USA: Association for Computing Machinery Press, pp. 818-825.

Cai, W., Turner, S. J., and Gan, B. P. (2001), "Hierarchical Federations: an Architecture for Information Hiding" in 15th Workshop on Parallel and Distributed Simulation, Los Alamitos, California, USA: IEEE Computer Society Press, pp. 67-74.

Chandy, K. M., and Misra, J. (1979), "Distributed Simulation: A Case Study in Design and Verification of Distributed Programs", IEEE Transactions on Software Engineering, SE-5(5), pp. 440-452.

Churchmann, C. W. (1971), The Design of Inquiring Systems, New York, USA: Basic Books.

Comer, D. E. (2000), Internetworking with TCP/IP Vol.1: Principles, Protocols, and Architecture (Vol. 1, 4 ed.), Upper Saddle River, New Jersey, USA: Prentice Hall.

Comer, D. E., and Stevens, D. L. (1997), Internetworking with TCP/IP Vol. 3. Client-Server Programming and Applications-Windows Sockets Version (Vol. 3, 2 ed.), Upper Saddle River, New Jersey, USA: Prentice Hall.

Connekt (2001), International State-of-the-Art in Container Logistics and Performance Requirements for Mega Hubs. A Vision for Container Logistics in the Port of Rotterdam, Delft, The Netherlands: Connekt.

Connekt (2003), Truck Afhandeling. Eindrapport FAMAS.MV2 - Project 12.1, Delft, The Netherlands: Connekt.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001), Introduction to Algorithms (2 ed.), Cambridge, Massachusetts, USA: The MIT Press.

Creswell, J. W. (2003), Research Design. Qualitative, Quantitative, and Mixed Methods Approaches. (Second ed.), Thousand Oaks, California, USA: SAGE Publication.

Crooks, M., Igarza, J. L., Löf, S., Zimmerman, P., and Turrell, C. (2004), "HLA Compliance Certification: Lessons learned from the US past activity and current developments within NATO and PfP nations" in European Simulation Interoperability Workshop, Edinburgh, Scotland: SISO (04E-SIW-071).

Daft, R. L. (1998), Organization Theory and Design, South-Western College.

Dahmann, J. S., Fujimoto, R. M., and Weatherly, R. M. (1998), "The DoD High Level Architecture: An Update" in Winter Simulation Conference, eds. D. J. Medeiros, E. F. Watson, J. S. Carson and M. S. Mantvannan, Washington DC, USA.

DaimlerChrysler. (2003), "Chrysler and the digital factory", www.allpar.com/history/digital.html.

De Hartog, A. H., et al. (2001), International state-of-the-art in container logistics and performance requirements for mega hubs. A vision for container logistics in the port of Rotterdam, Delft, The Netherlands: Connekt.

De Swaan Arons, H., and Boer, C. A. (2001a), "Retrieving Parameterized Simulation Models from a Database" in Proceedings of the 4th International Eurosim 2001 Congress, eds. A. Heemink, L. Dekker, H. De Swaan Arons, I. Smit and T. Van Stijn, Delft, The Netherlands.

De Swaan Arons, H., and Boer, C. A. (2001b), "Storage and Retrieval of Discrete-Event Simulation Models", Journal of Simulation Practice and Theory (SIMPRA), 8(8), pp. 555-576.

De Swaan Arons, H., and Boer, C. A. (2002), "Ranking a List of Discrete-Event Models" in Proceedings of the 35th Annual Simulation Symposium, ed. A. Jacobs, San Diego, USA: IEEE Computer Society, pp. 151-159.

De Vreede, G. J. (1995), "Facilitating Organizational Change. The Participative Application of Dynamic Modelling", PhD Thesis, Delft University of Technology, Delft, The Netherlands.

DIS Steering Committee. (1994), "The DIS Vision, A Map to the Future of Distributed Simulation", Technical Report IST-SP-94-01, Institute for Simulation and Training, Orlando, Florida, USA.

DMSO. (1998a), "High Level Architecture Object Rules v1.3", Technical Report, Defense Modeling and Simulation Office, Washington DC, USA, http://www.hla.dmso.mil/.

DMSO. (1998b), "High Level Architecture Interface Specification v1.3", Technical Report, Defense Modeling and Simulation Office, Washington DC, USA, http://www.hla.dmso.mil/.

DMSO. (1998c), "High Level Architecture Object Model Template v1.3", Technical Report, Defense Modeling and Simulation Office, Washington DC, USA, http://www.hla.dmso.mil/.

DMSO. (1999), "High Level Architecture - Federation Development and Execution Process (FEDEP) Model", Technical Report, Defense Modeling and Simulation Office, Washington DC, USA, http://www.hla.dmso.mil/.

Endres, A., and Rombach, D. (2003), A Handbook of Software and System Engineering, Harlow, England: Addison Wesley.

Fishwick, P. A. (1995), Simulation Model Design and Execution: Building Digital Worlds, Englewood-Cliffs: Prentice-Hall.

Flinsenberg, I. C. M. (2004), "Route Planning Algorithms for Car Navigation", Ph.D. Thesis, Eindhoven University of Technology, Eindhoven, The Netherlands.

Fredenhall, L. D., and Hill, E. (2001), Basics of Supply Chain Management, New York, USA: St. Lucie Press.

Fujimoto, R. M. (1998a), "Parallel and Distributed Simulation" in Handbook of simulation. Principles, Methodology, Advances, Applications, and Practice, ed. J. Banks, New York, USA: John Wiley and Sons, Inc., pp. 429-464.

Fujimoto, R. M. (1998b), "Time Management in the High Level Architecture", SIMULATION, 71(6), pp. 388-400.

Fujimoto, R. M. (2000), Parallel and Distributed Simulation Systems, New York, USA: John Wiley and Sons, Inc.

Gan, B. P., Liu, L., Jain, S., Turner, S. J., Cai, W., and Hsu, W. (2000), "Distributed Supply Chain Simulation Across Enterprise Boundaries" in Winter Simulation Conference, eds. K. Kang, P. A. Fishwick, J. A. Joines and R. R. Barton, Orlando, Florida, USA: Association for Computing Machinery Press, pp. 1245-1251.

Heineman, G. T., and Councill, W. T. (2001), Component-Based Software Engineering - Putting the Pieces Together, Boston, USA: Addison-Wesley.

Hoffer, J. A., George, J. F., and Valacich, J. S. (2002), Modern System Analysis and Design, New Jersey, USA: Prentice-Hall International, Inc.

Hugos, M. H. (2003), Essentials of Supply Chain Management, New York, USA: John Wiley and Sons.

IEEE. (2000a), "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Framework and Rules (IEEE Std 1516-2000)", Technical Report, Institute of Electrical and Electronics Engineers, Inc., http://standards.ieee.org/catalog/olis/compsim.html.

IEEE. (2000b), "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Federate Interface Specification (IEEE Std 1516.1-2000)", Technical Report. Institute of Electrical and Electronics Engineers, Inc., http://standards.ieee.org/catalog/olis/compsim.html.

IEEE. (2000c), "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Object Model Template (OMT) Specification (IEEE Std 1516.2-2000) ", Technical Report, Institute of Electrical and Electronics Engineers, Inc., http://standards.ieee.org/catalog/olis/compsim.html.

