Kareem Abdelgawad

A System-Level Design Framework for Networked Driving Simulation

Preface

The Heinz Nixdorf Institute focuses on interdisciplinary research areas and aims to establish a new direction for the design of intelligent technical systems. Specifically, the focus is given to the development of mechatronic systems, which typically involve multidisciplinary aspects due to the close interaction of different domains, such as mechanical engineering, electrical engineering, communications engineering, and control engineering. Networked driving simulation systems represent a typical example of such complex mechatronic systems.

In general, driving simulation represents a potent supportive tool in the automotive realm. It provides a development environment that does not subject drivers or road users to hazards while examining new vehicle systems. In particular, networked driving simulation is a more advanced supportive tool, specially, when it comes to the development of autonomous and cooperative vehicle systems. Several human-driven vehicles can participate and interact in a common virtual traffic scenario. The shared virtual driving environment delivers a much closer approximation of real-world traffic interactions. Nonetheless, it is often required to develop different system variants to fulfill the requirements of diverse application scenarios, such as development and testing, drivers training, behavior analysis, and marketing. As various system components are available, users and domain-specific developers are typically confronted with high design complexity. Hence, a methodological basis is necessary in order to design platforms of networked driving simulation for different application scenarios.

In this context, Kareem Abdelgawad proposes a framework for the system-level design of networked driving simulation. The framework consists of a procedure model and a system of systems configuration software. The procedure model describes the essential development phases and the involved tasks. The system of systems configuration software encloses methods and decision-making processes to facilitate the design process. Non-expert users can use the system of systems configuration software to create system models that consider the requirements of different application scenarios. The design framework was validated by creating system models and building platforms of networked driving simulation for three different application scenarios.

The developed design framework of Kareem Abdelgawad is distinguished. The ultimate result represents a valuable contribution to the development of application-oriented networked driving simulation as a multidisciplinary system of systems.

Paderborn, im Januar 2018 Prof. Dr.-Ing. J. Gausemeier

A System-Level Design Framework for Networked Driving Simulation

zur Erlangung des akademischen Grades eines
DOKTORS DER INGENIEURWISSENSCHAFTEN (Dr.-Ing.)
der Fakultät Maschinenbau
der Universität Paderborn

genehmigte
DISSERTATION

von
M.Eng. Kareem Abdelgawad
aus *Kairo, Ägypten*

Tag des Kolloquiums:	21. Dezember 2017
Referent:	Prof. Dr.-Ing. Jürgen Gausemeier
Korreferent:	Prof. Dr.-Ing. habil. Ansgar Trächtler

Foreword

This Ph.D. thesis was carried out during my work as a research associate in the groups ‘Strategic Product Planning and Systems Engineering’ and ‘Control Engineering and Mechatronics’ at the Heinz Nixdorf Institute, the University of Paderborn in Germany, from July 2013 to July 2017. This work is the result of my multidisciplinary research and development activities within academic and industrial projects in various fields, such as mechatronic systems development, augmented and virtual reality, and driving and traffic simulation.

My sincerest thanks go to Prof. Dr.-Ing. Jürgen Gausemeier, the head of the chair of Strategic Product Planning and Systems Engineering at the Heinz Nixdorf Institute, the University of Paderborn, for his valuable advice and continued support. He has always challenged and encouraged me. Working under his supervision was a great opportunity to enhance my technical and personal skills. Similarly, I would like to thank the co-supervisor Prof. Dr.-Ing. habil. Ansgar Trächtler, the head of the chair of Control Engineering and Mechatronics at the Heinz Nixdorf Institute, the University of Paderborn, for the suggestions and ideas he provided.

I would like to thank all my colleagues, who helped me during my work as a research associate at the Heinz Nixdorf Institute. My special thanks go to the first team under the supervision of Michael Grafe: Jan Berssenbrügge, Bassem Hassan, Jörg Stöcklein, Yin Tan, and Helene Wassmann. Similarly, I would like to thank my second team under the supervision of Sandra Gausemeier: Daniel Zimmermann, Sven Henning, Patrick Biemelt, and Nico Rüddenklau. In addition, I would like to express my gratitude to the chief engineer Karl-Peter Jäker. I would like to thank the secretaries: Alexandra Dutschke, Sabine Illigen, Milena Mungiuri Meissner, and Silke Krüger. Similarly, I would like to thank the laboratory engineers Karsten Mette, Jörg Schaffrath, and Martin Leibenger.

Finally, my heartfelt thanks go to my parents and brothers for encouraging and supporting me throughout the whole journey to realize my career objectives and life goals.

Paderborn, October 2017 *Kareem Abdelgawad*

List of Published Partial Results

[AAH+14a] ABDELGAWAD, K.; ABDELKARIM, M.; HASSAN, B.; GRAFE, M.; GRÄßLER, I.: A Scalable Framework for Advanced Driver Assistance Systems Simulation. In: Proceedings of SIMUL 2014 – 6th International Conference on Advances in System Simulation, IARIA XPS, October 12-16 2014, Nice, France, 2014, ISBN 9781612083711

[AAH+14b] ABDELGAWAD, K.; ABDELKARIM, M.; HASSAN, B.; GRAFE, M.; GRÄßLER, I.: A Modular Architecture of a PC-based Driving Simulator for Advanced Driver Assistance Systems Development. In: Proceedings of REM 2014 – 15th IEEE International Workshop on Research and Education in Mechatronics, September 9-11 2014, El Gouna, Egypt, 2014, ISBN 9781479930302

[AGD+17] ABDELGAWAD, K.; GAUSEMEIER, J.; DUMITRESCU; R.; GRAFE, M.; STÖCKLEIN, J.; BERSSENBRÜGGE, J.: Networked Driving Simulation: Applications, State of the Art, and Design Considerations. In: Designs – International Journal of Engineering Designs, June 2017, MDPI AG, Basel, Switzerland, Vol. 1, No. 1, pp. 4.1-4.17 – e-ISSN 24119660

[AGG+17] ABDELGAWAD, K.; GAUSEMEIER, J.; GRAFE; M.; BERSSENBRÜGGE, J.: Interest Manager for Networked Driving Simulation Based on High-Level Architecture. In: Designs – International Journal of Engineering Designs, May 2017, MDPI AG, Basel, Switzerland, Vol. 1, No. 1, pp. 3.1-3.11 – e-ISSN 24119660

[AGS+17] ABDELGAWAD, K.; GAUSEMEIER, J.; STÖCKLEIN, J.; GRAFE; M.; BERSSENBRÜGGE, J.; DUMITRESCU; R.: A Platform with Multiple Head-Mounted Displays for Advanced Training in Modern Driving Schools. In: Designs – International Journal of Engineering Designs, October 2017, MDPI AG, Basel, Switzerland, Vol. 1, No. 1, pp. 8.1-8.15 – e-ISSN 24119660

[AGT+17] ABDELGAWAD, K.; GAUSEMEIER, J.; TRÄCHTLER; A.; GAUSEMEIER, S.; DUMITRESCU, D.; BERSSENBRÜGGE, J.; STÖCKLEIN, J.; GRAFE; M.: An Application-Oriented Design Method for Networked Driving Simulation. In: Designs – International Journal of Engineering Designs, September 2017, MDPI AG, Basel, Switzerland, Vol. 1, No. 1, pp. 6.1-6.47 – e-ISSN 24119660

[AHB+15] ABDELGAWAD, K.; HASSAN, B.; BERSSENBRÜGGE, J.; STÖCKLEIN, J.; GRAFE, M.: A Modular Architecture of an Interactive Simulation and Training Environment for Advanced Driver Assistance Systems. In: International Journal on Advances in Software, June 2015, IARIA XPS, Wilmington, Delaware, USA, Vol. 8, No. 1, pp. 247-261 – ISSN 19422628

[AHB+16] ABDELGAWAD, B.; HENNING, S.; BIEMELT, P.; GAUSEMEIER, S.; TRÄCHTLER, A.: Advanced Traffic Simulation Framework for Networked Driving Simulators. In: Proceedings of AAC 2016 – 8th IFAC Symposium on Advances in Automotive Control, June 20-23 2016, Kolmarden, Sweden, 2016, Elsevier, ScienceDirect, IFAC-PapersOnLine, Vol. 49, No. 11, pp. 101-108 – DOI 10.1016/j.ifacol.2016.08.016

[AHB+17] ABDELGAWAD, K.; HENNING, S.; BIEMELT, P., GAUSEMEIER, S.; TRÄCHTLER, A.: Networked Driving Simulation for Future Autonomous and Cooperative Vehicle Systems. In: Proceedings of AUTOREG 2017 – 8th VDI/VDE Conference on Automated Driving and Connected Mobility, July 5-6 2017, Berlin, Germany, 2017, VDI Report No. 2292, ISBN 9783180922928

[AHK+15] ABDELGAWAD, K.; HASSAN, B.; KOHLSTEDT, A.; STÖCKLEIN, J.; BERSSENBRÜGGE, J.; GRAFE, M.; GAUSEMEIER, S.; JÄKER, K.P.; TRÄCHTLER, A.: Flexible Operation Workflow of a Driving Simulation Center for ADAS Development. In: Proceedings of DSC 2015 – Driving Simulation Conference, September 16-18 2015, Tübingen, Germany, 2015, pp. 245-246 – ISBN 9783981309935

[HBA+17a] HENNING, S.; BIEMELT, P.; ABDELGAWAD, K.; GAUSEMEIER, S.; TRÄCHTLER, A.: Modellbasierte Untersuchung der Zuverlässigkeit algorithmisch bestimmter kritischer Stellen in Straßennetzwerken. In: Proceedings of AUTOREG 2017 – 8th VDI/VDE Conference on Automated Driving and Connected Mobility, July 5-6 2017, Berlin, Germany, 2017, VDI Report No. 2292, ISBN 9783180922928

[HBA+17b] HENNING, S.; BIEMELT, P.; ABDELGAWAD, K.; GAUSEMEIER, S.; TRÄCHTLER, A.: Methodology for Determining Critical Locations in Road Networks Based on Graph Theory. In: Proceedings of IFAC 2017 – 20th World Congress of the International Federation of Automatic Control, July 9-14 2017, Toulouse, France, 2017

[HGA+15] HASSAN, B.; GAUSEMEIER, J.; ABDELGAWAD, K.; BERSSENBRÜGGE, J.; GRAFE, M.: Systematik für die Entwicklung von rekonfigurierbaren Fahrsimulatoren. In: GAUSEMEIER, J.; GRAFE, M. (Ed.): 12th Workshop on Augmented & Virtual Reality in Product Development, April 23-24 2015, Paderborn, Germany, 2015, HNI Publication Series, Vol. 342, pp. 213-229 – ISSN 21955239

Abstract

Autonomous and cooperative vehicle systems represent a key priority in the automotive realm. Networked driving simulation can be utilized as a safe, cost-effective experimental replica of real traffic environments in order to support and accelerate the development of such systems. In networked driving simulation, independent and multidisciplinary systems collaborate to achieve a common task: multi-driver traffic scenario simulation. As myriad alternatives of system components are available, developers are typically confronted with high design complexity. A systematic method is required to design platforms of networked driving simulation in accordance with the requirements of different application scenarios. The aim of this thesis is to develop a *System-Level Design Framework for Networked Driving Simulation*. The design framework consists of a procedure model and configuration software. The procedure model describes the necessary phases and the involved tasks to design application-oriented platforms of networked driving simulation. The configuration software embeds functions and decision-making processes that enable non-expert users to create different system models. The design framework was validated by generating system models and developing platforms of networked driving simulation for three different application scenarios.

Zusammenfassung

Autonome und kooperative Fahrzeugsysteme sind eines der großen Zukunftsthemen in der Automobilindustrie. Vernetzte Fahrsimulation dient als Entwicklungswerkzeug für das virtuelle Prototyping dieser Systeme. Bei der vernetzten Fahrsimulation nehmen mehrere Fahrer am selben virtuellen Fahrszenario teil. Dabei kollaborieren komplexe, multidisziplinäre Teilsysteme, um den Straßenverkehr möglichst real nachzubilden. Da zahlreiche Alternativen an Teilsystemen und Systemkomponenten für vernetzte Fahrsimulation existieren, sehen sich die Entwickler vernetzter Fahrsimulation in der Regel mit einer hohen Entwurfskomplexität konfrontiert. Es besteht daher Handlungsbedarf für die Entwicklung vernetzter Fahrsimulation, deren Komplexitätsgrad auf die Anforderungen der jeweiligen Anwendungsszenarien ausgerichtet ist. Ziel der vorliegenden Arbeit ist eine Entwicklungssystematik zur anwendungsgerechten Gestaltung vernetzter Fahrsimulation. Die Entwicklungssystematik besteht aus einem Vorgehensmodell und einem Konfigurationswerkzeug. Das Vorgehensmodell beschreibt die benötigten Entwicklungsphasen und die Aufgaben jeder Phase. Das Konfigurationswerkzeug unterstützt die Entwickler sowie die Nutzer bei der Gestaltung anwendungsorientierter Systemmodelle. Die Entwicklungssystematik wurde anhand drei unterschiedlichen Anwendungsszenarien validiert.

Table of Contents

Page

Appendix

Contents

Page

1 Introduction

Autonomous and cooperative vehicle technologies attract major attention among all key automotive players. These disruptive technologies create fascinating new mobility prospects and potentially enhance traffic safety and efficiency. In principal, they can be considered as a further evolution of the Advanced Driver Assistance Systems (ADAS) [WH17, p. 3]. ADAS technologies exist with various levels of support. Driver information systems, such as lane departure warning and blind spot assistance, monitor the vehicle environment and leave the driver in full control at all times. Moreover, there are active systems that offer more benefits beyond the delivery of information and warnings, like adaptive cruise control and lane keeping assistance. These systems enable the vehicle to carry out tasks in specific situations, where the driver must be ready to override the automatic intervention at all times [KH16]. Highly autonomous and cooperative functions emerge by the integration of several assistance systems that collaborate to obtain broader assistance levels. However, some milestones towards automation and cooperation must be achieved before vehicles can be considered as fully autonomous. When this maturity stage is accomplished, vehicles can operate on their own, with no or very little intervention from human drivers. The benefits of these advanced automotive technologies can be summarized in four main aspects [WH17, p. 3]:

- Increased safety levels by eliminating mistakes of human drivers
- Enhanced traffic flow by optimizing driving routes and parameters
- Reduced fuel consumption and polluted emissions
- Facilitated mobility of elderly and handicapped persons

Achieving these goals is a vital concern of key automotive players. Governments and authorities provide regulatory guidelines and carry out or supervise necessary infrastructure modifications. In addition, other sectors are positioning themselves firmly in this field, such as automobile manufacturers and suppliers, IT providers, insurance agencies, and logistics companies. All these sectors are fully aware of the extensive market emerging from autonomous and cooperative driving. When pursuing the economic benefits substantially, they must explore new business models that best suit the emerging potential [WGP11]. However, the evolution of autonomous and cooperative vehicle technologies is particularly the responsibility of automobile manufacturers that collaborate with automotive suppliers and IT providers to create innovative solutions. The hustle to deploy these technologies onto public roads is becoming more tangible as customer's expectations rise. However, human drivers and the ability to eventually control their vehicles still may remain necessary – even in the era of autonomous and cooperative driving [BW14, p. 8]. Moreover, the vehicle's ability to adapt itself to driver behavior and expectations is essential for building confidence and acceptance. The following section defines the main problem considered in this work.

1.1 Problem Definition

Automotive technologies are characterized in general by rapid change and evolution. Vehicle manufacturers need solutions that facilitate system growth and provide flexibility in integrating new features with minimal effort. Thus, virtual prototyping became an essential step in the development process of automotive products [Gau13a, p. 11], [Mey07]. The aim is to use virtual reality techniques to create software models that can be used in early phases of the development process instead of conventional physical prototypes [TBB+95]. In addition to the offered flexibility, virtual prototyping presents substantial time and cost benefits [GEK01]. However, ADAS in particular depend on the expectations and limitations of human drivers to perform their tasks. Hence, interactive solutions are necessary to consider human drivers during the development phases. Driving simulators are interactive virtual prototyping tools that can support the automotive field [AN13], [FSA+11]. They can be used for the interactive development and testing of ADAS as well as various vehicle systems [KH16]. In addition, driving simulators can serve more human-centric purposes, such as drivers' training, demonstration and marketing, as well as studying behavior and performance of drivers.

However, in the era of autonomous and cooperative vehicle technologies, systems become more complex, while human drivers still represent an indispensable factor. The technology is not the sole concern of the key automotive players. Various automobile manufacturers revealed practically their prowess in the autonomous and cooperative vehicle technologies. Yet new methods and tools are still required for additional refining development and testing loops. For instance, it is crucial to tackle different traffic and driving strategies, as well as the interoperability between technologies of different providers. Moreover, as interactivity between human drivers increases, various related factors must be thoroughly examined, such as the ethical values, driver behavior, and customer acceptance [MGL+16]. Furthermore, there are a lot of legal hurdles that can be overcome only by providing substantial safety evidences. Conventional driving simulation lacks the realism and multi-interactivity associated with the advanced automotive technologies of future. It provides only a rough representation of the unpredictability level encountered when multiple human drivers and different systems interact in real traffic environments. Hence, it is necessary to develop corresponding complex and interactive supportive tools.

Networked driving simulation can be used to address critical aspects related to the deployment of autonomous and cooperative vehicle technologies. For instance, interaction of drivers assisted by different levels of automation or by systems from different technology providers can be analyzed. In addition, the typical applications of conventional driving simulation can be extended to consider multi-driver scenarios. Vehicle systems can be developed and tested in complex and multi-driver scenarios. Conventional driver training can be extended to multi-driver training, where a driving instructor handles several drivers simultaneously in a life-like traffic environment. Studying driver performance and behavior can deliver substantial results, if drivers don’t react only to pro-

grammed traffic participants, but also interact with other human-driven and autonomous vehicles.

In general, there are endless choices of driving simulator components that exhibit different fidelity levels. Therefore, it is quite complicated to select a suitable driving simulator that fits a certain application scenario. Some assistance exists in the literature to support users of driving simulation while determining the fidelity level of each building component [Neg07]. Nonetheless, selecting different components to build a driving simulator requires some prior knowledge to guarantee their interoperability. Additional work in the literature presents a method to configure driving simulation environments while assuring the cohesion of the building components [Has14]. However, a more complex system is composed and additional components are added when it comes to networking driving simulators. The composed system involves further independent complex systems and can be considered principally as a typical realization for the system of systems paradigm [Jam08a]. Consequently, a multidisciplinary expertise is involved during system design and realization. This necessitates broader design considerations for the emerged system of systems to overcome the acknowledged complexity. Furthermore, extended application scenarios arise with more diverging and changing requirements. Hence, a clear impediment resists system users to progress in this area. The following section reveals the central objectives of this work.