Jansen, R., Huiskamp, W., Boomgaardt, J. J., and Brassé, M. (2004), "Real-time Scheduling of HLA Simulator Components" in Simulation Interoperability Workshop, Spring, SISO (04S-SIW-030).

Keen, P. G. W. (1991), Shaping the Future. Business Design through Information Technology, Harvard Business School Press.

Keen, P. G. W., and Scott-Morton, M. S. (1978), Decision Support Systems. An Organizational Perspective, London, UK: Addison-Wesley.

Kelton, W. D., Sadowski, R. P., and Sturrock, D. T. (2003), Simulation with Arena (3 ed.), Boston, USA: McGraw-Hill.

Kenis, D. (1995), "Improving Group Decisions. Designing and Testing Techniques for Group Decisions Support Systems Applying Delphi Principles." Ph.D. Thesis, University of Utrecht, Faculty of Social Sciences, Utrecht, The Nederlands.

Kuhl, F., Weatherly, R., and Dahmann, J. (1999), Creating Computer Simulation Systems: An Introduction to the High Level Architecture, New Jersey, USA: Prentice Hall.

Law, A. M., and Kelton, W. D. (2000), Simulation Modeling and Analysis, Boston, USA: McGraw-Hill.

Leenaarts, M., and Kentrop, M. (2003), "Distributed Planning of Container Terminal Resources with Agent Technology" in BNAIC, Amsterdam, The Netherlands.

Linstone, H. A. (1978), "The Delphi Technique" in Handbook of Future Research, ed. J. Fowles, London: Greenwood Press, pp. 273-300.

Linstone, H. A., and Turoff, M. (1975), The Delphi Method. Techniques and Applications., London, UK: Addison-Wesley.

Linthicum, D. S. (1999), Enterprise Application Integration, New York, USA: Addison-Wesley.

Linthicum, D. S. (2003), Next Generation Application Integration. From Simple Information to Web Services., Boston, USA: Addison-Wesley.

Lutz, R., Scrudder, R., and Graffagnini, J. (1998), "High Level Architecture Object Model Development And Supporting Tools", SIMULATION, 71(6), pp. 401-409.

McIlroy, M. D. (1969), "Mass Produced Software Components" in Software Engineering: Concepts and Techniques, ed. P. Naur, New York, USA: Mason/Charter Publishers Inc., pp. 138-150.

McLean, C., and Riddick, F. (2000), "The IMS MISSION Architecture for Distributed Manufacturing Simulation" in Winter Simulation Conference, eds. J. A. Joines, R. R. Barton, K. Kang and P. A. Fishwick, Orlando, Florida, USA: Association for Computing Machinery Press, pp. 1539 - 1548.

Miller, D. C., and Thorpe, J. A. (1995), "SIMNET: The Advent of Simulator Networking" in Proceedings of the IEEE, 83(8).

Miller, I., and Melis, M. A. C. (2001), "FAMAS.MV2 TA Project Proposal, Project 12.1:Pre-design Road Container Handling", Technical Report, Connekt, Rotterdam, The Netherlands.

Mitroff, I. I. (1974), The Subjective Side of Science, Amsterdam, The Netherlands: Elsevier.

Mullins, L. J. (2002), Management and Organizational Behaviour, Harlow, UK: Prentice Hall.

Nance, R. E. (1993), "A History of Discrete Event Simulation Programming Languages" in Proceedings of the Second ACM SIGPLAN History of Programming Languages Conference, pp. 149-175.

Nikoukaran, J., Hlupic, V., and Paul, R. J. (1999), "A Hierarchical Framework for Evaluating Simulation Software", Journal of Simulation Practice and Theory (SIMPRA), 7(3), pp. 219-232.

Olson, N., Willcocks, L., and Petherbridge, P. (2002), Making IT Count. Strategy, Delivery, Infrastructure, Oxford, UK: Butterworth Heinemann.

Oses, N., Pidd, M., and Brooks, R. J. (2003), "Critical Issues in the Development of Component-Based Discrete Simulation", Technical Report, LUMS Working Paper, The Department of Management Science, Lancaster University, UK.

Page, E. H., Canova, B. S., and Tufarolo, J. A. (1997), "A Case Study of Verification, Validation and Accreditation for Advanced Distributed Simulation", ACM Transactions on Modeling and Computer Simulation, 7(3), pp. 393-424.

Page, E. H., and Smith, R. (1998), "Introduction to Military Training Simulation: A Guide for Discrete Event Simulationists" in Winter Simulation Conference, eds. D. J. Medeiros, E. F. Watson, J. S. Carson and M. S. Mantvannan, Washington DC, USA: Association for Computing Machinery Press, pp. 53-60.

Paul, R. J., and Taylor, S. J. E. (2002), "What Use is Model Reuse: Is There a Crook at the End of the Rainbow?" in Winter Simulation Conference, eds. E. Yücesan, C. H. Chen, J. L. Snowdon and J. M. Charnes, San Diego, California, USA: Association for Computing Machinery Press, pp. 648-652.

Pidd, M. (2002), "Simulation Software and Model Reuse: A Polemic" in Winter Simulation Conference, eds. E. Yücesan, C. H. Chen, J. L. Snowdon and J. M. Charnes, San Diego, California, USA: Association for Computing Machinery Press, pp. 772-775.

Pidd, M. (2003), Tools for thinking. Modelling in Management Science, Chichester, UK: John Wiley and Sons.

Pope, A. (1989), "The SIMNET Network and Protocols", Technical Report 7102, MA:BBN Systems and Technologies, Cambridge, Massachusetts, USA.

Rabe, M., Jaekel, F. W., and De Gurtubai, G. G. (2001), "Modelling and Simulation for Globally Distributed Enterprises" in The 4th International EUROSIM 2001 Congress, eds. A. Heemink, L. Dekker, H. De Swaan Arons, I. Smit and T. Van Stijn, Delft, The Netherlands.

Rector, B., and Newcomer, J. M. (1997), Win32 Programming, Boston, USA: Addison Wesley Professional.

Reese, R., and Wyatt, D. L. (1987), "Software Reuse and Simulation" in Winter Simulation Conference, eds. A. Thesen, W. Grant and W. Kelton, Association for Computing Machinery Press, pp. 185-192.

Revetria, R., Blomjous, P. E. J. N., and Van Houten, S. P. A. (2003), "An HLA Federation for Evaluating Multi-Drop Strategies in Logistics" in European Simulation Symposium and Exhibition Conference, eds. A. Verbraeck and H. Hlupic, Delft, The Netherlands.

Rohrer, M. W., and McGregor, I. W. (2002), "Simulating Reality Using AutoMod" in Winter Simulation Conference, eds. E. Yücesan, C. H. Chen, J. L. Snowdon and J. M. Charnes, San Diego, California, USA: Association for Computing Machinery Press, pp. 173-181.

Ryde, M. D., and Taylor, S. J. E. (2004), "Designing Interoperating COTS Simulation Packages from the User Perspective" in UK Operational Research Society Simulation Workshop, eds. S. Robinson and S. J. E. Taylor, Hornton Grange, Birmingham, UK: ACM SIGSIM, pp. 235-241.

Schmelzer, R., Vandersypen, T., Bloomberg, J., Siddalingaiah, M., Hunting, S., and Qualls, M. (2002), XML and Web Services Unleashed, Sams.

Schmidt, S. K., and Werle, R. (1998), Coordinating Technology. Studies in the International Standardization of Telecommunication, Cambridge, Massachusetts, USA: The M.I.T. Press.

Schrage, M. (1995), No More Teams! Mastering the Dynamics of Creative Collaboration, New York, USA: Currency Doubleday (originally published as Shared Minds: The New Technologies of Collaboration by Random House, New York 1990).

Schreiber, G., Akkermans, H., Ajewierden, A., De Hoog, R., Shadbolt, N., Van de Velde, W., and Wielinga, B. (2000), Knowledge Engineering and Management, Cambridge, Massachusetts, USA: MIT Press.

Seale, C., Gobo, G., Gubrium, J. F., and Silverman, D. (2004), Qualitative Research Practice, London, UK: SAGE Publications Ltd.

Seidel, D. (1993), "Aggregate Level Simulation Protocol (ALSP) Program Status and History", Technical Report, The MITRE Corporation.

Shannon, R. E. (1975), Systems Simulation: the Art and Science, Englewood Cliffs, New Jersey, USA: Prentice-Hall.

Simon, H. A. (1965), Administrative Behaviour. A Study of Decision-Making Processes in Administration Organization, New York, USA: The Free Press.

Simon, H. A. (1969), The Science of the Artificial, Cambridge, Massachusetts, USA: M.I.T. Press.

Singhal, S., and Zyda, M. (1999), Networked Virtual Environments. Design and Implementation, New York, USA: ACM Press. Addison Wesley.

Sol, H. G. (1982), "Simulation in Information Systems Development", Ph.D. Thesis, University of Groningen, Groningen, The Netherlands.

Straßburger, S. (2001), "Distributed Simulation Based on the High Level Architecture in Civilian Application Domains", PhD thesis, University Otto-von-Guericke, Magdeburg, Germany.

Straßburger, S., Schmidgall, G., and Haasis, S. (2003), "Distributed Manufacturing Simulation as an Enabling Technology for the Digital Factory", Journal of Advanced Manufacturing Systems (JAMS), 2(1), pp. 111-126.

Swain, J. J. (1995), "Tools for Process Understanding and Improvement: Simulation Software Survey", OR/MS Today, 22(4), pp. 64-79.

Swain, J. J. (1997), "Simulation Software Survey: Simulation Goes Mainstream", OR/MS Today, pp. 35-46.

Swain, J. J. (2001), "Power Tools for Visualization and Decision Making: 2001 Simulation Software Survey", OR/MS Today, 28(1), pp. 52-63.

Szyperski, C. (1998), Component Software. Beyond Object-Oriented Programming, New York, USA: ACM Press, Addison Wesley.

Tanenbaum, A. S. (1995), Distributed Operating Systems, New Jersey, USA: Prentice-Hall International, Inc.

Tanenbaum, A. S., and Van Steen, M. (2002), Distributed Systems. Principles and Paradigms, New Jersey, USA: Prentice-Hall International, Inc.

Taylor, S. J. E., Bruzzone, A., Fujimoto, R., Gan, B. P., Straßburger, S., and Paul, R. J. (2002), "Distributed Simulation and Industry: Potentials and Pitfalls" in Winter Simulation Conference, eds. E. Yücesan, C. H. Chen, J. L. Snowdon and J. M. Charnes, San Diego, California, USA: Association for Computing Machinery Press, pp. 688-694.

Taylor, S. J. E., Gan, B. P., Straßburger, S., and Verbraeck, A. (2003a), "HLA-CSPIF Panel on Commercial Off-the-Shelf Distributed Simulation" in Winter Simulation Conference, eds. S. Chick, P. J. Sanchez, D. Ferrin and D. J. Morrice, New Orleans, Louisiana, USA: Association for Computing Machinery Press, pp. 881-887.

Taylor, S. J. E., Robinson, S., and Ladbrook, J. (2003b), "Towards Collaborative Simulation Modelling: Improving Human-to-Human Interaction through Groupware" in European Simulation Multiconference, ed. SCS, Notthingam, UK.

Tecnomatix. (2002), "eM-Plant Reference Manual 4.6", Technical Report, Tecnomatix Tecnologies Ltd.

Tewoldeberhan, T. W., Verbraeck, A., Valentin, E. C., and Bardonnet, G. (2002), "An Evaluation and Selection Methodology for Discrete-Event Simulation Software" in Winter Simulation Conference, eds. E. Yücesan, C. H. Chen, J. L. Snowdon and J. M. Charnes, San Diego, California, USA: Association for Computing Machinery Press, pp. 67-75.

Turban, E. (1995), Decision Support and Expert Systems, Englewood Cliffs, New Jersey, USA: Prentice-Hall.

Van de Ven, A. H., Emmett, D. C., and Koenig, R. (1980), "Frameworks for Interorganizational Analysis" in Interorganizational Theory, Kent, USA: Center for Business and Economic Research, Kent State University Press, pp. 19-38.

Van Til, K. P. (2003), "FAMAS.MV2: Simulatie Logistieke Performance", Technical Report, TBA Nederland, Delft, The Netherlands.

Vangheluwe, H., and De Lara, J. (2002), "Meta-Models are Models too" in Winter Simulation Conference, eds. E. Yücesan, C.H. Chen, J. L. Snowdon and J. M. Charnes, San Diego, California, USA: Association for Computing Machinery Press, pp. 597-605.

Veeke, H. P. M. (2003), "Simulation Integrated Design for Logistics", Ph.D. Thesis, Delft University, Delft, The Netherlands.

Veeke, H. P. M., Ottjes, J. A., Verbraeck, A., and Saanen, Y. (2002a), "A Simulation Architecture for Complex Design Projects" in European Simulation Symposium, eds. A. Verbraeck and W. Krug, Dresden, Germany: The Society for Computer Simulation International, pp. 221-225.

Veeke, H. P. M., Saanen, Y. A., Rengelink, W., and Verbraeck, A. (2002), "Final Report Simulation Backbone FAMAS MV2. Project 0.2 Functional Design", Technical Report, Research Report Connekt, Delft, The Netherlands.

Verbraeck, A. (1991), "Developing an Adaptive Scheduling Support Environment", PhD Thesis, Delft University of Technology, Delft, The Netherlands.

Verbraeck, A., Saanen, Y., Stojanovic, Z., Shishkov, B., Meijer, A., Valentin, E., and Van der Meer, K. (2002), " What are Building Blocks?" in Building blocks for Effective Telematics Application Development and Evaluation, eds. A. Verbraeck and A. Dahanayake, Delft, The Netherlands, pp. 8-21.

Verbraeck, A., Valentin, E., and Saanen, Y. A. (2000), "Simulation as a Real-time Logistic Control System: AGV Control with Simple++" in The New Simulation in Production and Logistics - Prospects, Views and Attitudes, eds. K. Mertins and M. Rabe, Berlin, Germany, pp. 245-255.

Waller, A. P., and Ladbrook, J. (2002), "Experiencing Virtual Factories of the Future" in Winter Simulation Conference, eds. J. L. Snowdon, E. Yücesan, C. H. Chen and J. M. Charnes, San Diego, California, USA: Association for Computing Machinery Press, pp. 513-517.

Wang, X., Turner, S. J., Low, M. Y. H., and Gan, B. P. (2004), "A Generic Architecture for the Integration of COTS Packages with the HLA" in Proceedings of the 2004 Operational Research Society Simulation Workshop, eds. S. Robinson and S. J. E. Taylor, Birmingham, UK: Association for Computing Machinery's - Special Interest Group for Simulation, pp. 225-233.