1.2 Objectives

The aim of this work is to develop "*A System-Level Design Framework for Networked Driving Simulation*". Specifically, the design framework can be used to build system models of platforms for networked driving simulation. These system models consider the requirements of different applications that make use of networked driving simulation. The design framework builds on and extends compelling methodological work from the literature that addressed the topic of driving simulation [Neg07], [Has14]. It depends on well-established concepts of systems engineering in general [GCW+13]. The key principles of system of systems engineering for an open system architecture, as well as the crucial measure of model-based systems engineering for building multidisciplinary system models are adopted in particular [Jam08a], [Aza08], [GCW+15]. A procedure model and a system of systems configuration software represent the primary components of the design framework. The procedure model discusses the necessary development phases, the involved tasks, as well as the results of each phase. The development phases are concretized through practical selection and decision-making methods. These are embedded within the system of systems configuration software to aid non-expert users while building application-oriented system models of platforms for networked driving simulation. To validate the design framework, system models for three different application scenarios are designed using the procedure model and generated using the system of systems configuration software. In accordance with these sys-

tem models, three platforms of networked driving simulation are built and tested. The following section presents the adopted approach to achieve the objectives of this work.

1.3 Approach

This rest of this work is structured in five more chapters. The following is a brief overview about the contents of these chapters towards accomplishing the objectives introduced in the previous section:

Chapter 2 analyzes the main problem considered in this work. The chapter begins with defining the significant terms and classifying the work. As they represent the motive behind this work, this chapter presents a brief discussion of the autonomous and connected vehicle technologies together with their early history. Conventional driving simulation is presented as a supportive tool in the automotive field. Furthermore, networked driving simulation is presented as an advanced supportive tool together with the potential applications, which are tightly related to the autonomous and connected vehicle technologies. In addition, this chapter discusses the development of mechatronic systems and the associated techniques. The need for systems engineering in this regard is presented and the emergence of system of systems engineering is highlighted. Finally, the requirements of the whole design framework are derived.

Chapter 3 examines the state of the art and the related methodological work. The chapter discusses and evaluates two outstanding literature approaches addressing the topic of driving simulation. These particular approaches are coupled and used primarily in the subsequent chapter. To highlight its widespread practice, this chapter presents some of the reviewed literature, where networked driving simulation was utilized for different studies and analyses. Moreover, it presents and evaluates three compelling facilities of networked driving simulation together with their application scopes. Afterwards, the call for action is derived according the state of the art analysis and evaluation. Finally, an initial solution approach is introduced to look into the topic of networked driving simulation in a systematic manner.

Chapter 4 represents the essence of this work. The chapter presents the *System-Level Design Framework for Networked Driving Simulation*. Specifically, this chapter introduces the design framework and its essential components in detail. Furthermore, it discusses the necessary development phases, the individual tasks, and the results of each development phase.

Chapter 5 presents an implementation prototype for a system of systems configuration software. In addition, a validation for the design framework is provided by generating system models for three distinct application scenarios. In accordance with these system models, three platforms of networked driving simulation are presented.

Chapter 6 provides a summary of the presented work and reveals the potential future extensions. Finally, the **Appendix** provides supplementary figures and information.

2 Problem Analysis

This chapter analyses the main problem. Section 2.1 defines the used key terms and declares the classification of the presented work. Section 2.2 presents a brief discussion of the autonomous and connected vehicle technologies and their early history. Section 2.3 presents the conventional driving simulation as a supportive tool in the automotive field. Moreover, networked driving simulation and its potential applications are presented thoroughly. Section 2.4 discusses the development of mechatronic systems and the related methods. In this regard, the utilization of systems engineering is presented together with the emergence of system of systems engineering. Section 2.5 concretizes the problem considered in this work. Section 2.6 derives the requirements of the whole design framework.

2.1 Definition of Terms and Classification of Work

This work describes "*A System-Level Design Framework for Networked Driving Simulation*". The system considered in this work is an environment or facility of networked driving simulation. A relevant and simple definition for the term "**system**" is presented in Reference [BP90, p. 3]:

> *"A system is an entity having a purpose and consisting of functional parts all helping to achieve that purpose" [BP90, p. 3].*

ULRICH and EPPINGER presented a generic model for system design and development [UE08, p. 15]. This model contains five phases: concept development, system-level design, detail design, testing and refinement, and production road-map. In particular, the result of the system-level design phase was specified by ULRICH and EPPINGER as:

> *"The output of this phase usually includes a geometric layout of the product, a functional specification of each of the product's subsystems, and a preliminary process flow diagram" [UE08, p. 15].*

In the system-level design phase, the system architecture is established before the detailed system design. This architecture illustrates how the system is composed by its constituent subsystems. Correspondingly, the term "**system-level design**" in the context of this work denotes the consistent and reproducible composition of a system from its structured subsystems [Jan12]. The significance of the system-level design is the ability to merge diverse disruptive technologies seen from a broad perspective. The general tools of the system design are system modeling concepts together with a procedure model for the analysis and description of the system [Jan12]. The focus in this work is always given to the overall system.

According to DUMITRESCU, the term "**design framework**" refers in general to the development of a procedure model and its related means, such as functions, methods, or

algorithms, as well as a supporting software tool [Dum11, p. 6]. The term "**procedure model**" covers the phases and tasks necessary to solve a certain technical problem [Dum11, p. 6]. The software tool supports users to carry out the tasks without a need to repeat comprehensive system analysis steps. The whole design framework is used to facilitate the development process of a certain complex technical system [DAG12]. The system under consideration in this context is the networked driving simulation and the main objective of the presented work is to support the application-oriented development. Driving simulators in general are designed and built for different purposes in order to address diverse questions related to the automotive field. The Oxford Dictionary of English defines the term "**simulator**" in general as:

> *"A machine designed to provide a realistic imitation of the controls and operation of a vehicle, aircraft, or other complex system, used for training purposes" [Oxf17-ol].*

This work considers the utilization of driving simulators, specifically, in multi-interactive virtual scenarios. A distinction between the terms reactive, interactive, and multi-interactive is presented in Reference [FVR+99]. In a reactive simulation scenario, the driver reacts to predetermined events of independent triggering. In an interactive simulation scenario, the occurrence and course of events depend on the actions of the driver. The term "**multi-interactive**" refers to scenarios, in which more than one driver can participate, hence, influencing the occurrence and course of events. The networked driving simulation considered in this work is composed of various constituent systems and building components connected together. At least two human drivers participate within the same multi-interactive virtual environment. SIEGFRIED made a distinction between the terms "**parallel simulation**" and "**distributed simulation**" as:

> *"Parallel simulation refers to the concurrent, independent execution of a simulation model using multiple threads, processors, or computing nodes" [Sie14, p. 50].*

> *"Distributed simulation refers to the execution of a simulation model using multiple, distributed computing nodes. Usually, the overall simulated model consists of multiple interconnected simulation models executed by the available computing nodes" [Sie14, p. 50].*

Parallel simulation uses tightly coupled computers or a single computer with multiple central processing units [MTK+09, p. 259]. It is considered typically to improve the execution performance of the simulation or to increase the reliability of the whole simulation environment. Distributed simulation uses geographically distributed computers. It is used to network existing simulators and achieve an overall simulation task [Sie14, p. 50]. The system of networked driving simulation considered in this work complies with the definition if distributed simulation. It is formed mainly via networking existing driving simulators operated by independent computers connected together through a network.

The presented work relies basically on the experiences acquired within the research project TRAFFIS. The term "**TRAFFIS**" is a German acronym that stands for Test and Training Environment for Advanced Driver Assistance Systems. The project TRAFFIS was carried out from September 2011 to May 2015 and was funded by the European Union and the Department for Economy, Energy, Industry, Mid-Tier Business, Skilled Crafts, and Trades of North Rhine-Westphalia in Germany. The consortium of the project TRAFFIS consisted of the Heinz Nixdorf Institute – University of Paderborn; dSPACE GmbH, a worldwide provider of solutions for the development and testing of control systems; Varroc Lighting Systems GmbH, a worldwide automotive supplier in the field of vehicle lighting systems; ILV UG & Co.KG, a modern traffic safety and training centers; and UNITY AG, a technology-oriented consulting company for strategies, processes and systems. The main objective of the project TRAFFIS was to support the development, testing, and training of/with ADAS, as well as to transfer the acquired experiences and project results to the automotive research and industry fields. The results of the research project TRAFFIS were utilized further within the subsequent transfer project inTraSim. The term "**inTraSim**" is a German acronym that stands for Conception of an Interactive Training Simulator for Continued Practice of Professional Truck and Bus Drivers. The project inTraSim was carried out from September 2015 to June 2016 and was funded within the cutting-edge cluster it's OWL (a German acronym standing for Intelligent Technical Systems Ostwestfalen-Lippe). The consortium of the project inTraSim consisted of the Heinz Nixdorf Institute – University of Paderborn; Fahrerakademie Paderborn, a modern training center for professional truck and bus drivers; Aerosoft GmbH, a provider of software packages for vehicles and airplanes simulation; VDL Bus & Coach GmbH, a manufacturer of a wide range of buses, coaches, and chassis modules. The main objective of the project inTraSim was to develop a driving simulator as a training means for truck and bus drivers. A head-mounted display was utilized as an advanced visualization system that provides the simulation immersion necessary for engaging training sessions. The presented work uses the results of the research project TRAFFIS and considers the recommendations concluded within the transfer project inTraSim to produce systems that serve the development of more advanced automotive technologies – more specifically, autonomous and connected vehicle technologies.

Using the design framework developed in this work, system users can create models (blueprints) of concretized solutions. These solutions fulfill the corresponding application requirements to the best extent possible according to components and constituent systems availability. As a logical subsequent step to composing system models, platforms of networked driving simulation can be realized based on the rigorous methodological basis of the design framework. The autonomous and connected vehicle technologies are the key driving force for the methodological development of networked driving simulation in this work. The following section discusses these technologies and their potential benefits. Moreover, the early history of the autonomous and connected vehicle technologies is presented.

2.2 Autonomous and Connected Driving

Advanced driver Assistance Systems (ADAS) are developed to increase traffic safety and efficiency, as well as to facilitate the driving task [KH16]. Some ADAS alert drivers to inconvenient traffic situations. Other ADAS do not only recognize traffic situations and warn the drivers, but also intervene actively in order to prevent possible collisions. Building on the ADAS development, autonomous and connected vehicle technologies are gaining very high attention in the automotive world. Autonomous and connected vehicle technologies are important revenue sources for automobile manufacturers and suppliers. The competition between automobile manufacturers is increasing to realize and promote these systems soon. Towards fully autonomous vehicles, automobile manufacturers and suppliers target different levels of automation according to the current technological state, regulatory permissions, and public acceptance. There are different scales of automation levels according to the different standard developing organizations. Table 1 shows the different automation levels defined by three major standard developing organizations [SAE17-ol], [NHT17-ol], [BAS17-ol].

Table 1: Automation levels defined by different standard developing organizations – reconstructed according to Reference [GPN17, p. 32]

Standards	Level 0	Level 1	Level 2	Level 3	Level 4	Level 5
SAE	No automation	Driver assistance	Partial automation	Conditional automation	High automation	Full automation
NHTSA	No automation	Function-specific automation	Combined function automation	Limited self-driving automation	Full self-driving automation	
BASt	Driver only	Driver assistance	Partial automation	High automation	Full automation	-

Legend SAE: Society of Automotive Engineers
NHTSA: National Highway Traffic Safety Administration
BASt: Die Bundesanstalt für Straßenwesen

The SAE J3016 standard identifies a range between automation levels 0 and 5. While automation level 0 represents full vehicle control by its human driver, automation level 5 represents full vehicle control by an automated driving system under all roadway and environmental conditions. With automation level 1, a driver assistance system carries out a specific driving task – like steering or acceleration/deceleration – and the human driver is expected to carry out all remaining driving tasks. With automation level 2, the execution of both steering and acceleration/deceleration is carried out by one or more driver assistance systems, and the human driver performs all remaining driving tasks. With automation level 3, an automated driving system carries out all aspects of the driving task, however, it is expected that the human driver responds appropriately to eventual requests to intervene. Automation level 4 is similar to level 3, but it is not assumed

that the human driver can respond appropriately to eventual requests to intervene. Automation levels of NHTSA and BASt corresponding to those of SAE are shown in Table 1 with different naming conventions. The NHTSA standard integrates automation levels 4 and 5 into one level [PRF+05], [Pea17, p. 410]. The BASt standard defines only five automation levels, where the last level is equivalent to automation level 4 of the SAE J3016 standard [GPN17, p. 32].

On the other side, connected vehicle technologies enable vehicles to communicate with each other and the world around them. These technologies supply useful information to drivers or vehicles to make safer or more appropriate decisions. Hence, connected vehicle technologies enable better management of traffic flow in general. For instance, preceding vehicles can inform following vehicles about traffic incidences in real time. There are mainly two types of communication in this regard: Vehicle to vehicle (V2V) and vehicle to infrastructure (V2I) communications [Gau13b, p. 18]. The slow vehicle warning system is an example application for V2V communication technology. This system is designed to aid the driver avoid rear-end collisions caused by unexpected slow vehicles on highways. The system does not attempt to control the vehicle to avoid the collision. The green light wave system is an example application for V2I communication technology. This system communicates with traffic lights in real-time to show drivers green wave speed recommendations [KYB14, p. 33]. Following these recommendations guarantees that drivers will reach the next traffic light at green. More promising applications for V2V and V2I communication technologies are presented in Reference [PRR10, pp. 35–37]. Automotive communication technologies can enhance the relatively limited or cautious functionalities of autonomous driving systems [GPN17, p. 34]. Figure 2-1 shows the relation between the autonomous and connected vehicles and the emergence of cooperative vehicles.

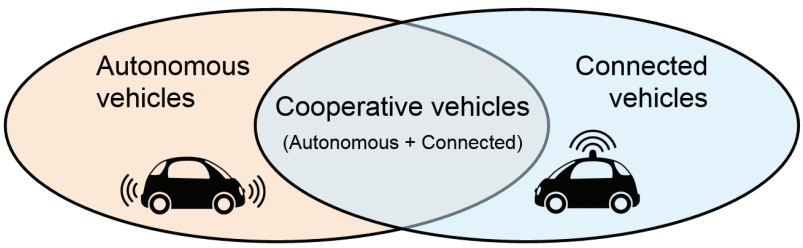


Figure 2-1: Relation between autonomous, cooperative, and connected vehicles – reconstructed according to Reference [GPN17, p. 34]

On the one hand, autonomous vehicles operate in an isolated fashion and depend on satellite data and diverse sensor technologies, like, for instance, Lidar, camera, and radar. On the other hand, connected vehicles make use of traffic information exchange but do not offer autonomous functionalities. Cooperative vehicles emerge if decisions regarding vehicle guidance depend on vehicle's sensors and other data provided by the

communication with road participants and infrastructure [GPN17, p. 34]. With V2V communication technology, autonomous vehicles can adjust their relative target positions more safely while driving with higher speeds. This reduces collision probabilities and may prevent traffic accidents decisively [SG16, p. 4]. With V2I communication technology, automated driving systems can carry out the vehicle guidance task more accurately based on received road information, such as dynamically changing speed limits [GPN17, p. 34]. Cooperative vehicles can enhance traffic safety, increase transportation efficiency, and consequently reduce emission levels beyond the limits of autonomous and connected vehicles. The following subsections give a brief, albeit not comprehensive, overview about the history of autonomous and connected driving.

2.2.1 Early History of Autonomous Driving

The thoughts about autonomous driving started not very long after the development of the first vehicle by Karl Benz in 1885 [Fra17, p. 24]. For instance, Figure 2-2 shows an advertisement of the Central Power and Light Company in 1950s. It was posted in many leading newspapers to introduce an imagination of future autonomous vehicles.

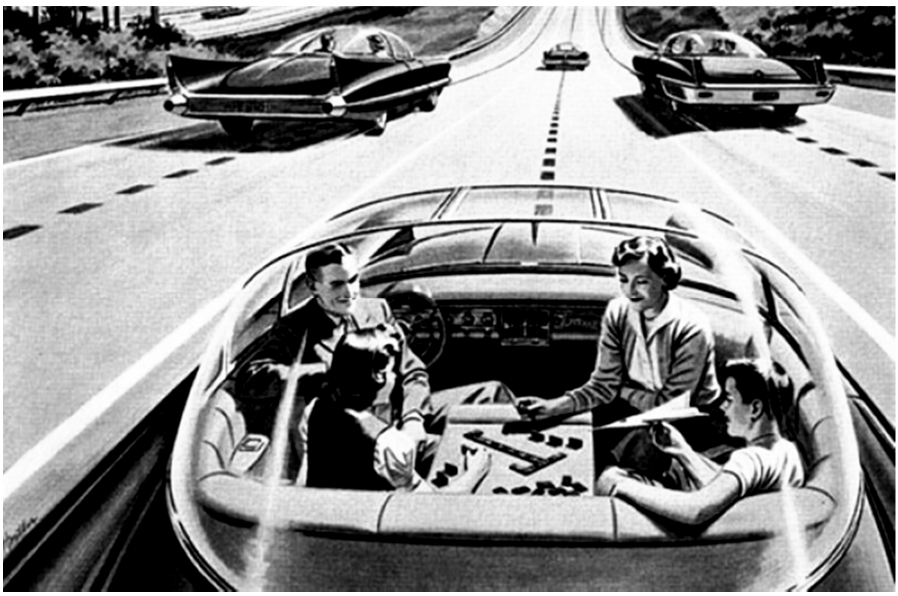


Figure 2-2: An early imagination of autonomous vehicles in 1950s [Fra17, p. 25]

During the 1950s and 1960s, General Motors conducted a series of experiments for autonomous vehicles [Fra17, p. 25]. Figure 2-3 shows an autonomous vehicle from General Motors during a test drive. The steering wheel and pedals of the vehicle were replaced with a joystick and an emergency brake. Front and rear magnetic coils were used to detect an electrical cable embedded in the road. The cable carried signals to warn

from obstacles in front of the vehicle. Meters on the dashboard displayed the vehicle's speed and the distance to the preceding vehicle. The vehicle could autonomously change lanes and apply its brakes when necessary. This can be considered as the first successful demonstration with a full-size autonomous vehicle [Bei14, p. 62].