Weatherly, R., Seidel, D., and J. Weissman. (1991), "Aggregate Level Simulation Protocol" in Summer Computer Simulation Conference, Baltimore, Maryland, USA.

Wierda, F. W. (1991), "Developing Interorganizational Information Systems", PhD Thesis, Delft University of Technology, Delft, The Netherlands.

Wilson, A. L., and Weatherly, R. M. (1994), "The Aggregate Level Simulation Protocol: An Evolving System" in Winter Simulation Conference, Orlando, Florida, USA: Association for Computing Machinery Press, pp. 781-787.

Yin, R. K. (1994), Case Study Research: Design and Methods (2 ed.), Thousand Oaks, California, USA: Sage.

Zeigler, B. P., Praehofer, H., and Kim, T. G. (2000), Theory of Modeling and Simulation, San Diego, USA: Academic Press.

NEDERLANDSE SAMENVATTING (SUMMARY IN DUTCH)

Gedistribueerde simulatie is een toepassing binnen de technologie van gedistribueerde systemen die ons in staat stelt modellen aan elkaar te koppelen via computernetwerken zo dat deze kunnen interacteren tijdens uitvoering van een simulatie. Gedistribueerde simulatie wordt uitgebreid toegepast in het militaire domein, en heden ten dage bestaat er toenemende belangstelling om dit ook in de industrie te gebruiken. Ondanks dat staat het toepassen van gedistribueerde simulatie in de industrie nog steeds in de kinderschoenen. We hebben geobserveerd dat de belangrijkste reden hiervoor is dat er geen geschikte en acceptabele architectuur bestaat voor het koppelen van simulatiemodellen in de industrie. Bijgevolg verwachten we dat het bestaan van zo'n architectuur een technologische 'push' zou kunnen betekenen, die voor de gedistribueerde simulatie een 'pull' richting industrie zou kunnen genereren. Deze redenering vormt de motivering voor de doelstelling van dit proefschrift, te weten *"realiseer een architectuur voor het koppelen van simulatiemodellen en onderzoek de geschiktheid daarvan voor de industrie"*.

Om deze doelstelling te kunnen bereiken hebben we drie onderzoeksvragen geformuleerd. De eerste onderzoeksvraag had tot doel een pakket van eisen vast te stellen voor een geschikte architectuur bedoeld om simulatiemodellen binnen de industrie aan elkaar te koppelen. De tweede onderzoeksvraag had tot doel om een architectuur voor gedistribueerde simulatie te vinden, aan te passen dan wel te ontwerpen die aan deze eisen voldoet. Tenslotte was de derde onderzoeksvraag erop gericht om de praktische toepasbaarheid te testen van de architectuur die het resultaat was van de tweede onderzoeksvraag.

Teneinde de eerste onderzoeksvraag te beantwoorden, zijn we begonnen met een literatuuroverzicht om te onderzoeken wat er al gedaan is en welke soorten gedistribueerde architecturen voor simulatie er al bestaan. We hebben in Hoofdstuk 2 verschillende architecturen gepresenteerd en besproken, en we concludeerden dat op dit ogenblik de 'High Level Architecture' (HLA), een oplossing die bij defensie ontworpen en in gebruik is, de meeste geschikte op de markt is. Deze uitkomst bracht ons op de gedachte om HLA uit te kiezen en te accepteren als de meest geschikte architectuur om simulatiemodellen aan elkaar te koppelen, en deze in de industrie toe te passen. Echter, uit een literatuurstudie, en ook uit direct contact met experts, begrepen we ook dat HLA nauwelijks toegepast wordt in de industrie. Deze observatie bracht ons aan het twijfelen over de vraag of het toepassen van HLA werkelijk de manier zou zijn om onze onderzoeksdoelstelling te realiseren. Derhalve besloten we, voordat we een eisenpakket zouden gaan formuleren, eerst te testen of onze observatie juist was en als dat zo was, of deze observatie ons tot de slotsom zou kunnen leiden dat HLA niet de geschikte architectuur was waar we naar op zoek waren.

Om uit te vinden of onze observatie juist was hebben we deze als hypothese geformuleerd en voorgelegd aan een aantal experts op het terrein van de gedistribueerde simulatie, dit door middel van een vragenlijst en een onderzoek via interviews. Naast de vraag of de experts het eens waren met de hypothese, vroegen we ze ook om een motivering van hun mening, en we vroegen hen alternatieve antwoorden op onze onderzoeksvragen aan te geven. Om een precies en niet-vooringenomen antwoord te kunnen krijgen, hebben we experts uit verschillende domeinen rond HLA en simulatie uitgekozen: experts die HLA ontworpen en ontwikkeld hebben, experts uit de defensie die HLA in het dagelijks gebruik toepassen, experts uit het industriële domein met kennis van zaken over gedistribueerde simulatie en/of HLA, personen die onderzoek doen over gedistribueerde simulatie en/of

HLA, en tenslotte verkopers van COTS simulatiepakketten. De gegevens die we van de experts verkregen hebben worden in Hoofdstuk 3 gepresenteerd.

We hebben de verkregen gegevens geanalyseerd in Hoofdstuk 4. In deze analyse stelden we allereerst de voor- en nadelen vast van het toepassen van gedistribueerde simulatie in de industrie, ten tweede behandelden we de geschiktheid van reeds bestaande methoden van aanpak voor de industrie, en tenslotte stelden we een ontwerpbenadering voor voor een architectuur voor gedistribueerde simulatie in de industrie. De voorgestelde ontwerpbenadering werd geformuleerd als een pakket van eisen, bestaande uit 25 onderdelen, waaraan voldaan moet worden als men van plan is gedistribueerde simulatie te ontwerpen en te ontwikkelen voor het industriële domein. Of deze eisen adequaat zijn is beoordeeld door de experts die aan het onderzoek deelgenomen hebben, alsmede door experts die niet direkt geparticipeerd hebben. Het resulterende eisenpakket vormde het antwoord op onze eerste onderzoeksvraag. Alhoewel het pakket van eisen, dat in dit proefschrift voorgesteld wordt slechts een initiatief is, geloven we dat het een nuttig beginpunt vormt om een stimulans te vormen voor een technische 'push' van de gedistribueerde simulatie in de industrie.

Als antwoord op de tweede onderzoeksvraag hebben we in Hoofdstuk 5 een lichtgewicht gedistribueerde simulatie-architectuur voorgesteld, genaamd de 'FAMAS Simulation Backbone Architecture'. We ontwierpen en ontwikkelden de Backbone met als doel aan de haven gerelateerde gedistribueerde simulatiemodellen aan elkaar te koppelen, speciaal modellen gebouwd met de COTS simulatiepakketten die regelmatig toegepast worden in de industrie. De Backbone was zo ontworpen dat hergebruik van bestaande modellen ondersteund werd, en dat al te specialistische implementatiedetails onzichtbaar gemaakt werden voor de gebruikers hetgeen het voor hen eenvoudiger maakte om hun modellen hieraan te koppelen. Voordat we een definitief antwoord op de tweede onderzoeksvraag formuleerden, hebben we eerst een evaluatie uitgevoerd van de geïmplementeerde architectuur in het kader van het onderzoeksplan waar het voor bedoeld was. Naast het feit dat de Backbone geschikt genoeg bleek voor de twee 'case study' projecten die in Hoofdstuk 6 aan de orde gesteld zijn, waarvoor de Backbone ook bedoeld was, bleek ook een bevestigend antwoord op de derde onderzoeksvraag mogelijk. Echter, om een volledig antwoord te kunnen geven op de tweede onderzoeksvraag moest de 'FAMAS Simulation Backbone Architecture' in een breder kader geëvalueerd worden, te weten tegen het pakket van eisen voor de industrie in het algemeen dat we geformuleerd hadden.

Derhalve hebben we in Hoofdstuk 7 de geschiktheid geëvalueerd van de voorgestelde architectuur om industriële simulatiemodellen te koppelen, dat wil zeggen de 'FAMAS Simulation Backbone Architecture', aan de hand van het pakket van eisen dat voorgesteld werd in Hoofdstuk 4. We hebben vastgesteld dat door de Backbone aan vrijwel alle eisen voldaan wordt. In deze zin is het ons ook gelukt om een antwoord te verkrijgen op onze tweede onderzoeksvraag.

Nu we antwoorden verkregen hebben op al onze onderzoeksvragen, kunnen we vaststellen dat we onze onderzoeksdoelstelling bereikt hebben. We hebben een lichtgewicht architectuur gerealiseerd waarmee simulatiemodellen aan elkaar gekoppeld kunnen worden, die allereerst gericht is op het industriële domein, en die ook eenvoudiger toegepast kan worden dan HLA door de industriële gemeenschap die COTS simulatiepakketten gebruikt. We hebben vastgesteld dat gedistribueerd rekenen in zijn algemeenheid te ingewikkeld is voor toepassers van simulatie, die daar niet aan gewend zijn en die zich derhalve veel inspanningen zullen moeten getroosten om ermee om te kunnen gaan. De tijd die besteed

wordt om te leren omgaan met deze complexiteit vertaalt zich in onvoorziene kosten die wel eens te hoog zouden kunnen zijn in verhouding tot de te verwachten baten. De benadering die wij voorstellen om de inleerkosten te beperken is om de complexiteit te reduceren door specialistische implementatiedetails rond gedistribueerd rekenen buiten het zicht te houden van toepassers van simulatie. In dit onderzoek toonden we aan dat het mogelijk is een lichtgewicht architectuur te bouwen om simulatiemodellen aan elkaar te koppelen, waarmee, zoals uit twee case studies bleek, de deelnemers en de probleemeigenaren tevreden waren. We hebben ook vastgesteld dat de experts uit de industrie geloven dat daar een plek is voor gedistribueerde simulatie, en voor standaarden die de concepten beschrijven voor het koppelen van industriële simulatiemodellen. We verwachten dat er een markt voor gedistribueerde simulatie in de industrie gegenereerd zal worden als er lichtgewicht architecturen ter beschikking komen die in grote mate voldoen aan het hier geformuleerde pakket van eisen.

ÖSSZEFOGLALÓ MAGYAR NYELVEN (SUMMARY IN HUNGARIAN)

Ezen dolgozat központi témája osztott szimuláció alkalmazása az iparban. Pontosabban, kutatásunk célja *egy olyan architektúra megtervezése és implementálása, amely osztott szimulációs modellek összekapcsolására szolgál és hatékonyan alkalmazható komplex ipari szimulációs projektek kivitelezésére*.

Egy modell egy valós vagy elképzelt rendszernek a részleges leképezése. Modellek segítségével betekintést nyerhetünk egy rendszer működésébe, elemezhetjük, megváltoztathatjuk, illetve kiértékelhetjük annak működését, anélkül, hogy valós rendszerbeli változtatásokat hajtanánk végre. A számítástechnikában a modellek építésére és elemzésére szolgáló egyik legelterjedtebb módszert szimulációnak nevezzük. A szimuláció célja, hogy modellek szintjén a szóban forgó rendszert ábrázolja és annak felhasználásakor az utánzott vagy elképzelt jelenséggel azonos, vagy ahhoz hasonló élményt nyújtson a felhasználó számára, mellőzve az eredeti rendszer használatát vagy megépítését.

Osztott szimuláció esetén a tanulmányozott rendszert több modell segítségével ábrázoljuk, melyek között a kapcsolat egy úgynevezett *architektúra* segítségével valósítható meg. Osztott szimulációt főleg komplex rendszerek elemzésére alkalmaznak, elsősorban a hadügyben és az űrkutatásban. Az ipari projektek komplexitásának folytonos növekedése következtében az osztott szimuláció új lehetőségeket nyújthatna az ipar számára, jelenleg azonban mégsem nyert igazán teret ebben a szférában. Észrevételeink alapján ennek legfőbb oka egy megfelelő architektúra hiánya.

A felvetett kutatási célunk elérése érdekében a dolgozatban a következő három kérdés megválaszolására törekedtünk:

1. Melyek azok a követelmények, amelyeknek eleget kell tennie egy olyan architektúrának, amely osztott szimulációs modellek összekapcsolására szolgál?
2. Hogyan lehet egy meglévő architektúrát átalakítani, vagy egy újat tervezni és implementálni, úgy, hogy eleget tegyen a követelményeknek?
3. Hatékonyan alkalmazható-e a tervezett és implementált architektúra az iparban?

Kutatásunk első lépéseként a szimulációs szakirodalmat tanulmányoztuk. Tanulmányunk alapját a napjainkban használatos osztott szimulációs modellek összekapcsolására szolgáló architektúrák elemzése képezte. Ezen architektúrákat elemezve arra a következtetésre jutottunk, hogy az egyik legelőrehaladottabb architektúra az amerikai hadügyminisztérium által kifejlesztett HLA (High Level Architecture – magas szintű architektúra). A HLA igen elterjedt és hatékony architektúra a komplex szimulációs hadügyi projektekben, viszont megfigyelésünk alapján az iparban nagyon kevés esetben használják. Ennek oka, észrevételeink alapján, hogy a HLA és más architektúrák az ipar számára nem megfelelőek és pillanatnyilag az adott formában nehézkesen használhatóak.

Dolgozatunk első, fő részében ezen megfigyelés alátámasztására törekedtünk. Kijelentésünk tesztelése érdekében a Delphi metodológiát követtük. Szakembereket kértünk általunk összeállított kérdőívek kitöltésére valamint interjúban való részvételre. A szakemberek mindannyian egyetértettek megfigyelésünkkel. Válaszaik igazolják kijelentésünket, azaz, hogy a létező architektúrák az ipar számára nem megfelelőek. Tehát annak érdekében, hogy az osztott szimulációt az iparban is megfelelően alkalmazni

lehessen, az ipar számára is elfogadható architektúrát kell kiépítenünk. Azt, hogy milyennek kell lennie egy „megfelelő” architektúrának, a kérdőívekből és az interjúk során összegyűjtött adatokból próbáltuk kikövetkeztetni. Az osztott szimuláció terén jártas szakemberek elégséges információt nyújtottak ahhoz, hogy megfogalmazzunk egy sor követelményt, amelyeknek egy megfelelő architektúrának eleget kell tennie. Az összeállított követelmények listája szolgáltatja a választ első kutatási kérdésünkre.