Figure 2-3: An experimental autonomous vehicle from General Motors in 1960s [Com17-ol], [Iee17-ol]

During the 1970s, Bendix Corporation developed similar concepts for autonomous vehicles that depended on cables embedded in the road [Bei14, p. 63]. Another autonomous vehicle was developed in 1977 from the Tsukuba Mechanical Engineering Laboratory in Japan [RV15, p. 207]. This vehicle was equipped with two cameras that used analog computer technology for signal processing. The vehicle could track the white lane markers and achieve speeds up to 30 km/h. In 1986, the Bundeswehr University in Munich equipped a Mercedes-Benz vehicle (5-ton van) with an autonomous system. The system could control the steering wheel, throttle, and brakes through computer decisions that were based on real-time evaluation of camera image sequences. The autonomous system drove the vehicle at speeds up to 90 km/h [Cor11, p. 84]. In 1987, the HRL Laboratories in the USA demonstrated an autonomous vehicle that drove on tough terrain with steep slopes and vegetation [LBJ+16]. The Eureka PROMETHEUS Project was conducted from 1987–1995 and funded by the European Commission [Cal90]. It is considered as the largest R&D project in this field, in which a group of universities, vehicle manufacturers, and electronics suppliers participated. In 1995, the Bundeswehr University in Munich equipped another Mercedes-Benz vehicle (S-class passenger car) with an autonomous system. It drove in normal traffic a distance of more than 1,000 kilometers with at speeds up to 185 km/h [Nie95, p. 382]. This autonomous system depended mainly on computer vision methods and cameras looking ahead, behind, and sideways to provide a view of 360 degrees [BBC+99, p. 37]. In 1996, the University of Parma in Italy tested an autonomous vehicle that could follow the lane marking on the highway [SG16, p. 3]. The vehicle could accomplish a distance of 1900 km in Italy with a speed up to 90 km/h. The autonomous system depended mainly on two low-cost video cameras and stereoscopic vision algorithms to detect and analyze the traffic environment. Despite the costs, technological difficulties, and regulatory obstacles, it is obvious

that the motivation for autonomous vehicles remained throughout a long period of history. In recent years, many major automotive manufacturers and well-known IT providers are testing various autonomous vehicle technologies. The following subsection gives a brief, albeit not comprehensive, historical overview about connected driving.

2.2.2 Early History of Connected Driving

The vision of connected driving started almost parallel to the thoughts about autonomous driving. For instance, Figure 2-4 shows a street intersection demonstrated at Futurama; an exhibition at the New York World's Fair designed by BEL GEDDES in 1939. For increased road capacity, the concept addressed the general idea of traffic management [Bel10, ch. 1]. To control their maximum and minimum speeds, the vehicles received radio signals from a base station [Fra17, p. 24]. Moreover, the concept provided an extent of automation. The vehicles were guided through electromagnetic fields provided by electrical circuits embedded in the road.

Figure 2-4: Street intersection at the Futurama exhibition in 1939 [Bel10, ch. 1]

The project CACS (Comprehensive Automobile Traffic Control System) started in 1970 and was funded by the Japanese Ministry of International Trade and Industry (MITI) [Man12, p. 11]. The main objective of the project was reducing road traffic congestion using automotive communication systems. Drivers receive information from a base station about the traffic status and recommendations for optimal routes [Kaw90]. The idea of vehicle platooning can be traced back to the 1970s, where various concepts were in-

troduced and research conducted at the Ohio State University [Bei14, p. 63]. The research and development program PATH was started at the University California in 1986 [WH15, p. 162]. A lot of tests for connected vehicle technologies were conducted in this project in cooperation with different universities, industry firms, and state agencies. The main objective of the project was solving the problems of California's transportation systems by interdisciplinary knowledge brought by the different project partners. A practical demonstration for vehicle platooning was held in 1997 on the Interstate 15 highway in California. Twenty vehicles (cars, busses, and trucks) were platooned in close distances to show and prove benefits with respect to energy and traffic efficiency [Bei14, p. 63]. The longitudinal control of the vehicles was based on radar sensor and V2V communication in order to keep a constant and safe distance between them. To keep the vehicles within the lane, on-board sensors were used to detect magnets implemented in the road surface. The testing of technical feasibility was led from the U.S. National Automated Highway System Consortium (NAHSC). The U.S. Department of Transportation (USDOT) started the research program VSC (Vehicle Safety Communications) in 2002 [Man12, p. 12]. Communications-based systems were developed within this project for improving road safety. The program was followed by another project VSC-A (Vehicle Safety Communications – Applications) that started in 2006. The main objective of this project was testing communications-based systems for developing more transportation safety applications [Man12, p. 12]. The project PRE-DRIVE C2X started in 2008 [Man12, p. 12]. A large group of European automotive manufacturers and universities participated in this project. Detailed system specifications for vehicle communication systems were developed. Moreover, an integrated simulation model for cooperative systems was developed to estimate the safety and efficiency benefits. The follower project DRIVE C2X started in 2011 [BFF15, p. 401]. Through various field tests, this project provided a comprehensive and practical assessment of different cooperative systems. Many other projects, such as PReVENT, CVIS, SAFESPOT, COOPERS, have proven the feasibility and benefits of various applications based on automotive communication technologies [Man12].

The move towards autonomous and connected driving faced different major challenges [Bei14, p. 64], [KW16]. However, the development in recent years has been accelerated due to the emergence of supportive tools. Driving simulation is presented in the following section as a supportive tool in the automotive field in general. Yet the potential of networked driving simulation is derived as a compelling supportive tool that serves applications related to the advanced automotive technologies in particular.

2.3 Supportive Development Tools

The automotive field requires supporting tools with different specifications and capabilities according to the different development phases [NDW09]. For instance, there are different available test and simulation loops, such as Software-in-the-Loop (SiL), Model-in-the-Loop (MiL), Hardware-in-the-Loop (HiL), and Driver-in-the-Loop (DiL). This

work is concerned mainly with the DiL environments that still can include other types of simulation loops [WLL13, p. 119]. Therefore, the following subsection discusses the topic of driving simulation and its applications.

2.3.1 Driving Simulation and its Applications

Driving simulation is a type of motion simulation that has gained a lot of attention in the last few decades. The early history of driving simulation is presented in References [Has14, pp. 10–12] and [ARC11]. With interactive driving simulation, the human driver plays a central role, and consequently, influences the simulation results. Driving simulation has a lot of benefits. It provides a safe environment, where there are no hazards to the driver or other road users while undergoing critical driving conditions. Comprehensive instrumentation and recording systems for real field drives are expensive to set up and maintain. Therefore, it is feasible and less expensive to build and operate simulations in a controlled laboratory environment than conducting real field drives. Various traffic scenarios and interactions can be simulated and reproduced. Different environmental effects like foggy or snowy roads can be simulated according to the requirements of different applications. In principal, there are a variety of possible applications of driving simulation related to the automotive field due to the proven benefits and capabilities. For instance, three major research areas that make use of driving simulation are defined in Reference [FCR+11] as: engineering, psychology, and medicine. In the engineering research field, driving simulation can be used to develop and validate in-vehicle systems, such as advanced driver assistance systems. Examining road infrastructure and geometries is another relevant application. In the psychology research field, different questions about the social or psychological limitations in specific driving situations can be addressed. Moreover, driving simulators can be used to gain insight into human perception and cognition aspects. Psychiatric rehabilitation of drivers after severe accidents is an important application for this research field. In the medical research field, driving simulation can be used to study performance of drivers, who have medical problems, such as neurological or sleep disorders. Furthermore, continuous fitness assessment of truck or bus drivers represents a curial application in this field. Applications of driving simulation are categorized in Reference [ABC+14, p. 21] as: vehicle systems evaluation, human factors research, medical research, and driver education and assessment. Another categorization of possible applications is presented in Reference [Slo08] as: research, training, and entertainment. Finally, a conclusive categorization of the applications of driving simulation is evolved in Reference [Has14, p. 13] as: research (engineering, psychology, and medicine), training, and entertainment.

Figure 2-5 shows different variants of driving simulators operated at the Heinz Nixdorf Institute. These driving simulators were developed with a multidisciplinary expertise within the research project TRAFFIS that was introduced briefly in Section 2.1.

Figure 2-5: Driving simulators developed within the research project TRAFFIS at the Heinz Nixdorf Institute – University of Paderborn

The shown driving simulators have different complexity grades, and hence, different interactive simulation capabilities and applications. The PC-based driving simulator shown at the upper-left part of Figure 2-5 has no motion platform. This fixed-base driving simulator has a commercial wheel-transmission-pedals set and a driving seat to provide low-cost, but reasonable, physical feedback and control cues. The simulation environment was developed with MATLAB/Simulink. The driving simulator utilizes a 60-inch high-definition screen. The driving simulator variant shown at the upper-right part of Figure 2-5 has a pneumatic motion platform, which is composed of an inverted hexapod system. A simple motion controller regulates the extension and contraction of six pneumatic elastic tubes based on the position and orientation of the simulated vehicle. This driving simulator has a 60-inch high-definition screen and a low-cost commercial wheel-transmission-pedals set similar to the previous fixed-base driving simulator. The truck driving simulator shown at the lower-left part of Figure 2-5 has no motion platform. It incorporates a truck cabin equipped with real driving instruments. Four projectors compose the scene in front of the driver. Two displays are used to simulate the side mirror views. The ATMOS driving simulator (Atlas Motion System) shown at the lower-right part of Figure 2-5 was first developed for the German Army in 1997 to perform safety training for military truck drivers. The Heinz Nixdorf Institute of the University

of Paderborn adopted this driving simulator in 2009 in cooperation with Rheinmetall Defence Electronics GmbH. This driving simulator incorporates a complex motion platform consisting of two dynamical components providing five Degrees Of Freedom (DOF) to fully simulate vehicle lateral and longitudinal accelerations. The first dynamical component is the moving base. It has two DOF and to simulate the lateral and longitudinal acceleration of the simulated vehicle. Four linear actuators are used to control the movements in both directions. The second dynamical component is the shaker system, which has three DOF to simulate the roll and pitch angular movements and the vertical translation of the simulated vehicle. The shaker is driven with three electrical motors in a drive crank mechanism. The motion platform is equipped with a fixation system, which allows the utilization of different driving cabins, such as truck cabin or passenger vehicle cabin, so that drivers can experience various realistic control cues. The ATMOS driving simulator has an eight-channel cylindrical projection system powered by eight LCD-projectors to cover a horizontal field of view of 240 degrees. Three small displays are used to simulate the side and rear mirror views. The ATMOS driving simulator is operated by software packages developed by dSPACE and the Heinz Nixdorf Institute. These software packages consist of an operator council GUI, main vehicle and traffic models, and visualization and audio generation components. The virtual driving scenes of the four driving simulators were developed with Unity 3D; a development engine that provides rich and easy functionalities for creating interactive 3D tools. Realistic 3D models for the main simulated vehicle and the surrounding traffic participants were created. Moreover, real highways and city streets can be generated; this is necessary for an engaging simulation sessions. Figure 2-6 shows sample screen shots of the virtual environment developed with Unity3D.

Figure 2-6: Sample screen shots for a virtual environment developed with Unity3D

In addition, the 3D models are accompanied with realistic sound effects to provide reasonable acoustic feedback cues. More detailed description for the building components of the presented driving simulators and more information about the results of the research project TRAFFIS are given in Reference [Has14]. Figure 2-7 shows a driving training simulator developed within the transfer project inTraSim that was introduced briefly in Section 2.1.

Figure 2-7: Driving training simulator developed within the transfer project inTraSim at the Heinz Nixdorf Institute – University of Paderborn

This driving simulator was developed to conduct training sessions for truck and bus drivers. In particular, the central requirement of the project partners was to build a feasible solution for drivers' training with driver assistance systems. For instance, Figure 2-7 shows a realistic virtual scenario, where a bus test driver intends to turn right under the support of a blind spot assistance system. This assistance system uses radar sensors mounted at the sides of the bus to monitor the zones that are invisible to the driver. The system warns the driver if relevant objects, whether bicycles, motorcycles, or vehicles, are detected and identified. The developed driving simulator has a commercial wheel-transmission-pedals set and a real bus's driving seat. 3D environments of detailed city streets and highways have been developed corresponding to their real counterparts. This is necessary for realistic and engaging driver training according to the requirements of the project. The driving simulator is equipped with three front screens to cover a horizontal field of view of 120 degrees. Moreover, full immersion can be provided using a head-mounted display (HMD) as an alternative visualization system for the front screens [BHM+04]. HMDs have low cost and less space requirements in comparison with traditional display systems, such as screens and projectors. Furthermore, they provide a full 3D viewing and deliver user-dependent scenes. HMDs not only offer free head motion, but also a good degree of body mobility. The driving simulator is equipped with speakers and a subwoofer to produce road noise and deliver slight vehicle vibration [GB11]. Even without a motion platform, this allows drivers to experience a good extent of realistic dynamic aspects, such as accelerating/deceleration and driving over different road surfaces. Figure 2-8 shows two driving simulators utilized at two modern driving schools in Germany for training purposes.

Figure 2-8: Training driving simulators at modern driving schools in Germany

The driving simulator shown in Figure 2-8 (left) is utilized at the Ringhoff driving school in Paderborn, Germany. This driving simulator has instruments of a real passenger car and a cylindrical visualization system that provides a horizontal field of view of 180 degrees. It is used to make drivers familiar with the necessary basic tasks, such as vehicle parking and lane change maneuvers, before driving in real traffic environments. The driving simulator shown in Figure 2-8 (right) is utilized at the HAINER driving school in Unna, Germany. This driving simulator has a realistic driving platform and three wide screens that provide a horizontal field of view of 120 degrees. Similarly, it is used to conduct basic driver training sessions for beginners. The following subsection presents the topic of networked driving simulation and the potential extended applications related to the automotive field.

2.3.2 Networked Driving Simulation and its Applications

Conventional driving simulation can be used to create scenarios that involve a single human-driven vehicle and other programmed traffic participants. It is utilized as an effective tool for the typical applications discussed in the previous subsection. However, as traffic density increases, drivers experience more often situations requiring reactions or adaptation to other drivers [OS15]. In addition to road infrastructure, drivers usually regulate the driving process according to informal traffic rules that depend mainly on the behavior of other drivers [BA05]. Furthermore, in the era of autonomous and connected vehicle technologies, systems are becoming more complex and interactive, while the influence of human drivers still represents an indispensable factor. Conventional driving simulation lacks the realism and multi-interactivity associated with these advanced automotive technologies. It provides only a rough representation of the unpredictability level typically encountered, when multiple human drivers and different systems interact in real traffic environments. Hence, driving simulation must keep up in terms of interoperability and shall provide new dimensions for flexibility and scalability [HS11]. To that end, it is necessary to adapt the conventional driving simulation to establish corresponding complex and interactive supportive tools.

Creating a virtual driving environment simultaneously accessed by two or more human drivers delivers a much closer approximation of real-world traffic interactions. Networked driving simulation emerges mainly to fulfill this demand. In networked driving simulation, several human-driven vehicles can participate and interact in a common virtual traffic scenario. A comprehensive discussion about the benefits of networked driving simulation is presented in Reference [OS15]. The following is an identified set of applications that can make use of networked driving simulation. Principally, these applications stem from the reviewed utilizations of networked driving simulation reported in the literature. Furthermore, they represent a reasonable extension of the typical applications of conventional driving simulation concluded in the previous subsection. Nonetheless, entertainment is an application specifically associated with the gaming industry. This particular industry has objectives typically not related to the advanced automotive technologies [QK14]. Therefore, entertainment is excluded from the considered potential applications for networked driving simulation in this work.

- **Networked driving simulation for research and development**

Engineering research: Conventional driving simulation is used in this field mainly to examine road and infrastructure design, as well as to develop and validate in-vehicle systems [FCR+11], [BKG08], [CDF+07]. On the one hand, the design of road geometry and infrastructure is becoming more complex [KGG+11]. It is necessary to involve the influence of multiple human drivers interacting together within the same traffic environment. Networked driving simulation can be utilized in this regard to evaluate the safety levels of complex traffic environments. Moreover, dependency on assumptions typically based on the use of programmed vehicles can be reduced while analyzing various traffic flow effects, such as shock wave and rubbernecking phenomena [WP16, p. 224], [SKH15]. On the other hand, vehicle systems are becoming more cooperative. Networked driving simulation can be used to interactively validate the interoperability between technologies of different providers. Furthermore, the cooperation between vehicles equipped with different levels of automation or connectivity can be examined [GPN17, p. 32]. Demonstration and marketing of advanced automotive technologies are crucial concerns that complement the research and development phases. Networked driving simulation can be used to introduce the benefits of these technologies to potential customers in a cost-effective – yet interactive – multi-driver environment.

Psychology research: Conventional driving simulation is used in this field to address various psychological aspects related to driving. In addition to objective driving parameters, subjective feedback of a human driver is required, for instance, to gain acceptance indications of vehicle systems. It is possible to determine the effect of different traffic conditions on a human driver [YWS06]. However, drivers are more attentive and they react more realistically, when other human drivers exist in the same traffic scenario. Networked driving simulation can be used to conduct various psychological studies on groups of drivers that participate within the same traffic scenarios. For instance, psychological probes can be conducted on drivers assisted by different automation levels, and

hence, showing different road attention levels [CM11]. The psychology research can achieve more reliable results about the behavioral aspects of drivers to assist the developers of advanced vehicle technologies.

Medicine research: Conventional driving simulation is used in the medicine research field for various purposes, like, for instance, to study the effects of alcohol or other drugs on driving performance [FCR+11]. Other physiological conditions of drivers, such as reduced vigilance and situation awareness, can be examined using driving simulators. However, for comparative research results, networked driving simulation can be used to conduct medical studies simultaneously on a group of drivers interacting in common traffic scenarios. The group of drivers can include persons of different ages and experience levels. Moreover, fitness and cognitive ability of different drivers to handle the complexity of advanced automotive technologies can be assessed in safe, multi-interactive scenarios [FVR+99]. Thereby, the medical research can deliver more substantial outcomes about the physiological aspects of drivers to support the advancements in the automotive field.

- **Networked driving simulation for training purposes**

Conventional driving simulation is used in driving schools, so that drivers can start in safe and controlled environments before subjecting them to real field drives [KSK11]. They learn how to handle different situations encountered in real traffic environments. Simulation scenarios are designed to cover various aspects, such as pre-drive checks, traffic rules, and driving in hard weather conditions [IOL+13]. Introducing ADAS functionalities in driving schools has gained particular interest recently [OM15]. Driving simulators are used in this regard to make drivers familiar with their user interface, as well as to let them understand their limitations [PBT10].
However, with networked driving simulation, conventional driver training can be extended to multi-driver training. Thereby, a driving instructor handles several drivers at the same time in a life-like traffic environment. Drivers have to react to each other and adapt their driving behavior accordingly. This increased training interactivity enables drivers to act in the same way as in real traffic environments [FVR+99]. Yet training many drivers at the same time definitely adds time and cost benefits. Moreover, a driving instructor can participate interactively by performing specific maneuvers to subject drivers to sudden or unpredictable traffic situations. As participants of future traffic environments are becoming more interconnected, networked driving simulation can be used to safely and efficiently learn various advanced automotive technologies and avoid the typical overestimation of system capabilities [MMM+12]. For instance, Figure 2-9 shows schematic of a feasible platform for networked driving simulation for driving schools.