A második kérdés megválaszolása érdekében a dolgozat második részében egy architektúrát mutattunk be, amelyet komplex kikötői szimulációs projektek megvalósítására terveztünk és implementáltunk. A szóban forgó FAMAS Simulation Backbone (FAMAS szimulációs gerinc) architektúra, mely osztott szimulációs modellek összekapcsolására szolgál, eleget tesz a legtöbb követelménynek. A FAMAS architektúrát alkalmaztuk két valós komplex kikötői szimulációs projektben, melyeket a dolgozatban mint esettanulmányokat mutattunk be. Habár az ipari szektornak csak a kikötő logisztikai részében használtuk architektúránkat, a pozitív kiértékelés azt sugallja, hogy más területen is megfelelően alkalmazható.

A felvetett kutatói kérdésekre adott válaszok alapján állíthatjuk, hogy sikerült megterveznünk és implementálnunk egy architektúrát, amely osztott szimulációs modellek összekapcsolására szolgál és hatékonyan alkalmazható komplex ipari szimulációs projektek kivitelezésére. Ezzel sikerült elérnünk kutatói célunkat.

CURRICULUM VITAE

Csaba Attila Boer was born in Satu Mare, Romania, on 29 October, 1975. He completed his secondary education at Kölcsey Ferenc High School, in Satu Mare, in 1994. In the same year he started his higher education at Babeş-Bolyai University, Faculty of Mathematics and Computer Science, Cluj-Napoca, Romania, where he received his B.Sc. degree in Computer Science, in 1998, and his M.Sc. degree with major in Information Systems, specialization Designing and Implementing Complex Systems, in 1999. During these years, he obtained fellowships at the Eötvös Lóránd University, and at the Computer and Automation Research Institute of the Hungarian Academy of Sciences, Budapest, Hungary within the Central European Exchange Program for University Studies (CEEPUS).

Since 1999, he has been affiliated with the Computer Science Department, Faculty of Economics at Erasmus University Rotterdam, The Netherlands. There, he worked as a researcher for one year, studying the storage and retrieval of discrete event simulation models, research that resulted in three scientific articles. Between 2000 and 2004, he was associated with the same department as a Ph.D. candidate aiming to research the area of distributed simulation and its application in industry. His topic being close to the research carried out at the Faculty of Technology, Policy and Management, Delft University of Technology, and the BETADE research program, he started to collaborate with researchers from these groups, getting involved in two joint practical case study projects. This collaboration resulted in seven joint scientific articles, presented at various international conferences. Furthermore, Csaba has maintained international contacts with researchers from the distributed simulation area. He has been invited twice to Brunel University, London to give a presentation concerning the application of distributed simulation in industry. Currently, he is working as a simulation consultant at TBA Nederland, being involved in large-scale container terminal simulation projects.

ERASMUS RESEARCH INSTITUTE OF MANAGEMENT (ERIM)

ERIM PH.D. SERIES
RESEARCH IN MANAGEMENT

ERIM Electronic Series Portal: http://hdl.handle.net/1765/1

Appelman, J.H., *Governance of Global Interorganizational Tourism Networks; Changing Forms of Co-ordination between the Travel Agency and Aviation Sector*, Promotors: Prof. dr. F.M. Go & Prof. dr. B. Nooteboom, EPS-2004-036-MKT, ISBN 90-5892-060-7, http://hdl.handle.net/1765/1199

Assen, M.F. van, *Empirical Studies in Discrete Parts Manufacturing Management,* Promotors: Prof. dr. S.L. van de Velde & Prof. dr. W.H.M. Zijm, EPS-2005-056-LIS, ISBN 90-5892-085-2,

Berens, G., *Corporate Branding: The Development of Corporate Associations and their Influence on Stakeholder Reactions*, Promotor: Prof. dr. C. B. M. van Riel, EPS-2004-039-ORG, ISBN 90 –5892–065–8, http://hdl.handle.net/1765/1273

Berghe, D.A.F., *Working Across Borders: Multinational Enterprises and the Internationalization of Employment*, Promotors: Prof. dr. R.J.M. van Tulder & Prof. dr. E.J.J. Schenk, EPS-2003-029-ORG, ISBN 90-5892-05-34, http://hdl.handle.net/1765/1041

Bijman, W.J.J., *Essays on Agricultural Co-operatives; Governance Structure in Fruit and Vegetable Chains*, Promotor: Prof. dr. G.W.J. Hendrikse, EPS-2002-015-ORG, ISBN: 90-5892-024-0, http://hdl.handle.net/1765/867

Boer, N.I., *Knowledge Sharing within Organizations: A situated and relational Perspective,* Promotors: Prof. dr. K. Kumar, EPS-2005-060-LIS, ISBN: 90-5892-X-X,

Brito, M.P. de, *Managing Reverse Logistics or Reversing Logistics Management?* Promotors: Prof. dr. ir. R. Dekker & Prof. dr. M. B. M. de Koster, EPS-2004-035-LIS, ISBN: 90-5892-058-5, http://hdl.handle.net/1765/1132

Campbell, R.A.J., *Rethinking Risk in International Financial Markets*, Promotor: Prof. dr. C.G. Koedijk, EPS-2001-005-F&A, ISBN: 90-5892-008-9, http://hdl.handle.net/1765/306

Chen, Y., *Labour Flexibility in China's Companies: An Empirical Study*, Promotors: Prof. dr. A. Buitendam & Prof. dr. B. Krug, EPS-2001-006-ORG, ISBN: 90-5892-012-7, http://hdl.handle.net/1765/307

Daniševská, P., *Empirical Studies on Financial Intermediation and Corporate Policies*, Promotor: Prof. dr. C.G. Koedijk, EPS-2004-044-F&A, ISBN 90–5892–070–4, http://hdl.handle.net/1765/1518

Delporte-Vermeiren, D.J.E., *Improving the Flexibility and Profitability of ICT-enabled Business Networks: An Assessment Method and Tool*, Promotors: Prof.mr. dr. P.H.M. Vervest & Prof. dr. ir. H.W.G.M. van Heck, EPS-2003-020-LIS, ISBN: 90-5892-040-2, http://hdl.handle.net/1765/359

Dijksterhuis, M., *Organizational Dynamics of Cognition and Action in the Changing Dutch and US Banking Industries*, Promotors: Prof. dr. ing. F.A.J. van den Bosch & Prof. dr. H.W. Volberda, EPS-2003-026-STR, ISBN: 90-5892-048-8, http://hdl.handle.net/1765/1037

Fenema, P.C. van, *Coordination and Control of Globally Distributed Software Projects*, Promotor: Prof. dr. K. Kumar, EPS-2002-019-LIS, ISBN: 90-5892-030-5, http://hdl.handle.net/1765/360

Fleischmann, M., *Quantitative Models for Reverse Logistics*, Promoters: Prof. dr. ir. J.A.E.E. van Nunen & Prof. dr. ir. R. Dekker, EPS-2000-002-LIS, ISBN: 3540 417 117, http://hdl.handle.net/1765/1044

Flier, B., *Strategic Renewal of European Financial Incumbents: Coevolution of Environmental Selection, Institutional Effects, and Managerial Intentionality*, Promotors: Prof. dr. ing. F.A.J. van den Bosch & Prof. dr. H.W. Volberda,

EPS-2003-033-STR, ISBN: 90–5892–055–0, http://hdl.handle.net/1765/1071

Fok, D., *Advanced Econometric Marketing Models*, Promotor: Prof. dr. P.H.B.F. Franses, EPS-2003-027-MKT, ISBN: 90-5892-049-6, http://hdl.handle.net/1765/1035