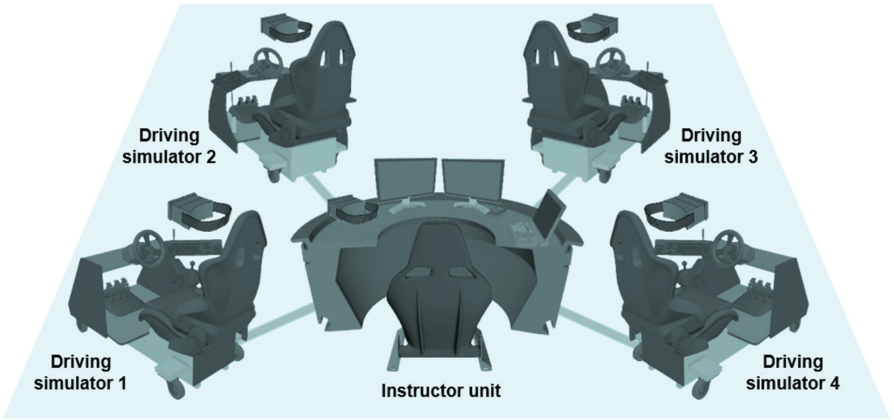


Figure 2-9: Schematic of a feasible platform of networked driving simulation with multiple head-mounted displays for training at modern driving schools

The shown platform consists of a central instructor unit and four networked driving simulators. Figure 2-10 shows a corresponding virtual environment developed with Unity3D for networked driving simulation. The scenario includes four human-driven virtual vehicles sharing the same 3D environment in a roundabout traffic situation. The vehicles can be equipped with simulation models of vehicle-to-vehicle communication systems. Beyond basic traffic instructions, training with vehicle-to-vehicle communication systems can be conducted in a safe, cost-effective environment at modern driving schools.

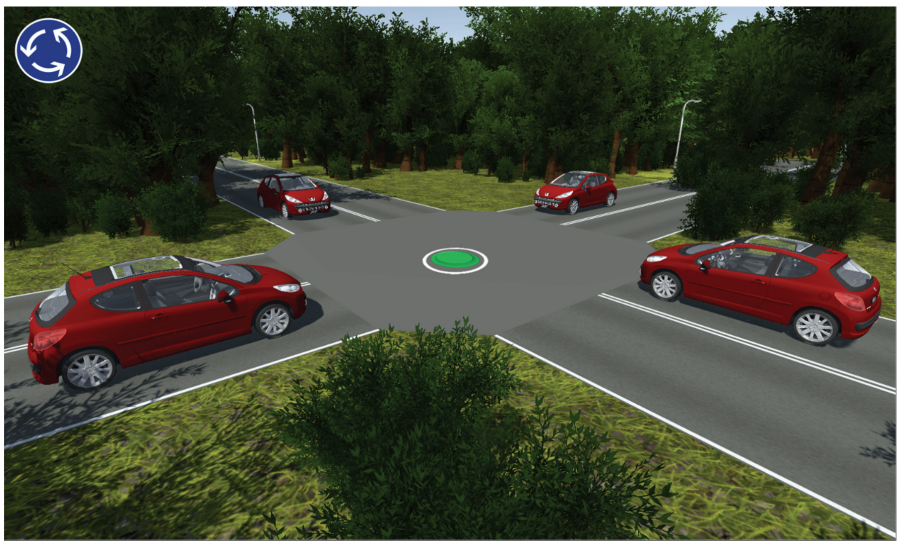


Figure 2-10: A 3D environment for networked driving simulation in driving schools

Owners of driving schools usually point to the importance of the feasibility aspect of the training facilities. That is, the entire training system shall have reasonable space requirements and remain cost-effective. Among all other components of driving simulators, motion platforms consume a major part of the available budget and require considerable space. Moreover, the operation costs of driving simulators increase substantially if motion platforms are utilized due to the associated huge power consumption. Therefore, driving simulators without motion platforms can be utilized in response to the particular cost and space requirements of driving schools. However, a compromise with these requirements must be found if the utilization of fixed-base driving simulators negatively influences the training effectiveness. In addition, Head-Mounted Displays (HMDs) can be used as the main visualization systems for the drivers [BHM+04]. Rapid advancements in the field of virtual and augmented reality pushed the produced HMDs to be cheaper and have lighter weights. These characteristics contribute to the fulfillment of the particular cost and space requirements of driving schools.

Facilities of networked driving simulation can be used to serve more than one application, provided that they exhibit a good extent of system flexibility according to a rigorous methodological basis. For instance, Figure 2-11 shows a schematic of a facility that could be utilized for various interactive applications related to the truck-platooning topic, such as development, testing, training, and demonstration.

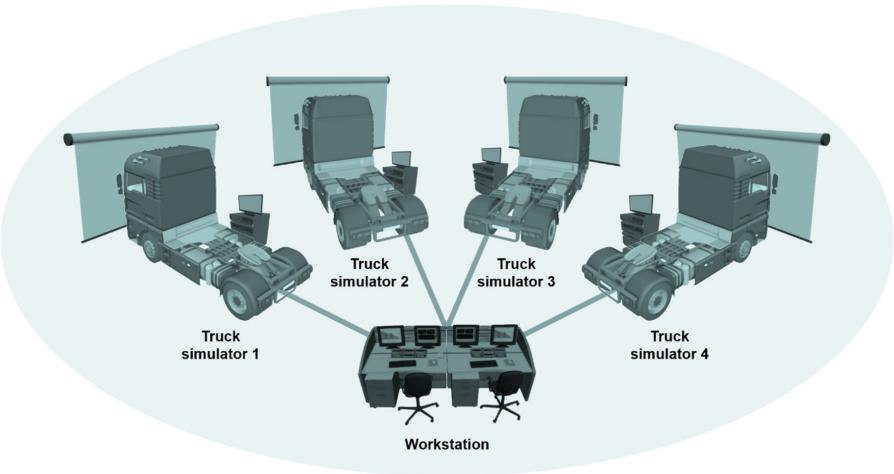


Figure 2-11: Schematic of an example platform for networked driving simulation for various interactive applications related to truck-platooning

The shown platform consists of a central workstation and four networked truck driving simulators. The workstation is utilized principally to provide control and monitoring operations on the platform. To be feasible to a far extent, the shown platform can incorporate out-of-service trucks as stationary driving platforms. In addition, the visualiza-

tion systems can be low-cost commercial projectors. Figure 2-12 shows a corresponding virtual environment developed with Unity3D for networked driving simulation.

Figure 2-12: A virtual environment for networked driving simulation for various applications related to truck-platooning

Figure 2-14 shows another schematic of a platform for networked driving simulation that can be utilized to address diverse questions related to the evolution of traffic environments in general and the advancement of automotive technologies in particular.

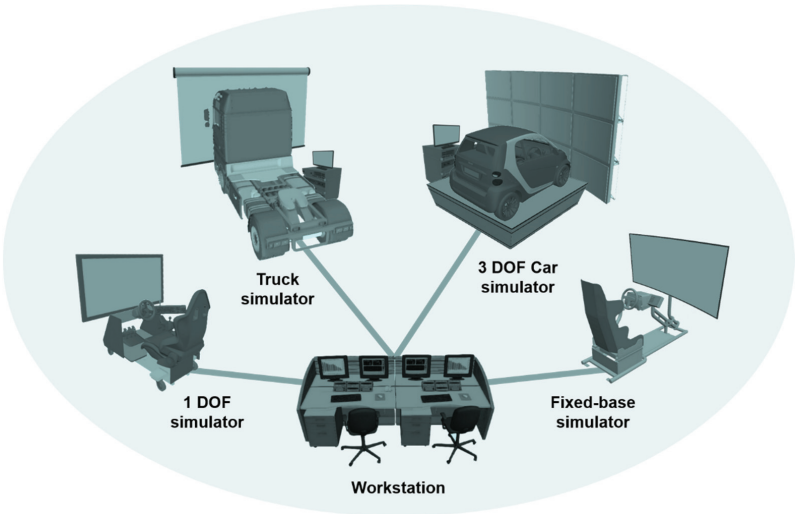


Figure 2-13: Schematic of a platform of networked driving simulation with mixed-fidelity levels for diverse automotive applications

The shown example schematic combines systems of different complexity grades and fidelity levels in one driving simulation facility in order to serve various applications. More specifically, the platform consists of a passenger car simulator, truck simulator, fixed-base driving simulator, and a workstation. In addition to typical control and monitoring operations, the workstation can include a database console that captures and saves simulation data for after-action-review and analysis.

The aim of this work is to develop the methodological basis necessary to establish platforms of networked driving simulation for various application scenarios. Yet driving simulators in general are complex mechatronic systems that involve various development disciplines. The following section highlights the development process of mechatronic systems and introduces some of the available relevant methods and guidelines.

2.4 Complex Mechatronic Systems

The term "Mechatronics" was first initiated from Japanese engineers from the Yasukawa Electric Corporation in 1969 [Heg10, p. 2]. Basically, this term denotes a merger for the disciplines of mechanics and electronics [HTF96]. Typical mechanical systems are designed to generate motions or to transfer forces or torques. These systems are driven by inputs, such as pressure, force, torque, or heat, to produce the desired outputs. Mechatronic systems emerge principally if electrical components, such as sensors and electrical drives, are integrated with the typical mechanical systems [Heg10, p. 3]. Adding electrical components to mechanical systems increases the reliability, precision, and/or performance of the whole system [SK10, p. 6].

Recently, the term "Mechatronics" is used more generally to describe the integration between the disciplines of mechanical, electrical/electronic, control, and software engineering [Sch89], [MDR91]. There are a lot of definitions for the term "Mechatronics" in the literature [Ise05, p. 5]. Almost all definitions agree that mechatronics is a multidisciplinary domain. Mechatronic systems make use of the capabilities of the combined fields. In particular, the presented work adopts the definition of mechatronics that was introduced by HARASHIMA et al. as:

> *"The synergetic integration of mechanical engineering with electronic and intelligent computer control in the design and manufacturing of industrial products and processes" [HTF96, p. 1].*

This particular definition was recognized by the VDI-guideline 2206 entitled "Design methodology for mechatronic systems" [VDI2206]. According to this guideline, a typical mechatronic system structure is composed of four main units: basic system, sensors, actuators, and information processing unit as shown in References [VDI2206] and [Dum11, p. 7]. These four units form an internal closed control loop for the mechatronic system. Figure 2-14 depicts the primary structure of a typical mechatronic system according to Reference [VDI2206, p. 14]. The basic system can be a mechanical, electro-

mechanical, hydraulic, or pneumatic component. Yet a combination of these components can compose the basic system too. The sensors are used to measure the variables of the basic system and the conditions of the surrounding environment. The measurements of the sensors are forwarded to the information processing unit. Accordingly, this unit prepares the inputs for the actuators that regulate and change the states of the basic system.

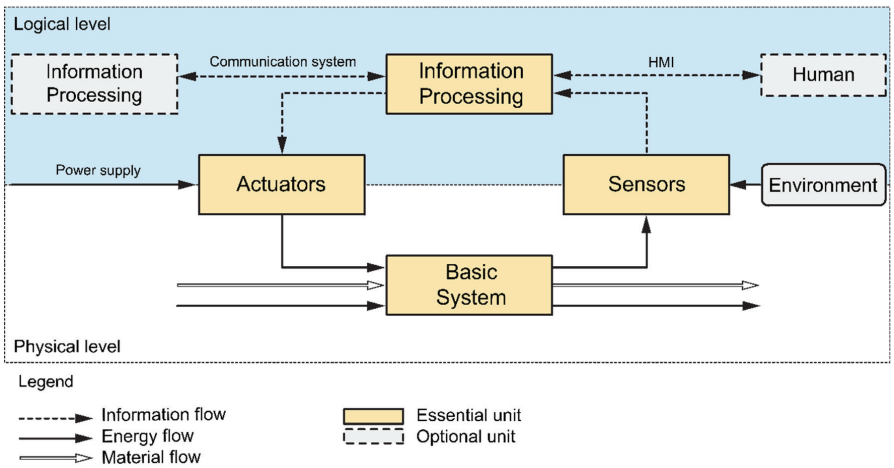


Figure 2-14: Structure of a typical mechatronic system according to Reference [VDI2206, p. 14], [Dum11, p. 8], [Has14, p. 18]

The mechatronic system interacts with the surrounding environment basically through two interface types [Dum11, p. 7]. On the one hand, a Human-Machine-Interface is used to handle the system and set its parameters via a human operator or a user. On the other hand, a communication system is used to exchange information with other surrounding technical systems. The main units of the mechatronic system are connected together using three flow types:

- **Information flow:** This denotes the exchange of information between the units of the mechatronic system, such as the measured system variables or environment conditions.
- **Energy flow:** This denotes the transfer of energy between the units of the mechatronic system, such as mechanical, thermal, or electrical energy.
- **Material flow:** This denotes the exchange of materials between the units of the mechatronic system, such as gases, liquids, or solids.

The development of mechatronic systems is challenging. Developers have to handle various aspects from different domains [GB14]. The following subsection presents the VDI-guideline 2206 established to mitigate the pitfalls of mechatronic systems development.

2.4.1 Development of Mechatronic Systems

The nineteenth century showed a rapid revolution in the development of mechanical systems [Ise05]. The actual development of mechatronic systems can be traced back to the 1980s [Ise05]. This was accompanied by the introduction of digital control systems for electrical drives, industrial robots, steam turbines, etc. Sensors, actuators, microcontrollers were integrated with the conventional mechanical elements. Precise electromechanical systems, such as video recorders and printers, were introduced and presented a wide step towards the development of more complex mechatronic systems [Ise05]. A comprehensive discussion about the historical development of mechanical, electronic, and mechatronic systems is presented in Reference [Ise05]. According to the literature review, there are a lot of development models and methods for mechatronic systems. A comparison between different significant development approaches is presented in Reference [MB03].

In particular, the VDI-guideline 2206 entitled “Design methodology for mechatronic systems” has proven its convenience and practicality for the development of mechatronic systems [MB03]. The objective of this guideline is to provide a methodological basis for developers of mechatronic systems. This basis includes generic procedures, methods, and tools that can be used to facilitate the development process of mechatronic systems. Specifically, the VDI-guideline 2206 proposes a procedure model that consists of three main components: V-model, problem-solving cycle, and process modules [GM03]. In particular, the V-model of the software engineering domain was adopted by the VDI-guideline 2206 on the macro level to accommodate the development of mechatronic systems. Figure 2-15 shows the V-model as a macro-cycle according to the VDI-guideline 2206 [VDI2206, p. 29].

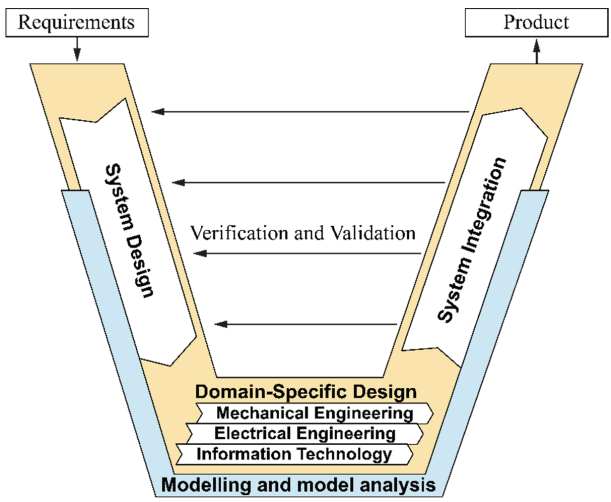


Figure 2-15: The V-model on the macro-level [VDI2206, p. 29], [Has14, p. 19]

The V-model uses a top-down approach in the system design phase to divide the overall functionality of the system into sub-functions [GM03]. A bottom-up approach is used in the system integration phase to merge the results of the preceding domain-specific design phases. The V-model refers to the necessary continuous verification and validation processes between the system design and system integration phases. Moreover, it emphasizes the significance of modeling and model analysis and their integration with the development process to manage the complexity and heterogeneity of mechatronic systems. The different components and phases of the V-model are described as follows [VDI2206, p. 29], [GM03]:

- **Requirements:** The determined requirements are the starting point of the product development. The requirements help to define the essential development tasks. Moreover, they are used later as a measure to evaluate the developed product.
- **System design:** The objective of this phase is to define a cross-domain solution concept that describes the main physical and logical characteristics of the system. The overall system functionality is divided into further sub-functions. Different methods and tools can support while handling the various involved disciplines [GM03], [VDI2206].
- **Domain-specific design:** The objective of this phase is to concretize the developed cross-domain solution concept from the perspective of each involved specific domain. Detailed design specifications are necessary in this phase to guarantee the desired performance level of the system.
- **System integration:** The objective of this phase is to integrate the concretized design results of the involved specific domains in an overall system. The interrelations between the specific domains must be considered in this phase [GM03].
- **Verification/validation:** The progress must be checked continuously using the specified solution concept and the requirements. That is, the actual system characteristics are compared to the requirements
- **Modeling and model analysis:** The properties of the system must be studied with system models using modeling and simulation software tools.
- **Product:** The designed system or product is the result of the macro-cycle. The product is not particularly a finished final solution. It can represent an intermediate level before the final solution, i.e., the product can be still under a continuous maturity progress.

There are different fields involved during the development of mechatronic systems as indicated by the domain-specific design phase of the V-model. As more complex and mechatronic systems evolve, the development process becomes a highly interdisciplinary task [ABD+14, p. 117]. The following subsection presents a well-established domain-spanning specification technique for complex mechatronic systems.

2.4.2 Specification Technique of Complex Systems

During the conceptual design of complex mechatronic systems, experts from the different involved domains work together to develop an entire solution. The conventional existing design methods of mechatronic systems typically do not consider the ever increasing interdisciplinary nature of these systems [ABD+14, p. 117]. Complex mechatronic systems consist of various components or partial models. A partial solution developed by experts from a specific involved domain may be optimal only for this particular domain but not for the entire system. Therefore, conventional design methods do not guarantee that the integration of all domain-specific solutions leads to the best possible entire solution [ABD+14, p. 117]. Moreover, the integration process of the partial solutions consumes time and is usually prone to failures. Furthermore, the defined requirements usually describe the desired overall system characteristics or behavior. It is often challenging to correlate the individual items of requirements to the involved domains [Has14, p. 20]. Hence, a domain-spanning method is required to develop an entire solution for complex mechatronic systems, i.e., principle solution, while still considering the individual involved domains [ABD+14, p. 117]. The method shall ensure a fundamental understanding of the entire system by the all involved experts at early phases of the development process. The principle solution shall describe the basic structure, operation modes, and behavior of the entire system.