Ganzaroli , A., *Creating Trust between Local and Global Systems*, Promotors: Prof. dr. K. Kumar & Prof. dr. R.M. Lee, EPS-2002-018-LIS, ISBN: 90-5892-031-3, http://hdl.handle.net/1765/361

Gilsing, V.A., *Exploration, Exploitation and Co-evolution in Innovation Networks*, Promotors: Prof. dr. B. Nooteboom & Prof. dr. J.P.M. Groenewegen, EPS-2003-032-ORG, ISBN 90-5892-05-42, http://hdl.handle.net/1765/1040

Graaf, G. de, *Tractable Morality: Customer Discourses of Bankers, Veterinarians and Charity Workers*, Promotors: Prof. dr. F. Leijnse & Prof. dr. T. van Willigenburg, EPS-2003-031-ORG, ISBN: 90–5892–051–8, http://hdl.handle.net/1765/1038

Hermans. J.M., *ICT in Information Services, Use and deployment of the Dutch securities trade, 1860-1970.* Promotor: Prof. dr. drs. F.H.A. Janszen, EPS-2004-046-ORG, ISBN 90-5892-072-0, http://hdl.handle.net/1765/1793

Heugens, P.M.A.R., *Strategic Issues Management: Implications for Corporate Performance*, Promotors: Prof. dr. ing. F.A.J. van den Bosch & Prof. dr. C.B.M. van Riel, EPS-2001-007-STR, ISBN: 90-5892-009-7, http://hdl.handle.net/1765/358

Hooghiemstra, R., *The Construction of Reality*, Promotors: Prof. dr. L.G. van der Tas RA & Prof. dr. A.Th.H. Pruyn, EPS-2003-025-F&A, ISBN: 90-5892-047-X, http://hdl.handle.net/1765/871

Jansen, J.J.P., *Ambidextrous Organizations*, Promotors: Prof.dr.ing. F.A.J. Van den Bosch & Prof.dr. H.W. Volberda, EPS-2005-055-STR, ISBN 90-5892-081-X

Jong, C. de, *Dealing with Derivatives: Studies on the Role, Informational Content and Pricing of Financial Derivatives*, Promotor: Prof. dr. C.G. Koedijk, EPS-2003-023-F&A, ISBN: 90-5892-043-7, http://hdl.handle.net/1765/1043

Keizer, A.B., *The Changing Logic of Japanese Employment Practices, A Firm-Level Analysis of Four Industries* Promotors: Prof.dr. J.A. Stam & Prof.dr. J.P.M. Groenewegen, EPS-2005-057-ORG, ISBN: 90-5892-087-9, http://hdl.handle.net/1765/6667

Kippers, J., *Empirical Studies on Cash Payments*, Promotor: Prof.dr. Ph.H.B.F. Franses, EPS-2004-043-F&A. ISBN 90-5892-069-0, http://hdl.handle.net/1765/1520

Koppius, O.R., *Information Architecture and Electronic Market Performance*, Promotors: Prof. dr. P.H.M. Vervest & Prof. dr. ir. H.W.G.M. van Heck, EPS-2002-013-LIS, ISBN: 90-5892-023-2, http://hdl.handle.net/1765/921

Kotlarsky, J., *Management of Globally Distributed Component-Based Software Development Projects,* Promotors: Prof.dr. K. Kumar, EPS-2005-059-LIS, ISBN: 90-5892-088-7,

Langen, P.W. de, *The Performance of Seaport Clusters; A Framework to Analyze Cluster Performance and an Application to the Seaport Clusters of Durban, Rotterdam and the Lower Mississippi*, Promotors: Prof. dr. B. Nooteboom & Prof. drs. H.W.H. Welters, EPS-2004-034-LIS, ISBN: 90-5892-056-9, http://hdl.handle.net/1765/1133

Le Anh, T., *Intelligent Control of Vehicle-Based Internal Transport Systems*, Promotors: Prof.dr. M.B.M. de Koster & Prof.dr.ir. R. Dekker, EPS-2005-051-LIS, ISBN 90-5892-079-8, http://hdl.handle.net/1765/6554

Liang, G., *New Competition; Foreign Direct Investment And Industrial Development In China,* Promotor: Prof.dr. R.J.M. van Tulder, EPS-2004-047-ORG, ISBN 90–5892–073–9, http://hdl.handle.net/1765/1795.

Loef, J., *Incongruity between Ads and Consumer Expectations of Advertising*, Promotors: Prof. dr. W.F. van Raaij & Prof. dr. G. Antonides, EPS-2002-017-MKT, ISBN: 90-5892-028-3, http://hdl.handle.net/1765/869

Maeseneire, W., de, *Essays on Firm Valuation and Value Appropriation,* Promotor: Prof. dr. J.T.J. Smit, EPS-2005-053-F&A, ISBN 90-5892-082-8,

Mandele, L.M., van der, *Leadership and the Inflection Point: A Longitudinal Perspective*, Promotors: Prof. dr. H.W. Volberda, Prof. dr. H.R. Commandeur, EPS-2004-042-STR, ISBN 90–5892–067–4, http://hdl.handle.net/1765/1302

Meer, J.R. van der, *Operational Control of Internal Transport*, Promotors: Prof. dr. M.B.M. de Koster & Prof. dr. ir. R. Dekker, EPS-2000-001-LIS, ISBN:90-5892-004-6, http://hdl.handle.net/1765/859

Miltenburg, P.R., *Effects of Modular Sourcing on Manufacturing Flexibility in the Automotive Industry: A Study among German OEMs*, Promotors: Prof. dr. J. Paauwe & Prof. dr. H.R. Commandeur, EPS-2003-030-ORG, ISBN: 90-5892-052-6, http://hdl.handle.net/1765/1039

Moerman, G.A., *Empirical Asset Pricing and Banking in the Euro Area,* Promotors: Prof. dr. C.G. Koedijk, EPS-2005-058-F&A, ISBN: 90-5892-090-9,

Mol, M.M., *Outsourcing, Supplier-relations and Internationalisation: Global Source Strategy as a Chinese Puzzle*, Promotor: Prof. dr. R.J.M. van Tulder, EPS-2001-010-ORG, ISBN: 90-5892- 014-3, http://hdl.handle.net/1765/355

Mulder, A., *Government Dilemmas in the Private Provision of Public Goods*, Promotor: Prof. dr. R.J.M. van Tulder, EPS-2004-045-ORG, ISBN: 90-5892- 071-2, http://hdl.handle.net/1765

Muller, A.R., *The Rise of Regionalism: Core Company Strategies Under The Second Wave of Integration*, Promotor: Prof. dr. R.J.M. van Tulder, EPS-2004-038-ORG, ISBN 90–5892–062–3, http://hdl.handle.net/1765/1272

Oosterhout, J., van, *The Quest for Legitimacy: On Authority and Responsibility in Governance*, Promotors: Prof. dr. T. van Willigenburg & Prof.mr. H.R. van Gunsteren, EPS-2002-012-ORG, ISBN: 90-5892-022-4, http://hdl.handle.net/1765/362

Pak, K., *Revenue Management: New Features and Models*, Promotor: Prof. dr. ir. R. Dekker, EPS-2005-061-LIS, ISBN: 90-5892-092-5,

Peeters, L.W.P., *Cyclic Railway Timetable Optimization*, Promotors: Prof. dr. L.G. Kroon & Prof. dr. ir. J.A.E.E. van Nunen, EPS-2003-022-LIS, ISBN: 90-5892-042-9, http://hdl.handle.net/1765/429

Popova, V., *Knowledge Discovery and Monotonicity*, Promotor: Prof. dr. A. de Bruin, EPS-2004-037-LIS, ISBN 90-5892-061-5, http://hdl.handle.net/1765/1201

Pouchkarev, I., *Performance Evaluation of Constrained Portfolios*, Promotors: Prof. dr. J. Spronk & Dr. W.G.P.M. Hallerbach, EPS-2005-052-F&A, ISBN 90-5892-083-6,

Puvanasvari Ratnasingam, P., *Interorganizational Trust in Business to Business E-Commerce*, Promotors: Prof. dr. K. Kumar & Prof. dr. H.G. van Dissel, EPS-2001-009-LIS, ISBN: 90-5892-017-8, http://hdl.handle.net/1765/356

Romero Morales, D., *Optimization Problems in Supply Chain Management*, Promotors: Prof. dr. ir. J.A.E.E. van Nunen & Dr. H.E. Romeijn, EPS-2000-003-LIS, ISBN: 90-9014078-6, http://hdl.handle.net/1765/865

Roodbergen , K.J., *Layout and Routing Methods for Warehouses*, Promotors: Prof. dr. M.B.M. de Koster & Prof. dr. ir. J.A.E.E. van Nunen, EPS-2001-004-LIS, ISBN: 90-5892-005-4, http://hdl.handle.net/1765/861

Schweizer, T.S., *An Individual Psychology of Novelty-Seeking, Creativity and Innovation,* Promotor: Prof.dr. R.J.M. van Tulder. EPS-2004-048-ORG, ISBN: 90-5892-07-71, http://hdl.handle.net/1765/1818

Six, F.E., *Trust and Trouble: Building Interpersonal Trust Within Organizations*, Promotors: Prof. dr. B. Nooteboom & Prof. dr. A.M. Sorge, EPS-2004-040-ORG, ISBN 90–5892–064–X, http://hdl.handle.net/1765/1271

Slager, A.M.H., *Banking across Borders*, Promotors: Prof. dr. D.M.N. van Wensveen & Prof. dr. R.J.M. van Tulder, EPS-2004-041-ORG, ISBN 90-5892-066–6, http://hdl.handle.net/1765/1301

Speklé, R.F., *Beyond Generics: A closer look at Hybrid and Hierarchical Governance*, Promotor: Prof. dr. M.A. van Hoepen RA, EPS-2001-008-F&A, ISBN: 90-5892-011-9, http://hdl.handle.net/1765/357

Teunter, L.H., *Analysis of Sales Promotion Effects on Household Purchase Behavior*, Promotors: Prof. dr. ir. B. Wierenga & Prof. dr. T. Kloek, EPS-2002-016-ORG, ISBN: 90-5892-029-1, http://hdl.handle.net/1765/868

Valck, K. de, *Virtual Communities of Consumption: Networks of Consumer Knowledge and Companionship*, Promotors: Prof.dr.ir. G.H. van Bruggen, & Prof.dr.ir. B. Wierenga, EPS-2005-050-MKT, ISBN 90-5892-078-X

Verheul, I., *Is there a (fe)male approach? Understanding gender differences*

in entrepreneurship, Prof.dr. A.R. Thurik, EPS-2005-054-ORG, ISBN 90-5892-080-1, http://hdl.handle.net/1765/2005

Vis, I.F.A., *Planning and Control Concepts for Material Handling Systems*, Promotors: Prof. dr. M.B.M. de Koster & Prof. dr. ir. R. Dekker, EPS-2002-014-LIS, ISBN: 90-5892-021-6, http://hdl.handle.net/1765/866

Vliet, P. van, *Downside Risk and Empirical Asset Pricing*, Promotor: Prof. dr. G.T. Post, EPS-2004-049-F&A ISBN 90-5892-07-55, http://hdl.handle.net/1765/1819

Vromans, M.J.C.M., *Reliability of Railway Systems*, Promotors: Prof. dr. L.G. Kroon & Prof. dr. ir. R. Dekker, EPS-2005-062-LIS, ISBN: 90-5892-089-5

Waal, T. de, *Processing of Erroneous and Unsafe Data*, Promotor: Prof. dr. ir. R. Dekker, EPS-2003-024-LIS, ISBN: 90-5892-045-3, http://hdl.handle.net/1765/870

Wielemaker, M.W., *Managing Initiatives: A Synthesis of the Conditioning and Knowledge-Creating View*, Promotors: Prof. dr. H.W. Volberda & Prof. dr. C.W.F. Baden-Fuller, EPS-2003-28-STR, ISBN 90-5892-050-X, http://hdl.handle.net/1765/1036

Wijk, R.A.J.L. van, *Organizing Knowledge in Internal Networks: A Multilevel Study*, Promotor: Prof. dr. ing. F.A.J. van den Bosch, EPS-2003-021-STR, ISBN: 90-5892-039-9, http://hdl.handle.net/1765/347

Wolters, M.J.J., *The Business of Modularity and the Modularity of Business*, Promotors: Prof. mr. dr. P.H.M. Vervest & Prof. dr. ir. H.W.G.M. van Heck, EPS-2002-011-LIS, ISBN: 90-5892-020-8, http://hdl.handle.net/1765/920

Distributed Simulation in Industry

While distributed simulation is widely accepted and applied in defence, it has not gathered ground yet in industry. In this thesis, we investigate the reasons behind this phenomenon by surveying the expectations of industry with respect to distributed simulation solutions. Simulation models in industry are mainly designed and developed in commercial-off-the-shelf (COTS) simulation packages. The existing distributed simulation architectures in defence, however, do not focus on coupling models created in COTS simulation packages. Therefore, in order to motivate the industrial community into easily accepting and using distributed simulation, one should strive to couple models built in these packages. Furthermore, coupling these models should be possible without requiring too much extra effort from modellers.

In this thesis, based on a survey with experts in domain, we propose a list of requirements for designing and developing distributed simulation architectures that would encourage the industrial community to accept and apply distributed simulation. Furthermore, we present a lightweight distributed simulation architecture which has been successfully applied in two industrial projects, and which satisfies, to a large extent, the proposed requirements.

ERIM

The Erasmus Research Institute of Management (ERIM) is the Research School (Onderzoekschool) in the field of management of the Erasmus University Rotterdam. The founding participants of ERIM are RSM Erasmus University and the Erasmus School of Economics. ERIM was founded in 1999 and is officially accredited by the Royal Netherlands Academy of Arts and Sciences (KNAW). The research undertaken by ERIM is focussed on the management of the firm in its environment, its intra- and inter-firm relations, and its business processes in their interdependent connections.

The objective of ERIM is to carry out first rate research in management, and to offer an advanced graduate program in Research in Management. Within ERIM, over two hundred senior researchers and Ph.D. candidates are active in the different research programs. From a variety of academic backgrounds and expertises, the ERIM community is united in striving for excellence and working at the forefront of creating new business knowledge.

ERIM ERASMUS RESEARCH INSTITUTE OF MANAGEMENT

www.erim.eur.nl

ISBN 90-5892-093-3

Design: B&T Ontwerp en advies www.b-en-t.nl Print: Haveka www.haveka.nl