There have been various attempts to establish adapted methods for the development of complex mechatronic systems. An overview of some significant attempts is provided by GAUSEMEIER et al. [GFD+09, p. 207]. The examination of the existing methods is concluded as:

> *"The analysis of the current state of the art shows that there are a lot of approaches on specifying mechatronic systems. One part of the approaches focuses on kinematic, dynamic, and controlling behavior. Other approaches give priority to communication relations, operating procedures of the system, and state transitions. All of the analyzed approaches just fulfill a single part of the requirements on the addressed specification technique, stated in Sect. 3. This applies especially for the aspect of a holistic description of the principle solution. Furthermore, the analyzed approaches do not provide a widespread transition from the domain-spanning specification towards the domain-specific concretization." [GFD+09, p. 209]*

Therefore, the specification technique CONSENS has been developed within the Collaborative Research Center CRC 614 "Self-Optimizing Concepts and Structures in Mechanical Engineering" of the University of Paderborn. The term "**CONSENS**" stands for Conceptual Design Specification Technique for the Engineering of Complex Systems. It is a well-established method for the domain-spanning development of principle solutions for complex mechatronic systems. The method divides the description of the

principle solution of a complex mechatronic system into eight interrelated aspects: requirements, environment, system of objectives, application scenarios, functions, active structure, shape, and behavior [ABD+14, p. 119]. These aspects can be considered in the form of partial models, as shown in Figure 2-16.

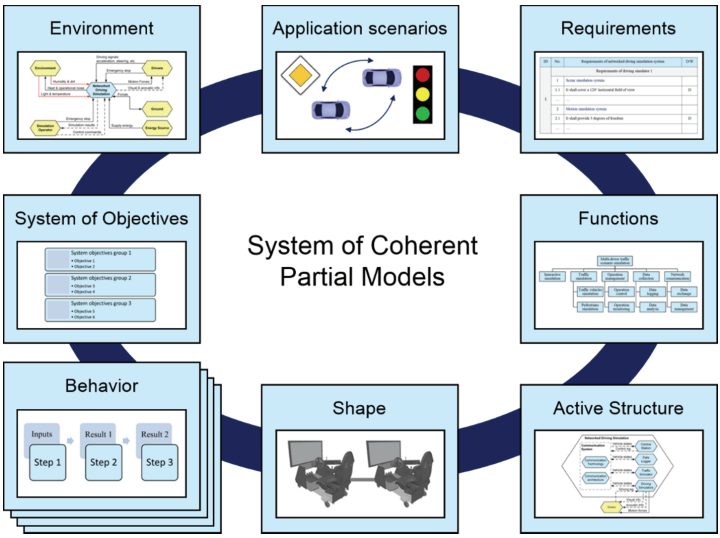


Figure 2-16: The coherent partial models of the CONSENS specification technique – reconstructed according to Reference [ABD+14, p. 120], [GGT13, p. 3]

The CONSENS partial models are described as follows [ABD+14, pp. 119–127]:

- **Environment:** This partial model considers the system as a black-box. It specifies mainly the interrelations between the system and its environment. The interrelations between the system and the environment elements are denoted as flows of three types: information flows, energy flows, and material flows. Elements of the environments can be users and other technical systems. Environmental conditions, such as temperature and humidity, are considered in this partial model. Moreover, possible influences of disturbing impact on the system, such as noise or vibrations, are specified. The identification process of the environment elements and their influences can be supported with catalogues or checklists. Specifying all environment elements and analyzing all possible external influences ensure that the final system will work properly within its intended environment.

- **Application scenarios:** This partial model describes the common operation modes of the system and the corresponding behaviors. Each application scenario addresses a specific situation and the expected system reaction. For more illustration, each application scenario can be complemented with a sketch or image. Specifying the application scenarios supports the determination of system requirements considered in the next partial model.

- **Requirements:** This partial model is based on the main problem definition as well as the two previous partial models – environment and application scenarios. This partial model includes all requirements that shall be fulfilled by the final system. All system requirements are listed under categories in a table. There are various table forms that support the identification of requirements [PBF+07], [ABD+14, p. 122]. Each item within the requirements list has a unique ID and it is specified with parameters and explicit values. The requirements can be denoted as demands or wishes. Eventually, system requirements can be divided into functional and non-functional requirements. The partial model of requirements forms a basis for the validation and verification processes in further development phases.

- **Functions:** This partial model is based principally on the determined requirements. It describes the overall functionality of the system. This overall functionality is further subdivided hierarchically into sub-functions. The functions are realized by solution patterns. After the definition of the overall functionality and the sub-functions, a classification scheme can be used to compose an overall solution. The morphological box defined by ZWICKY is a well-established classification scheme that can be used in this regard [Zwi69], [PBF+07]. In this classification scheme, the sub-functions and the available solutions are inserted into the rows of a morphological matrix. The solutions of sub-functions are combined systematically to obtain the overall solution. Only compatible solutions should be combined together.

- **Active structure:** This partial model is built based on the defined functions and the combined solutions. The active structure concretizes the internal structure of the system in contrast to the environment partial model, which considers the system as a black box. It defines the system elements and their relationships and attributes. If necessary, environment elements can be included to show their relationships with the internal system elements.

- **System of objectives:** This partial model describes the external and inherent objectives of the system and their interrelations. The external objectives are set from users or other systems. The inherent objectives refer to the desired operation characteristics of the system. The objectives can be further subdivided hierarchically into sub-objectives. The external and inherent objectives strived by the system at a particular operation instance are called internal objectives. The mutual influences between the internal objectives can be modeled in an influence matrix. The eventual negative influences between the internal objectives indicate that a self-optimization may be necessary. A prioritization of the objectives may be necessary if there are realization conflicts during system operation. The selection of the internal objectives and the prioritization process take place during system operation.

- **Shape:** This partial model presents a first description for the body structure of the system. It is crucial to model the shape of the system during the domain-spanning conceptual design. Thereby, eventual geometrical restrictions can be detected from

the experts of the involved domains at the early development phase. In mechanical engineering in particular, the shape and active structure partial models represent the core of the principle solution. Creating the shape partial model can be supported with 3D CAD programs.

- **Behavior:** This partial model describes the behavior of the system. There is a group of three behavior partial models: behavior–states, behavior–activities, and behavior–sequence. The behavior–states partial model describes the possible states, state transitions, and the events invoking the state transitions. The behavior–activities partial model describes the tasks of the system that are performed during operation to carry out a self-optimization process. The behavior–sequence partial model describes the interactions between the system elements. The messages exchanged during the interactions are modeled in a chronological order. In addition to these three behavior types, other types of system behavior can be specified eventually, such as system kinematics and dynamics.

Although there is a certain order, it is possible to alternate between these partial models to perform enhancement iterations [ABD+14, p. 126]. However, the partial models should be kept interrelated to form a coherent system during their processing and analysis. The main result is a principle solution that can be considered as a communication and cooperation basis between the experts of the involved domains [ABD+14, p. 119]. This basis can be used for further domain-specific design and development phases. However, in the era of intelligent and self-governing systems, it is not only necessary to develop systems according to standard methods, but also to manage systems from broader perspectives throughout their entire lifetime [GAC+13]. The following subsection presents the benefits of system engineering in this regard. Moreover, the paradigm of systems of systems and the emergence of system of systems engineering are introduced.

2.4.3 Systems Engineering and Systems of Systems

Systems Engineering (SE) is an interdisciplinary field combining engineering and business aspects for the design and management of complex systems during their life cycles. The International Council on Systems Engineering (INCOSE) defines the Systems Engineering as:

> *"Systems Engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem." [WRF+15]*

Systems Engineering cannot be considered as a very new engineering branch [GCW+15]. SE has been practiced for decades to analyze technologies or safety-related issues that cannot be handled using conventional design and prototyping methods. The demand for Systems Engineering increased considerably in recent years [GCW+15]. According to the INCOSE, Systems Engineering adopts an interdisciplinary process that consists of seven tasks: *S*tate the problem, *I*nvestigate alternatives, *M*odel the system, *I*ntegrate, *L*aunch the system, *A*ssess performance, and *R*e-evaluate [WRF+15], [BG98]. These tasks are summarized with the acronym *SIMILAR* according to the first letters of the seven tasks respectively. It is not necessary to go through this interdisciplinary process sequentially. The seven tasks can be carried out in a parallel or sequential manner. A comprehensive discussion about these tasks is presented in Reference [BG98].

SE considers the system to be developed as well as the associated project. On the one hand, the system aspect covers the entire development process that includes some particular tasks, such as requirements definition and management, system design, system analysis, system simulation, and system validation [GCW+15]. Complex technologies and applications necessitate the engagement of interdisciplinary teams throughout the entire development process. On the other hand, the project aspect covers the coordination of activities while considering the resources, time, and costs to ensure that the goals of system development are achieved. Figure 2-17 depicts these two aspects and the related tools.

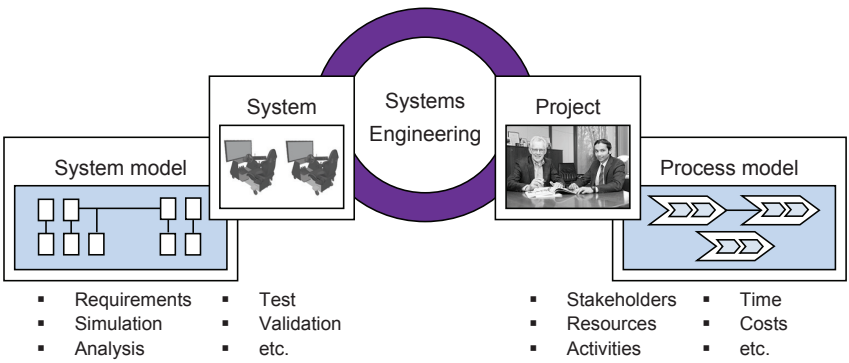


Figure 2-17: Joint analysis of the system and project aspects in system engineering – reconstructed according to Reference [GCW+15], [GCW+13]

With the help of the system and project aspects, a consistent and multidisciplinary description of the system can be created. This particular description leads to a unique system model [GCW+15]. This usually includes a graphic representation (diagram) and an internal computer representation as a data model (repository). Although data of the repository appears only once, it can be used repeatedly in the diagrams with different interpretations in order to create various views of the system. In this regard, Model-based Systems Engineering (MbSE) places a multidisciplinary system model at the thrust of

the development process. The MbSE does not neglect other domain-specific models of the developed system, but it integrates those using proper interfaces. There are different methods (e.g. CONSENS, SysMod) and languages (e.g. SysML) to create system models. These can be combined eventually with each other. According to the survey presented in Reference [GCW+15], networking is one of the most significant features of future complex systems. Systems provided with this feature are able to communicate and cooperate with other systems. The geographic proximity of networked systems is no longer a requisite due to the rapid evolution of communication systems. The overall functionality of networked systems can be obtained only by the interaction of the individual systems. However, neither the network nor the contribution of the individual systems should be static [GCW+15]. The interconnections between the networked systems must be modifiable to permit eventual changes of the overall functionality. In this regard, there has been a growing interest in a class of complex systems that themselves are composed of independent systems, i.e., systems of systems (SoS) [Dim10]. Based on the literature review, numerous definitions exist for the evolving term system of systems [Jam08a]. The following are two appealing definitions from the literature:

- **SoS definition 1:** *"Systems of systems exist when there is a presence of a majority of the following five characteristics: operational and marginal independence, geographic distribution, emergent behavior, and evolutionary development." [Jam05], [Jam08a]*
- **SoS definition 2:** *"Systems of systems are large-scale concurrent and distributed systems that are comprised of complex systems." [CF01], [Jam08a]*

A comprehensive literature survey for more definitions and discussions about SoS is presented in References [Jam05] and [Jam08a]. Nonetheless, the complexity of designing a system of systems is daunting. The primary challenge is to pursue a synergy between the constituent systems in order to attain the desired system goal. Several concepts and design considerations for the theme of SoS have been addressed in the literature [Jam08a]. Basically, no engineering field is more essentially required than systems engineering (SE) to overcome the pitfalls of SoS design [Jam08a]. Extending systems engineering concepts to accommodate the SoS paradigm is discussed in References [Joh08] and [KRU+15]. This led principally to the emergence of System of Systems Engineering (SoSE) [Jam08b], [Kea08], [WS08]. One definition for SoSE with particular reference to enterprise management according to CARLOCK and FENTON is:

> *"Enterprise system of systems engineering is focused on coupling traditional systems engineering activities with enterprise activities of strategic planning and investment analysis." [CF01], [Jam08a]*

Yet architecting SoS environments through an evolutionary process is a crucial requirement in this regard. To that end, the open systems approach is adopted by SoSE [DK08], [Jam08a]. This approach defines the general key principles for an open system

architecture suitable for future evolution [Aza08], [Jam08b, pp.193–195]. The eight open systems principles according to AZANI are [Aza08]:

- **Open interface principle:** The SoS shall have open boundaries that allow the exchange of information, material, and energy with other interacting systems.
- **Synergism principle:** The interaction between the constituent systems of the SoS shall have a greater outcome than the sum of their individual results.
- **Self-government principle:** The SoS shall maintain and develop their internal order without intervention by external sources. This can be realized through homeostasis, cybernetic control, or self-organization.
- **Emergence principle:** Coherent and novel structures, patterns, and characteristics shall evolve during the self-organization of the SoS.
- **Conservation principle:** The energy and mass shall conserve within the SoS. They can neither be created nor destroyed. However, they can be transformed into other forms.
- **Reconfiguration principle:** The SoS shall have the capability of reconfiguration and adaptation to sustain themselves with respect to changes of the environment.
- **Symbiosis principle:** The constituent systems of the SoS shall have symbiotic interrelationships. The sustainment of the SoS depends on the symbiotic collaboration of their constituent systems.
- **Modularity principle:** The SoS shall be composed of independent constituent systems. This leads to more flexibility for the evolution and modification of the SoS.

Following these principles results in a flexible and open SoS that can be modified easily by exchanging the constituent systems and/or altering the characteristics of some building components [Jam08a]. Moreover, the open SoS remains able to exchange information, energy, and material with the environment throughout its entire lifetime. Yet building system models before establishing real SoS is one of the significant measures recommended by SoSE [MZM+08]. The modeling process itself is challenging due to the complexity of the independent constituent systems. The following section elaborates the problem in light of the topic of networked driving simulation as a typical system of systems.

2.5 Problem Description

Research and development in the automotive sector in general is multidisciplinary by its nature [FCR+11. 1, p. 4]. Yet the problem is becoming more complex due to the introduction of future highly automated and connected vehicle systems. Performing an effective system analysis to support decisions becomes challenging and unmanageable due to the involvement of heterogeneous distributed systems. DELAURENTIS showed how the

future traffic environment is considered as a system of systems problem [Del08]. JAMSHIDI presented a relevant and quite simple definition for system of systems in this regard as:

> *"Systems of systems are large-scale integrated systems that are heterogeneous and independently operable on their own, but are networked together for a common goal." [Jam08a]*

An efficient tool is essential for the collective understanding of the possible interactions of the heterogeneous constituent systems [Del08]. Thereby, decisions in technological, socio-economic, and regulatory contexts can lead to the expected outcomes. As discussed thoroughly in Section 2.3, networked driving simulation can present an intelligible solution to examine various aspects concerning the future traffic and transportation systems. Networked driving simulation satisfies this definition for system of systems. Specifically, two or more driving simulators exchange information and share a common virtual environment, where human drivers interact with each other. Each participating driving simulator per se represents an independent system. However, a common system goal is accomplished through the collaboration within a system of systems environment. In a nutshell, the ultimate goal is to simulate multi-driver traffic scenarios close to the real traffic environment with its attendant risks and uncertainties.

Various complex constituent systems are combined together to establish an environment of networked driving simulation. This leads particularly to an increased overall system complexity. System modeling plays an important role, especially, with increased system complexity [GCW+15]. The consistent virtualization of the development process is crucial to master the complexity of technical systems. Constructing a multidisciplinary system model is an approach to address system consistency [GCW+15]. Domain-specific models are usually created and used independently of each other, i.e., a horizontal consistency is not obtained. A key benefit of creating multidisciplinary system models is to ensure vertical and horizontal consistency in development. That is, it is convenient to trace system requirements despite the complexity of the networked constituent systems [GCW+15]. A multidisciplinary expertise must be involved while building system models and during system realization. The field of SoS is relatively a new research discipline [Jam08b, p.192]. The current modeling techniques for SoS are still in their infancy. Fortunately, MbSE provides a rigorous foundation for the modeling and conceptual design of SoS [KRU+15]. In this regard, a system model is created and used as a baseline that includes the requirements, analysis, design, and verification of a target system [Mic14]. This system model represents a link between various disciplines, such as electrical, mechanical, software, communication, and requirements engineering [GCW+15]. The system model is not specific to one particular discipline; it provides a comprehensive description for the real system as a whole. However, a design method and a complementary software tool are required to establish different system models or configurations [GCW+15].

According to the thorough literature review, a rigorous methodological basis and tool for networked driving simulation are absent to date. This work presents *A System-Level Design Framework for Networked Driving Simulation*. Specifically, it is a systems engineering framework for designing system models of platforms for networked driving simulation. The key principles of SoSE for an open system architecture, as well as the MbSE measure for building multidisciplinary system models are considered particularly. The design framework builds on and extends methodological work in the literature that considered the topic of driving simulation. The design framework consists of the following key components:

- **Procedure model:** It defines the design phases and the corresponding required tasks systematically in order to develop application-oriented platforms of networked driving simulation. The procedure model considers the different involved disciplines to mitigate the complexity of system development. The procedure model relies on methods and algorithms to make decisions about constituent system composition.
- **Supportive software tool:** It embeds the design phases specified in the procedure model. The software tool supports non-expert users to create multidisciplinary system models according to the requirements of the concerned applications. Platforms for networked driving simulation can be built in accordance with these system models to deliver the expected results.

The following section defines the requirements of the procedure model, the supportive software tool, as well as the resulting platforms of networked driving simulation.

2.6 Design Framework Requirements

Based on the comprehensive problem analysis, the essential requirements of the *System-Level Design Framework for Networked Driving Simulation* have been derived. The following subsections present the requirements of the procedure model, the system of systems configuration software, and the networked driving simulation as a system of systems.

2.6.1 Requirements for the Procedure Model

The procedure model represents the core of the design framework. It defines the required phases in sequence in order to create application-oriented system models for networked driving simulation. Each phase within the procedure model includes a set of tasks. These particular tasks must be carried out in order to achieve the objectives of the respective phases. The derived requirements of the procedure model are as follows.

R1 – Systematic approach: The procedure model shall support users to generate system models, and hence, build platforms of networked driving simulation for their particular application scenarios. Therefore, the procedure model shall present the develop-

ment phases in a systematic manner. Concrete development phases shall be defined. The set of tasks within each development phase shall be specified precisely. The results of each development phase shall be described.

R2 – Complexity mitigation: Networked driving simulation is a typical system of systems (SoS) with acknowledged complexity. The procedure model shall reduce the system design complexity for users. Thereby, the procedure model shall take into consideration that users of networked driving simulation may have no sufficient expertise in driving simulation technologies in general or networked systems in particular. However, users still shall be able to generate application-oriented system models for networked driving simulation.

R3 – Domain-spanning: Networked driving simulation is a typical multidisciplinary system. A multidisciplinary expertise is involved for building system models and during system realization. This expertise encompasses various disciplines, such as electrical, mechanical, software, communication, and requirements engineering. Therefore, the procedure model shall address these different domains under a common system-level design procedure. Users of particular disciplines still shall be able to understand and use the procedure model.

R4 – Automation potential: Some tasks of the procedure model may use particular functions or algorithms. In these cases, the selection of these functions and algorithms shall be based mainly on their potential for automation. That is, the functions and algorithms shall be executed automatically with the help of a software tool.

2.6.2 Requirements for the SoS Configuration Software

An aiding software tool is necessary to enable users of networked driving simulation to generate application-oriented system models. In this work, this tool is called system of systems (SoS) configuration software. It shall fulfill the following essential design requirements.

R5 – Separation of concerns: The SoS configuration software shall not overwhelm users with the system modeling functions or design algorithms. Users shall be able to handle only an easy-to-use graphical interface to generate system models. To that end, the development of the SoS configuration software shall follow the separation of concerns design principle. While the graphical user interface is presented as an upper layer, the underlying complex algorithms and data storage shall be considered in other hidden layers. The eventual future modification of any particular layer shall be independent of the other layers.

R6 – Extendable architecture: The SoS configuration software shall maintain a good extent of extensibility. The possibility to modify existing or eventually to add future functionalities shall be taken into consideration. Moreover, the associated components

database shall be extendable. Entries of available system components within the database shall be editable.

2.6.3 Requirements for Networked Driving Simulation Systems

The main objective of the procedure model and the accompanying SoS configuration software is to design system models for networked driving simulation. These system models consider the requirements of the concerned application scenarios. As a logical subsequent step, corresponding platforms for networked driving simulation are built. General detailed requirements and design recommendations of the next generation of networked simulation environments are discussed in Reference [SBH+13]. However, the designed system models should lead to platforms of networked driving simulation that fulfill the following particular requirements. These requirements are motivated by the principles of the open system approach of the system of systems engineering discussed in Section 2.4.

R7 – Open interface principle: The developed platforms of networked driving simulation shall conform to the open interface principle. This results in open systems that have permeable boundaries. Consequently, the exchange of information with constituent systems eventually added in future shall be allowed through a straightforward integration process. This requirement can be realized, for example, by the utilization of standard interfacing or networking techniques.

R8 – Reconfiguration principle: The developed platforms of networked driving simulation shall have a good extent of reconfigurability to sustain themselves against eventual changes in requirements. This feature can be realized, for instance, through the utilization of reconfigurable constituent systems.

R9 – Modularity principle: The developed platforms of networked driving simulation shall have a good extent of modularity. This means that each constituent system shall perform a major specific task or function independently. Constituent systems performing more than one task or function are not preferred due to associated complexity of adaptation. The modularity principle enables greater evolution flexibility for the developed system of systems.

The following chapter presents and analyzes the state of the art. Related approaches for the topic of driving simulation are discussed. Remarkable utilizations of networked driving simulation reported in literature are presented. Moreover, a selection of outstanding facilities of networked driving simulation is analyzed with respect to the technical specifications and application scopes.

3 State of the Art

There has been an increasing demand for efficient analysis methods for traffic engineering and advanced vehicle systems that show dynamic interactions between road users. Several studies have made use of networked driving simulation to address diverse research questions in this regard. This chapter presents the state of the art with respect to existing driving simulation approaches in general and some remarkable utilizations and facilities for networked driving simulation in particular.

Section 3.1 addresses two outstanding approaches from the literature for the topic of driving simulation. These specific approaches are coupled and used afterwards in the presented work. To highlight its widespread practice, Section 3.2 presents some of the reviewed literature that utilized networked driving simulation for different studies and investigations. Section 3.3 presents three compelling facilities for networked driving simulation. The call for action is derived in Section 3.4 according the state of the art analysis and evaluation. Finally, Section 3.5 introduces an initial solution approach to address the topic of networked driving simulation in a systematic manner.

3.1 Driving Simulation Methods

There is a broad spectrum of driving simulator variants that can be used as supportive tools in the automotive world. This ranges from game-oriented to high-end driving simulators, which vary mainly in their cost and structural complexity. Manufacturers of driving simulators provide different levels of fidelity as an endeavor to fulfill the requirements of different application scenarios. A simple classification of driving simulators into three categories based on fidelity is presented in [Jam11]: low-level, mid-level, and high-level. A low-level driving simulator usually has a driving seat fixed to the ground. A simple force feedback is provided to the driver through a game-style steering wheel. The visual scene is displayed on a single monitor that provides a narrow field of view in front of the driver. A mid-level driving simulator typically incorporates a simple vehicle cabin. Vehicle vibration and road roughness are replicated by actuators located under the driver's seat. Suitable kinesthetic feedback and control cues are provided by a real steering wheel and pedals set. The visual scene is provided through projectors displaying the images on a cylindrical surface. A high-level driving simulator commonly includes a complete vehicle. Vehicle movement is simulated by a complex motion system of at least six degrees of freedom. A real dashboard with all typical instruments provides the necessary kinesthetic feedback and control cues. The visual scene is displayed by projectors providing a surround field of view.

Low-level driving simulators may not provide the immersion necessary for drivers to be fully involved in the simulation. High-level driving simulators may present challenges to overcome the distraction of drivers and to reduce the learning time. Therefore, the selection of driving simulators should properly consider the purpose of use in particular

[GB11]. Three generic application scenarios for driving simulation are defined in [Jam11]: driver behavioral research, vehicle design and engineering, and driver training. In addition, a rough correlation between these application scenarios and the aforementioned classification of driving simulators is provided. Nonetheless, driving simulators are composed of many building components. For an effective utilization of resources and costs, combining low-fidelity with high-fidelity components in one driving simulator can be considered [CJ11]. That is, a driving simulator may have high capability for one particular component and low capability for other components according to the purpose of use. However, it is challenging for non-expert users to select individual simulator components that fit particular application scenarios. The following subsection presents guidelines from the literature to mitigate this problem.

3.1.1 Determining Fidelity Levels According to NEGELE

While purchasing driving simulators, users usually undergo a selection process based on their own understanding of the capabilities of available solutions. The selection process tends to use the available budget to purchase simulator components with the highest possible fidelity level. This results typically in rough selections, where the end benefits are not as great as the purchase and operation costs. Guidelines for determining the fidelity level for each primary simulator component required for a specific purpose of use were introduced by NEGELE [Neg07]. Hereafter in this work, these guidelines are referred to as "NEGELE'S guidelines" according to the author's name.

On the one hand, NEGELE'S guidelines considered the human behavior that generally falls into one of three distinct categories: skill-based, rule-based, and knowledge-based behavior [Cac04]. In particular, the driving behavior is affected correspondingly by the skills, experiences, and situation familiarity of drivers [WSS15]. Skill-based responses occur in very familiar and routine driving situations that require fast actions. In a driving simulator, these responses are triggered automatically only if the sensory stimuli are realistic enough for the driver. That is, high fidelity levels are required for this type of responses. Rule-based responses are invoked in driving situations that require identification and recall of previously instructed actions. The driver is fully aware of the situations and the corresponding necessary rules. In these situations, the responses are triggered moderately and the driver has some time to compensate missing cues. Therefore, a modest deviation from reality in a driving simulation environment is permitted for this type of responses. Knowledge-based responses emerge in unfamiliar driving situations that require effort and conscious attention. The driver exerts much intellectual effort to find out an appropriate response for the situation. These responses occur slowly, so that the driver has enough time to mentally compensate necessary cues. Therefore, a large deviation from reality in a driving simulation environment is allowed for this type of responses.

On the other hand, NEGELE'S guidelines differentiated between three groups of driving tasks: primary, secondary, and tertiary. The primary tasks are further subdivided into: stabilization, guidance, and navigation. Maintaining vehicle state while driving through a curve, interacting with other traffic participants, and planning an entire driving route are examples for the three types of primary tasks respectively. While handling a driver assistance system is an example for the secondary driving tasks, adjusting the air conditioner and tuning the radio are typical tertiary driving tasks. As a matter of course, realistic simulation cues for appropriate vehicle control are more significant for the primary than the tertiary driving tasks. The secondary driving tasks are considered intermediate with respect to the fidelity level required to control the vehicle. Figure 3-1 shows a matrix between the defined driving tasks and driver response types. The mutual intersections result in 15 classes of driving simulator applications. Users have to specify the concerned driving task and response in order to determine the relevant application class.

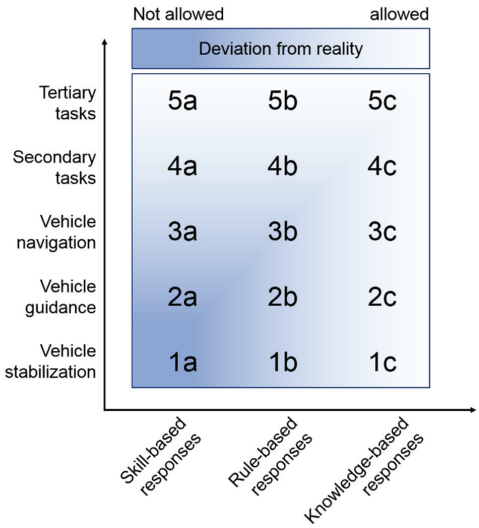


Figure 3-1: Scheme for classifying driving simulator applications – reconstructed according to Reference [Neg07, p. 94]

In addition, NEGELE'S guidelines defined the main driving simulator subsystems and ordered them roughly according to their contribution to the overall simulation fidelity for each application class: visual simulation, motion simulation, driver's platform, acoustic simulation, and objects database along with traffic simulation. Each subsystem has a group of features characterized by different fidelity levels. These are distinguished by letters and ordered by numbers. Table 2 shows the features of the driver's platform along with their different fidelity levels as an example – reconstructed according to Reference [Neg07, p. 56]. The tables of the remaining driving simulator components are provided in the Appendix.

Table 2: Features and fidelity levels of the driver's platform [Neg07, p. 56]

Features and fidelity levels		
Feature	**Key**	**Fidelity levels**
Mock-up	S1	Driving seat and HMI without chassis
	S2	Partial vehicle (quarter or half vehicle)
	S3	Complete vehicle – no modifications apparent
	S4	Series production vehicle – no modifications apparent
HMI	T1	Basic and simple HMI
	T2	Complete and realistic HMI
	T3	Complete HMI with reconfigurable display
Steering	U1	Steering moment proportional to steering angle
	U2	Electrical steering moment, damping, and friction
	U3	Electrical steering moment with high frequency
Pedals set	V1	Force feedback passive
	V2	Force feedback adaptive – characteristic curve modifiable
	V3	Force feedback active – effects of control systems tangible

According to the application class of interest, users can determine the allowed fidelity deviation from reality for their driving simulator. However, it still may be challenging for non-expert users to select particular fidelity levels for the features of each subsystem. Therefore, NEGELE'S guidelines provided examples for common application classes as a means of orientation. Moreover, reasonable feature fidelity levels for each of these classes were deduced and presented in form of profile tables. Table 3 shows the profile of application class 1a – reconstructed according to Reference [Neg07, p. 102].

Table 3: Profile of driving simulator application class 1a [Neg07, p.102]

Skill-based response and vehicle stabilization task		
Visual simulation		
Viewing distance A1	Field of view B2	Stereo vision --
Head tracking --	Rear-view mirrors E2	Field continuity F3
Resolution G2	Frame rate H1	Projector type J2
Motion simulation		
Motion platform < 6 DOF K1	Standard Platform = 6 DOF --	Standard platform > 6 DOF M1/M2/M3
Vehicle dynamics N3	Tire O4	
Driver's platform		
Mock-up S3	HMI T2	Steering U3
Pedals set V2		
Acoustic simulation		
Primary sound P1 (P2)	Auxiliary sound Q1(Q2)	Sound system R1, R3
Environment database		
Database type W1	District type Y1	
Traffic objects simulation		
General traffic vehicles Z1	Special objects --	

For better visualization and easy interpretation of the fidelity levels, the defined simulator application classes are presented in form of specification radar charts. For all simulator subsystems, these charts depict the features and the fidelity levels in comparison to

the maximum achievable fidelity levels. Figure 3-2 shows the specification radar chart of application class 1a as an example.

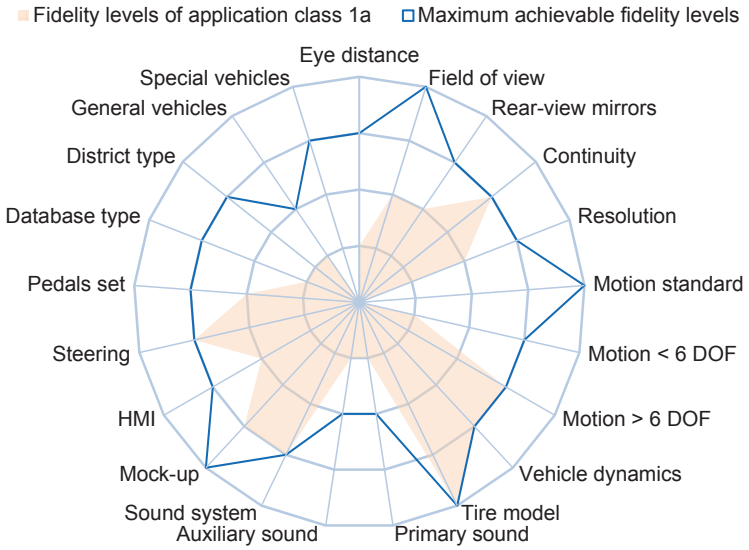


Figure 3-2: Specification radar chart of the simulator application class 1a – reconstructed according to Reference [Neg07, p. 103]

Evaluation

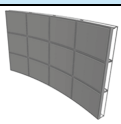
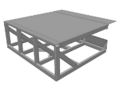
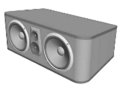
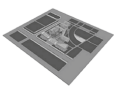
NEGELE'S guidelines assist users to determine the necessary overall fidelity level of a driving simulator. The result is a driving simulator with a complexity level intended for one specific application scenario. However, users may have to alternate between different fidelity levels to address further application scenarios. This process is challenging for non-expert users as it requires technical knowledge of system structure and components compatibility and interoperability. The following section presents a method from the literature to tame this complexity.

3.1.2 Configuring Simulation Environments According to HASSAN

Driving simulation facilities are used in practice to cover possibly diverse application scenarios at the same time. Users may have access to various simulator components of different fidelity levels within the same driving simulation facility. A maintainable and flexible environment for driving simulation is required to easily exchange driving simulator components. A method to reconfigure driving simulation environments by system users is presented in Reference [Has14]. Hereafter in this paper, this method is referred to as "HASSAN's method" according to the author's name.

Principally, HASSAN's method applied a morphological box containing entries of the main driving simulator components together with the available variants [PBF+07]. These variants are called solution elements; they represent products with different fidelity levels and characteristics provided by different simulator manufacturers and developers. In the morphological box, simulator components are listed vertically, while the corresponding available solution elements are listed horizontally. Table 4 shows an excerpt of the morphological box, where the solution elements are represented as symbols. In addition, the morphological box of HASSAN's method is altered in this work, where the naming convention of the driving simulator components of NEGELE'S guidelines is used to maintain consistency.

Table 4: Morphological box for driving simulators components – reconstructed excerpt according to Reference [Has14, p. 54, p. 89]

Driving simulator components	Solution elements		
	1	2	3
Scene simulation system			
Motion simulation system			
Driver's platform			
Acoustic simulation system			
Environment database			
Traffic objects simulator			

System users can select simulator components and solution elements in a process similar to browsing an online catalogue to customize a product before purchasing. The thrust of HASSAN'S method incorporates a consensus check algorithm that has mainly two levels. The first level is the logical dependency check between simulator compo-

nents or subsystems. This dependency check process gives an indication whether the selection of one particular component necessitates or affects the selection of other components. For instance, the selection of the driver's platform (HMI) may depend on the selection of the motion platform and vice versa. A two-dimensional dependency matrix between system components is created for this purpose. System components are listed in the first row and column of the matrix as shown in Table 5. The dependency matrix is mirrored about the diagonal line. The intersection of each pair of different system components determines the logical dependency.

Table 5: Dependency matrix of driving simulator components – reconstructed excerpt according to Reference [Has14, p. 82]

Dependency matrix										
		Hardware				**Software**				
Dependency scheme 0 = Independent components 1 = Dependent components		Visualization system	Motion platform	HMI	Acoustic system	Visualization software	Platform controller	Vehicle dynamics	HMI interface	Acoustic software
Hardware	Visualization system									
	Motion platform	1								
	HMI	0	1							
	Acoustic system	0	0	0						
Software	Visualization software	1	0	0	0					
	Platform controller	0	1	0	0	0				
	Vehicle dynamics	0	0	0	0	0	0			
	HMI interface	0	0	1	0	0	0	0		
	Acoustic software	0	0	0	1	0	0	0	0	

The second level of the consensus check algorithm is the logical consistency analysis. This consistency check process gives an indication whether the selection of one particular solution element is consistent with the selection of other solution elements. For that purpose, a two-dimensional consistency matrix between the solution elements is created. The solution elements of each system component are listed in the first row and first column of the matrix. The consistency matrix is mirrored about the diagonal line. The intersection of each pair of different solution elements determines the logical consistency. Table 6 shows an excerpt of a consistency matrix reconstructed according to HASSAN'S method. The consistency check process makes use of the preceding process of dependency check. If two components are independent, the two corresponding solution elements inherit the independence. In this case, it is not necessary to check the consistency between these particular solution elements.

Table 6: Consistency matrix of driving simulator solution elements – reconstructed excerpt according to Reference [Has14, p. 83]

Consistency matrix											
Consistency scheme 0 = Inconsistent solution elements 1 = Independent solution elements 2 = Consistent solution elements			**Hardware**								
			Visualization system		Motion platform		HMI		Acoustic system		…
			A	B	A	B	A	B	A	B	
Hardware	Visual. system	A									
		B									
	Motion platform	A	2	0							
		B	0	2							
	HMI	A	1	1	2	0					
		B	1	1	0	2					
	Acoustic system	A	1	1	1	1	1	1			
		B	1	1	1	1	1	1			
	…										

The consistency matrix is extendible to account for eventual availability change of solution elements. Both dependency and consistency matrices must be filled out by a system expert. However, HASSAN'S method was embedded within a configuration software. This can be used by non-expert system users to compose different configurations of driving simulators from available solution elements. In addition to the consensus check algorithm, the configuration software checks the compatibility between the selected solution elements. That is, different characteristics of their input and output signals are compared together to assure proper technical operation.

Evaluation

HASSAN'S method provides a procedure to reconfigure driving simulation environments. The focus is given to the configuration process to assure the consensus of simulator components. An accompanying configuration software facilitates the configuration process. No substantial knowledge of simulator components or available solution elements is required from non-expert system users. However, the specifications of driving simulator components are not correlated to the requirements of possible application scenarios. Users still need to manually determine the requirements or the necessary simulator fidelity level for their application scenarios according to some criteria, such as NEGELE'S guidelines. Moreover, users have to manually analyze available simulator components determine their fidelity levels. The effort increases considerably if more than one driving simulator will be utilized in a networked driving simulation environment. The following section gives an extensive review of literature that presents examinations or

experiments using networked driving simulation. The utilized systems are described together with the purposes of use.

3.2 Networked Driving Simulation Utilizations

The topic of driver-driver interaction and multiple-user driving simulation has attracted great attention of researchers. Several studies have made use of networked driving simulation for diverse research questions, which mainly address the interaction between human drivers. For example, a framework for realistic traffic flow simulation was proposed in [CLC10]. The framework includes two types of programmed vehicles. One type of traffic vehicles is operated by basic driver models that simulate the traffic flow in far-field areas. The other type of traffic vehicles is operated by cognitive driver models that simulate near-field areas. The latter type of driver models provides more human-like behavior. Moreover, the simulation framework allows the participation of two driving simulators within the same virtual traffic scenario. These are simple desktop driving simulators with commercial wheel-transmission-pedals sets. The framework is used to study drivers' behavior while traveling with advanced traffic vehicles of human-like behavior. An approach for a multi-driver simulation is described in [MMM+11]. The system is used to study the effects of cooperative ADAS on drivers' subjective feelings, such as, e.g., driving excitement. Up to five persons can drive simultaneously in the same virtual environment. The network of this platform consists of 25 personal computers connected via gigabit Ethernet. The platform uses the commercial software SILAB. A study to examine the differences in driving behaviors during changing lanes is presented in [BS12]. The work used a networked simulation framework based on the SIGVerse simulator environment [TI11]. The cooperative behavior of drivers under different conditions, such as the availability of lanes, was examined in [HBK+12]. The study used the multi-driver simulation environment MoSAIC, where three driving simulators are integrated into the same framework. Another study on a multi-user driving simulation environment is presented in [YBP08]. Four low-cost connected driving simulators were used to test how voice-based command systems, such as GPS-based assistance systems, influence the driving behavior at intersections. Various examinations of collision-likely driving conditions are presented in [HR03]. The studies were performed on drivers, who meet each other in a situation that has a strong potential for a collision. The work used a simulation platform that utilizes two adjacent full-vehicle simulators sharing a common virtual world. Two linked driving simulators were used in [Hou08] to analyze interactions at intersections. While one of the drivers can be a normal participant, the other driver adapts his driving behavior to this participant in order to form a safety-critical situation. A study for anxiety experience while driving with traffic light assistant is presented in [RMM+15]. The work used the multi-driving simulation environment at WIVW GmbH, where four drivers can participate in the same virtual environment. An examination of Intelligent Transportation Systems using driving simulators is presented in [SH12]. The work used multiple-seat driving simulation

platform, in which several human drivers interact in the same virtual environment. The platform consists of three networked training simulators. These are fixed-base simulators with real vehicle mock-ups. A platform composed of two simple fixed-base driving simulators and an integrated traffic simulator has been introduced in [TT15]. The system is used to conduct experiments on drivers' behavior in realistic traffic environments. Another simple environment of multi-driving simulators is presented in [NS13]. The setup consists of several simple PC-based driving simulators, where 3ds Max and Virtools modeling software were used to build the virtual driving scenes [LJX+16]. It is used to conduct experiments on dynamic speed guidance strategies, such as green wave speed and eco-driving speed guidance strategies. A framework for driving behavior studies using multi-user networked 3D virtual environments is provided in [GNM+13]. It incorporates simple driving simulators and uses the Scenario Markup Language (SML) to specify dynamic traffic situations. The framework can be used to examine interactive car-following behavior of drivers.

In addition to the reviewed scientific studies, which used multi-driving simulation environments to address specific research questions, there are several commercial or ready solutions in the market for building networked driving simulation facilities. COCODRIS (Cooperative Competitive Distributed Simulator) from Liophant Simulation is a training station based on real-time cooperative driving simulation [DPV+11]. It has been developed in cooperation with McLeod Institute for Simulation Science International. The software simulates vehicles and handling equipment in networked multi-operator environments. It is used mainly for virtual logistics training and education purposes to reduce the need for real equipment. COCODRIS provides different training environments, such as urban, construction yard, terminals, highways, and country roads. Moreover, different scenarios involving interaction, cooperation, and competition can be experienced, like, e.g., loading/unloading, handling cranes, connecting trailers, and transfer of goods. The COCODRIS software can be installed in different platforms; from laptops and personal computers till simulators with real vehicle mock-ups and motion platforms. SCANeR DT from OKTAL is a software suite with modules specifically designed for drivers training purposes in general and session supervision in a multi-simulator environment in particular. The software contains other modules such as visualization, traffic intelligence, and vehicle dynamics. It can be utilized in sites involved in driver training and driver awareness, such as training centers, transport operators, insurance companies. ST software from ST Simulator Systems is a research simulator platform developed in cooperation with the University of Groningen and the University of Delft. It consists of some software packages and modules. The included functionalities enable researchers to configure a range of different types of driving simulators. With its multi-cabin mode, multiple simulators can be linked into a simulation environment, where all drivers participate and interact with each other. The software can be used for training in areas, such as fire or police departments and other emergency services that involve coordinated actions. PatrolSim from L-3 DPA is a solution for coordinated-response training for law enforcement agencies and emergency vehicle opera-

tors. Different driving simulators can be connected in an interactive and realistic environment for training purposes. The instructor can monitor, record and playback the interactions of users of the networked driving simulators. Websites of the mentioned example commercial tools should be easily discoverable by the given tool name.

Evaluation

Despite promising results delivered by the presented example research studies on the one hand, the utilized tight setups for networked driving simulation remain purpose-specific. Changing system fidelity or capability to address other application scenarios is burdensome and may require expertise from different disciplines. On the other hand, most of the available commercial solutions are closed or protected products; this makes it hard to find much information about the implementation or the underlying concepts. Hence, adapting these software packages to tailor various application requirements can be made only by the vendors. The following section considers a selection of larger and more complex platforms of networked driving simulation. The platforms are analyzed with respect to their technical specifications and purposes of use. Moreover, a brief evaluation is provided as a preparation step before the comprehensive evaluation in the subsequent section.

3.3 Networked Driving Simulation Facilities

Few facilities for networked driving simulation exist around the world in different research and development centers. This section presents three distinguished example facilities for networked driving simulation. The main constituent systems of each facility and the scope of application are described. Each facility is evaluated by elaborating the advantages and drawbacks. These particular systems were selected for analysis as they represent permanent non-commercial solutions that are used since many years for conducting extensive automotive studies. In addition, the selected systems exist at very well-known, competent research centers in three different countries.

3.3.1 Multi-Driver Simulation Lab at DLR

The first example facility is the Multi-driver Simulation Lab operated by the Institute of Transportation Systems of the German Aerospace Center (DLR). Researchers from various fields such as engineering, psychology, and computer sciences develop automotive and railway systems, as well as traffic management strategies at the Institute of Transportation Systems. Using its Multi-Driver Simulation Lab specifically, three human drivers can interact within the same virtual environment to form realistic traffic scenarios [OS15]. The simulation facility was established in 2010; different research projects are conducted to improve future traffic safety and efficiency.

Technical description: The facility for networked driving simulation at DLR includes three similar driving simulators and a workstation as shown in Figure 3-3. It operates

with the DOMINION software architecture of the DLR, the MoSAIC framework, and the VTD simulation environment of VIRES. The DOMINION software architecture is used to facilitate the integration between applications of various development phases. The abbreviation MoSAIC stands for 'Modular and Scalable Application Platform for ITS Components'. It is a flexible architecture for multi-driver simulation [LBJ+11]. The MoSAIC framework allows existing driving simulators to be linked together and operated from a central workstation. VTD stands for Virtual Test Drive; it is modular tool for the development, test, and demonstration of different vehicle technologies, such as advanced driver assistance systems [LVG+10].

Figure 3-3: Multi-Driver Simulation Lab at the Institute of Transportation Systems, German Aerospace Center (DLR) [OS15]

The participating driving simulators are fixed-base systems with driving seats. The visual field is formed by three front monitors that together provide 150 degrees of forward vision. A virtual rear-view mirror is embedded within the middle front monitor. A LCD screen (10-inch) represents the left-side mirror. A digital instrument panel provides the main vehicle states, like, e.g., speed, motor rpm, as well as different states of the simulated driver assistance systems. The driving simulators have active steering wheels that provide haptic feedback up to 38 Nm and active pedals that deliver haptic feedback up to 28 Nm. A camera monitors the test persons and delivers the video data to the central workstation. A total of 26 computers carry out the whole simulation, visualization, monitoring, and analysis tasks.

Application scope: Apart from the technical demands of their development, the introduction of cooperative assistance systems is challenged by the complexity of the human-machine and human-human interactions. Although these systems are designed to reduce the burden on drivers, the complexity of user interface grows with increasing automated and cooperative functionalities. This demands some of driver's attention and introduces more cognitive loads. Researchers working on the multi-driving simulation environment at the Institute of Transportation Systems are concerned mainly with the behavior analysis of human drivers interacting together within the same virtual driving scenario. Specifically, interactions of drivers assisted by different levels of automation are examined using this multi-driving simulation environment. Substantial simulation data are recorded and analyzed. The results are used to conduct design enhancements of the human-machine interface, and hence, increase system acceptance and usability.

Evaluation: Drivers can experience different traffic scenarios using the multi-driving simulation environment at the Institute of Transportation Systems. Moreover, vehicles equipped with different assistance functions can be simulated. However, the utilized driving simulators in this platform have the same fidelity level. Due to the predetermined platform scale and structure, varying the fidelity levels of the building components is troublesome. Moreover, the network connecting the three driving simulators uses Ethernet as a typical communication technology with predetermined network characteristics. No standard architectures for networked simulation are utilized. In addition, eventual future system extensions or modifications according to scientific methods are not envisaged. The platform at the Institute of Transportation Systems has fixed capabilities. Therefore, only a limited set of application scenarios for networked driving simulation may be considered with this platform.

3.3.2 Tokyo Virtual Living Simulation Lab at NII

Tokyo Virtual Living Lab at the National Institute of Informatics (NII) is an experimental space based on 3D Internet technology [PGZ+13]. Researchers of this laboratory conduct controlled driving and travel studies that involve multiple users interacting in the same shared space. Users can interactively experience different innovative transportation technologies. The utilized networked 3D environment is called 3D (social) virtual world or 3D Internet [STN+10]. Using this 3D Internet technology, users are represented as “avatars” (graphical self-representations) and share the same 3D space. They can interact with the environment and other avatars in a natural way. Principally, the virtual living lab can be accessed from anywhere.

Technical description: The laboratory at the National Institute of Informatics (NII) depends mainly on the Distributed Virtual Environments middleware (DiVE) to create multi-user networked 3D virtual spaces [BGR06]. This middleware has a centric architecture that presents a scalable and cross platform communication between different simulators. A core component of the simulation environment is the navigation network.

This includes all information required for the traffic simulation. The navigation network and a corresponding 3D road network are created automatically using the OpenStreetMap (OSM) and CityEngine tools respectively. OSM is an open source tool for creating geographic data based on crowdsourcing [NRJ+13]. CityEngine is a commercial 3D modeling software for generating 3D urban environments [HLH+13]. Moreover, a traffic simulator is used to generate programmed vehicles. These vehicles travel along the road network, where road markings and traffic signals and signs are considered. The traffic simulation is networked with a multi-user driving simulation. More than one driving simulator of low-fidelity levels can be integrated in the same virtual environment as shown in Figure 3-4. These are desktop driving simulators that include commercial wheel-pedals sets to provide low-cost, but reasonable, physical feedback and control cues [GNM+13].

Figure 3-4: Networked driving simulators (left) and a screen shot of the shared 3D environment (right) at the National Institute of Informatics (NII) [PGZ+13], [GNM+13]

Application scope: A comprehensive 3D replica of a part of Tokyo was built in the virtual living laboratory at NII to perform experiments including more than one human driver. The general aim of the developed virtual environment it to analyze interactive driving behavior in future smart cities. For instance, empirical studies with human drivers are conducted to examine the impact of eco-friendly traffic and "normal" (non-eco) traffic on human drivers and how human drivers interact in different cases. For this purpose, the iCO2 software tool was developed with Unity3D based on the DiVE middleware architecture [POC+14]. Programmed vehicles travel continuously through the road network and obey traffic signals and signs. Unexpected traffic situations are created as human drivers interact with each other and with the programmed vehicles. One example scenario task is to drive as far as possible with a given amount of fuel. Drivers must travel in an eco-friendly way by accelerating and decelerating smoothly and keeping the provided speed limits. Non-eco-friendly driving behavior is penalized with more rapid decrease of the fuel supply.

Evaluation: In the Tokyo Virtual Living Simulation Lab at NII, various scenarios can be created with different traffic densities. Principally, geographically distributed partici-

pants can join the shared simulation environment. However, the currently utilized driving simulators have similar fixed low-fidelity level. No particular scientific consideration is given to determine the fidelity level of the participating driving simulators. Moreover, the network connecting the simulators of the laboratory utilizes the Ethernet communication technology that presents a fixed set of network characteristics. A non-standard architecture for networked simulation is utilized. Hence, integrating additional simulators to the environment may be burdensome and requires particular experience with the developed environment. Extending or modifying the whole simulation environment does not follow a particular methodology. Therefore, addressing other application scenarios for networked driving and traffic simulation may be not feasible.

3.3.3 Driving and Bicycling Simulation Lab at OSU

The Driving and Bicycling Simulation Lab at the Civil and Construction Engineering College, Oregon State University (OSU), represents another compelling example facility. In this advanced simulation facility, two human drivers can interact in the same virtual environment. The simulation facility was constructed in collaboration with Realtime Technologies, Inc. in 2010.

Technical description: The first main component in the facility at OSU is the high-fidelity driving simulator shown in Figure 3-5. It is a moving-base simulator with an electric pitch motion system that rotates ±4 degrees about the focal point of the driver's eye [SJI+13]. The motion system simulates the acceleration and deceleration on tangent road segments.

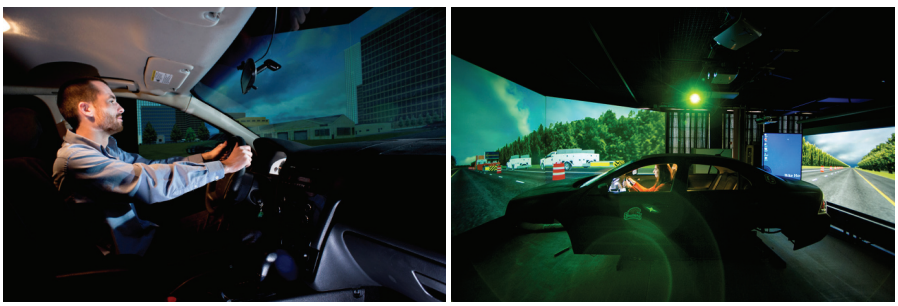


Figure 3-5: Driving simulator at the Oregon State University (OSU) [SJI+13]

The dynamics computer system consists of a quad-core host that runs SimCreator Software from Realtime Technologies, Inc. With this software, vehicle dynamics are simulated by a 15 DOF multi-body vehicle model and a Pacejka tire model. The visualization system includes three front projectors with a resolution of 1400x1050; they provide 180 degrees of forward vision. LCD screens are integrated in both side-view mirrors and a rear projection system provides the scenes for the center rear-view mirror. The visual computer system is comprised of dual-core computers that run Nvidia 280

graphics cards for an update rate of 60 Hz. The driver's platform is a Ford Fusion cabin (model 2009) with a digital configurable instrument panel. A steering control loading system is used to accurately represent steering torques based on vehicle speed and steering angle. The vehicle cabin is mounted on the pitch motion system with the driver's eye-point located at the center of the viewing volume. The acoustic simulation is provided by surround sound speakers of 500 watts. The driver's foot position, face, hands, and the overall driving environment are captured with four separate cameras. These deliver video data to a four-way DVR at a sampling rate of 60 Hz to be used for simulation session assessment. Moreover, eye movement data are collected using the Mobile Eye-XG platform from Applied Science Laboratories. This platform allows drivers to have unconstrained eye and head movement. It has a sampling rate of 30 Hz and an accuracy of 0.5 to 1.0 degree.

The second main component within the facility of OSU is the bicycling simulator shown in Figure 3-6. It is one of just a few full-scale bicycle simulators in the world [HMJ+15]. Moreover, it is the only bicycling simulator that can be operated simultaneously with a driving simulator in the same virtual environment. The braking, pedaling, and steering inputs from the cyclist are reflected in the visual scene. The visualization system includes one front projector and an LCD display representing a rear-view mirror.

Figure 3-6: Bicycling simulator at the Oregon State University (OSU) [HMJ+15]

Application scope: Researchers working in the Driving and Bicycling Simulation Lab at OSU are concerned in general with the safety issues of transportation systems from a multi-modal perspective. Due to the complexity of transportation problems, examinations in this laboratory are interdisciplinary and require expertise from different domains, such as transportation engineering, human factors, psychology, and medicine. In general, this laboratory is an experimental tool to address the behavior of road users. The target of analysis is to improve the operation and safety level of transportation systems. When it comes to driving and biking safety in particular, one of the challenges for civil engineers is predicting when and how crashes occur. There is relatively little substantial data about driver-biker interactions. Hence, the OSU utilizes the driver-biker simulation facility mainly to solve this conundrum [WHM+17]. To that end, the driving and bicycling simulators are networked to share the same virtual environment. Figure 3-

7 shows two virtual scenarios including a vehicle and a bicycle that are controlled by humans.

Figure 3-7: Two different virtual scenarios combining a vehicle and a bicycle controlled by humans (OSU) [SJI+13], [HMJ+15]

Using these networked driving and bicycling simulators, specific crash scenarios can be generated to collect a robust set of data. The data collected from both networked simulators can be used to design safer transportation systems and more effective traffic infrastructure to reduce possible fatal crashes.

Evaluation: The driving simulator incorporated in the simulation laboratory at OSU exhibits a fixed high-fidelity level. For the bicycling simulator, three different bicycles (men, women, and children-size bicycles) can be used to increase the demographic variety of subjects. Various scenarios can be created to invoke different collision-likely situations. However, no actual basis was used to determine the necessary fidelity levels of the building components before establishing the simulation facility. Moreover, the network connecting the simulators utilizes Ethernet as a typical communication technology with a fixed set of network characteristics. No particular scientific methodology is followed for extending or modifying the whole simulation environment. Hence, an adequate consideration of other application scenarios of networked driving simulation may be not feasible.

3.4 Evaluation and Call for Action

The approaches discussed in Section 3.1 represent compelling methodological endeavors for the driving simulation field. NEGELE's guidelines facilitate the determination of the necessary simulator fidelity level. HASSAN's method leads to the flexible configuration of driving simulation environments. All example utilizations of networked driving simulation presented in Section 3.2 depended on closed platforms that are intended for particular research questions. The considered example facilities for networked driving simulation presented in Section 3.3 can deliver good results only with respect to the corresponding specific application scenarios. Moreover, these facilities represent rigorous solutions that unfortunately disregard eventual future system modification. The top-

ic of networked driving simulation necessitates broad design considerations as a typical system of systems (SoS) with acknowledged complexity. To date, no comprehensive methods exist to cover all the requirements previously defined in Section 2.6. Table 7 shows an evaluation overview of the discussed related approaches and the considered example facilities for networked driving simulation according to the previously defined requirements. The evaluation with respect to the individual requirements is elaborated afterwards.

Table 7: State of the art evaluation according to the determined requirements

Evaluation of the analyzed approaches and facilities **Evaluation scale:** ○ = Not fulfilled ◐ = Partially fulfilled ● = Totally fulfilled		**Requirements**								
		Systematic Approach	Complexity Mitigation	Domain-Spanning	Automation Potential	Separation of Concerns	Extendable Architecture	Open Interface Principle	Reconfigurability Principle	Modularity Principle
		R1	R2	R3	R4	R5	R6	R7	R8	R9
Driving Simulation Approaches	Determining Fidelity Levels of Driving Simulators according to NEGELE	◐	◐	◐	○	○	○	○	○	◐
	Configuring Driving Simulation Environments according to HASSAN	◐	◐	◐	◐	●	●	◐	◐	◐
Networked Driving Simulation Facilities	Multi-Driver Simulation Lab at DLR	○	◐	◐	○	○	○	○	○	●
	Tokyo Virtual Living Simulation Lab at NII	○	◐	◐	○	○	○	●	●	●
	Driving and Bicycling Simulation Lab at OSU	○	◐	◐	○	○	○	○	◐	●

R1 – Systematic approach: There are no approaches to date that describe the development of application-oriented networked driving simulation with a systematic manner. NEGELE's guidelines are useful as a preliminary systematic approach for determining the fidelity levels of driving simulators. HASSAN's method presents the necessary systematic steps to reconfigure conventional driving simulation environments. NEGELE's guidelines and HASSAN's method did not consider the systematic development of networked driving simulation as a system of systems that serves a set of extended application scenarios. On the other side, the analyzed facilities of networked driving simulation were not developed according to any particular systematic approaches.

R2 – Complexity mitigation: There are no approaches to date that reduce the design complexity of networked driving simulation for system users. For instance, users have to undergo an exhausting analysis of NEGELE's guidelines to determine the necessary fidelity levels of the constituent systems. HASSAN's method aids system users while reconfiguring a simulation environment. It reduces the reconfiguration complexity with respect to conventional driving simulation. However, both approaches did not consider the participation of various constituent systems that are networked together via a communication system. To use both approaches together, users still have to manually correlate the specifications of available solution elements to the defined component fidelity levels and their possible applications. On the other side, the analyzed facilities of networked driving simulation reduce complexity for developers by using well-defined system structures. However, system users still lack the appropriate support for handling the complex system and eventually addressing new application scenarios.

R3 – Domain-spanning: As a typical system of systems, the design and realization of networked driving simulation involve various disciplines. The analyzed approaches for driving simulation addressed only particular design aspects of the system. On the one hand, NEGELE's guidelines consider the aspect of component fidelity levels. On the other hand, HASSAN's method deals with the reconfigurability aspect of conventional driving simulation environments. These approaches did not consider the multidisciplinary nature exhibited by the networked driving simulation as a system of systems. On the other side, different system aspects have been considered in order to establish the analyzed facilities of networked driving simulation. However, system users still lack a domain-spanning method that enables them to understand and handle the different involved domains.

R4 – Automation potential: As a theoretical analysis, NEGELE's guidelines did not provide aiding practical tools or functions for the determination of fidelity levels of driving simulator components. HASSAN's method utilizes functions that show high potential for automation. These functions were embedded within a configuration software to facilitate the generation of driving simulator variants while disregarding the requirements of the application scenarios. Users still have to manually analyze the application scenarios to find out the necessary fidelity levels of the driving simulator components. After that, users have to manually correlate the determined fidelity levels to the specifications of available solution elements in order to begin with the driving simulator configuration process. On the other side, the analyzed facilities of networked driving simulation were not developed according to any particular approaches that show automation potential.

R5 – Separation of concerns: NEGELE's guidelines were not embedded within a software tool. Hence, this particular requirement was not fulfilled by NEGELE's guidelines. In contrast, HASSAN's method is associated with a configuration software. This software tool follows the separation of concerns design principle. A graphical user interface, con-

figuration functions, and a components database represent the main separate layers of the developed configuration software. On the other side, the analyzed facilities of networked driving simulation do not utilize a software tool for system design and development. Thus, this requirement is not applicable, and hence, not fulfilled within these facilities.

R6 – Extendable architecture: Similar to the previous requirement, this specific requirement is not fulfilled by NEGELE's guidelines as they were not embedded within a software tool. Notwithstanding, the configuration software associated with HASSAN's method exhibits a very good level of extensibility. It is possible to modify the configuration software and add more functionalities. Furthermore, the associated components database is extendable. The entries of available solution elements within this database are editable. On the other side, no software tool for overall system design and development is used within the analyzed facilities for networked driving simulation. Therefore, these facilities do not fulfill this particular requirement.

R7 – Open interface principle: This requirement is not applicable for NEGELE's guidelines as the relation between simulator components were not considered. A general interfacing module was introduced in HASSAN's method to easily connect simulator components together. However, HASSAN's method considered the driving simulator as an isolated system that does not have to exchange information with other systems. On the other side, the Multi-Driver Simulation Lab at DLR and the Driving and Bicycling Simulation Lab at OSU do not follow the open interface principle. They utilize only customized, i.e., not standard, networked simulation software tools. Hence, exchanging simulation data with eventually integrated new constituent systems is not feasible and may require considerable system changes. The Virtual Living Simulation Lab at NII considers the open interface principle. It is convenient to integrate additional components with the system to enhance its capabilities.

R8 – Reconfiguration principle: NEGELE's guidelines did not consider the reconfiguration principle. HASSAN's method addressed the reconfigurability principle, however, with respect to driving simulators only as isolated systems. Apart from changing the simulation scenarios, the Multi-Driver Simulation Lab at DLR does not support system reconfigurability. Moreover, the constituent systems have similar fidelity levels that cannot be changed without considerable effort. Principally, the Virtual Living Simulation Lab at NII does not have particular restrictions on the participating constituent systems. The system follows the reconfigurability principle. The Driving and Bicycling Simulation Lab at OSU provides partial reconfigurability. Different bicycle variants can be integrated within the system. However, the utilized driving simulator has fixed fidelity level that cannot be varied without extensive effort.

R9 – Modularity principle: To a far extent, NEGELE's guidelines and HASSAN's method consider the modularity principle with respect to the driving simulators as isolated systems. However, traffic simulation is considered as part of driving simulation in both

approaches. On the other side, the analyzed facilities for networked driving simulation follow the modularity principle. Each involved constituent system within these facilities performs one particular task or function.

Conclusion: The analyzed approaches for driving simulation do not meet all the determined requirements. They did not consider the topic of networked driving simulation and its design demands as a system of systems. Furthermore, the existing facilities for networked driving simulation do not follow any particular systematic methods for system design and development. The following section presents a reference architecture for networked driving simulation to introduce a first systematic solution approach that can fulfill the determined requirements.

3.5 Solution Approach

The main aim of the presented work is to provide a systematic approach for the design and development of networked driving simulation platforms that address different application scenarios. The literature lacks to date rigorous methodological work that can present a consistent notion of what constitutes a platform of networked driving simulation as a system of systems. Therefore, the concept of reference architecture is adopted in this context as a first endeavor to visualize the main system components and their roles. The reference architecture is used as basis for the specification of the *System-Level Design Framework for Networked Driving Simulation*.

In general, there are different driving forces for the identification and use of reference architectures [CMV+10]. The term “reference architecture” was used by many instances in the literature. It has different meanings according to the diverse contexts [CMV+10]. While some contexts focus on technology and implementation, other contexts are oriented to business goals. Even with respect to the information technology community, the term “reference architecture” has many definitions, various purposes of use, and different abstractions or levels of details. However, the information technology community often uses this term for the infrastructural concepts, such as layering, communication, and persistency [CMV+10].

The concept of reference architecture from a systems engineering perspective is presented in Reference [CMV+10]. One simple and relevant definition for the term “reference architecture” is introduced in Reference [CMV+10] as:

> *“Reference architectures capture the essence of existing architectures, and the vision of future needs and evolution to provide guidance to assist in developing new system architectures.” [CMV+10, p. 17]*

It is a general definition that takes into account most of the driving forces for identifying and using reference architectures. In particular, the related driving motives for the identification of a reference architecture in this work are:

- Providing a guidance for system modularization
- Articulating possible involved domains for system development
- Facilitating decisions about constituent systems exchange or system upgrade

The reference architecture shown in Figure 3-8 was identified according to the aforementioned general definition of reference architectures, the determined motivation, and the acquired experiences in the field of networked driving simulation. It provides a baseline to develop new and reshape already existing systems according to the requirements of diverse application scenarios.

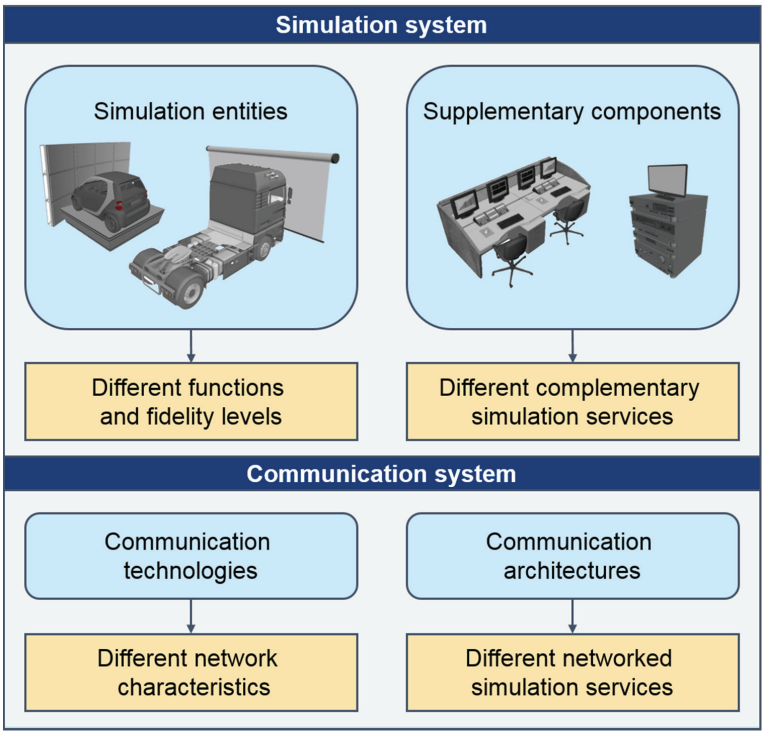


Figure 3-8: Identified reference architecture for networked driving simulation

The identified reference architecture is composed of two main layers:

- **Simulation system layer:** This layer involves mainly systems components responsible for entity simulation, such as driving simulators, traffic and pedestrian simulators. Moreover, other supplementary system components can be included in this layer, such as workstations for simulation control and monitoring, and database consoles for data logging and session analysis and evaluation.

- **Communication system layer:** This layer involves two categories of system components. The first category includes different communication technologies responsible for information exchange, such as Ethernet, CAN, and FlexRay. These communication technologies differ mainly through the provided networking characteristics. The second category includes different communication architectures responsible for networked simulation management, such as Distributed Interactive Simulation (DIS) and High-Level Architecture (HLA) [XSC+11], [WYX+09]. These communication architectures differ mainly through the provided functions and services for networked simulation.

The collective inputs to this reference architecture are the requirements of different application scenarios with respect to the simulation and communication systems. The main output of the reference architecture is represented as system models of concretized solutions that tailor different application scenarios for networked driving simulation.

Platforms of networked driving simulation involve disruptive technologies from different disciplines. Therefore, it is challenging to provide an objective proof for the absolute wholeness of the identified reference architecture. Nonetheless, a satisfying alternative principle for validation is defined in Reference [CMV+10] as:

> *"A reference architecture is based on concepts proven in practice. Most often preceding architectures are mined for these proven concepts. Architecture renovation and innovation validation and proof can be based on reference implementations and prototyping." [CMV+10, p. 20]*

Hence, the system models and the corresponding platforms for networked driving simulation discussed in Chapter 5 present an indirect, yet rigorous, proof for the validity of the identified reference architecture. The know-how represented in the form of an abstract reference architecture provides guidance to later design and development phases. More concrete description of system development will be presented in the next chapter by discussing the *System-Level Design Framework for Networked Driving Simulation*.

4 A System-Level Design Framework for Networked Driving Simulation

The problem analysis presented in Chapter 2 and the state of the art reviewed and evaluated in Chapter 3 proved that the development of networked driving simulation possesses a multidisciplinary nature. Networked driving simulation per se is a typical system of systems (SoS) with acknowledged complexity. Fortunately, model-based systems engineering can provide a rigorous base for the modeling and conceptual design of SoS [KRU+15]. In this regard, a system model is created and used as a foundation that includes the requirements, analysis, and design of a target system [Mic14]. The system model provides a comprehensive description of the real system as a whole. However, this approach imposes a demand for a systematic design method and a complementary software tool to establish different system models tailored for the requirements of various application scenarios [GCW+15]. This chapter represents the essence of the work towards achieving this particular goal. The chapter presents the *System-Level Design Framework for Networked Driving Simulation*. Section 4.1 introduces the overall design framework and its essential components. Section 4.2 provides an overview of the procedure model that encompasses the core methodological work. The necessary development phases, the individual tasks, and the results of each development phase are discussed thoroughly from Section 4.3 to Section 4.7.

4.1 Design Framework Components

The design framework is developed to support non-expert users and domain-specific developers in order to create multidisciplinary system models for platforms of networked driving simulation. These platforms are tailored specifically for the requirements of different application scenarios. To that end, the key principles of SoS engineering for an open system architecture, as well as the crucial measure of model-based systems engineering for building multidisciplinary system models are particularly considered in the design framework. Specifically, the design framework responds to the call for action declared in Section 3.4 subsequent to the comprehensive analysis and evaluation of the state of the art presented in Chapter 3. The design framework consists basically of a procedure model and a SoS configuration software. These primary components are described as follows:

- The **procedure model** includes the necessary development phases in a specific hierarchy towards the design of multidisciplinary system models for platforms of networked driving simulation. Each development phase contains a set of specific tasks, which shall be carried out in order to obtain the phase objectives. To that end, the procedure model specifies the methods and approaches used in each task. In addition, the procedure model explains the results of each phase.

- The **SoS configuration software** is mainly a system-level design software. It embeds the methods and approaches of the procedure model to generate application-oriented system models. The SoS configuration software guides the non-expert system users in a sequential process to achieve the end objective. System users can be operators or domain-specific experts. They do not have to acquire deep multidisciplinary knowledge in order to use the SoS configuration software for the system model design and generation.

Figure 4-1 depicts these primary components together with an overview of the overall design framework. The different components are denoted with the corresponding section numbers of this work.

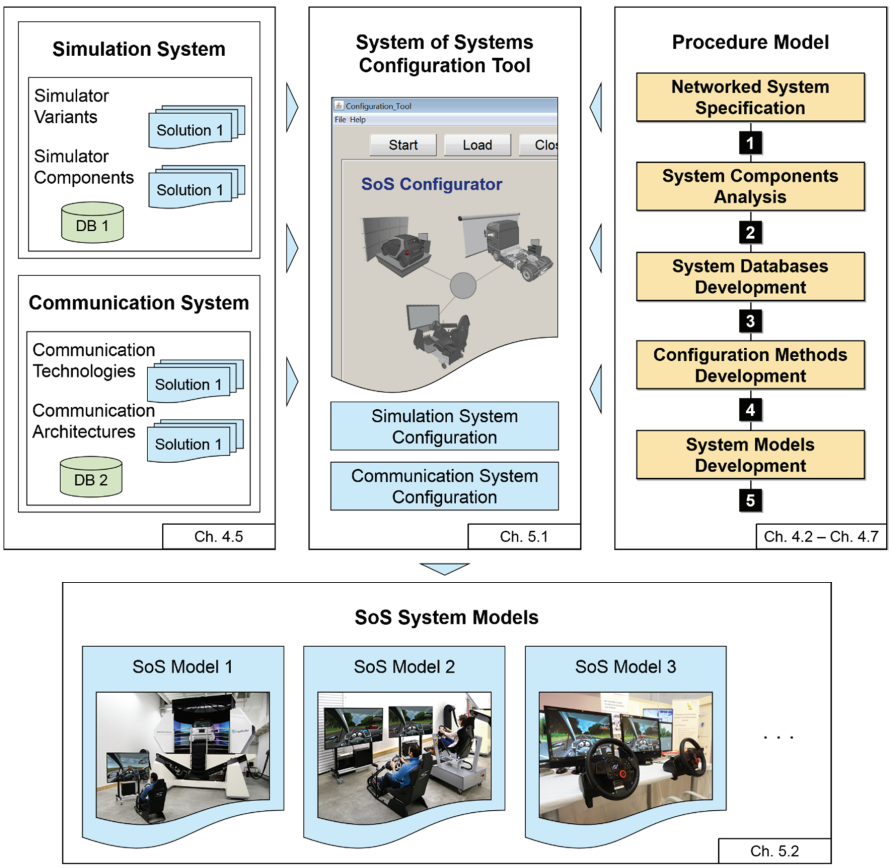


Figure 4-1: A system-level design framework for networked driving simulation

In addition to the primary components, Figure 4-1 shows a system database that represents a supplementary component of the design framework. It is designed within the development phases of the procedure model to be processed by the SoS configuration

software. Figure 4-1 depicts also the results of the design framework in a symbolic form. These are SoS models for platforms of networked driving simulation that vary regarding the structure, complexity, and functionality in accordance with the requirements of the concerned application scenarios. The following section gives an overview of the procedure model and the development phases. More comprehensive discussion of these development phases and the involved tasks is presented in the subsequent section.

4.2 Procedure Model Overview

The procedure model is the thrust of the design framework. It comprises all the theoretical fundamentals used in this work. Specifically, the procedure model consists of five subsequent development phases. Each development phase represents an upper mission that includes further specific tasks. Carrying out these tasks leads to particular partial results that together form the overall outcome of the developed methodology. Figure 4-2 shows the procedure model in the form of consequent milestones.

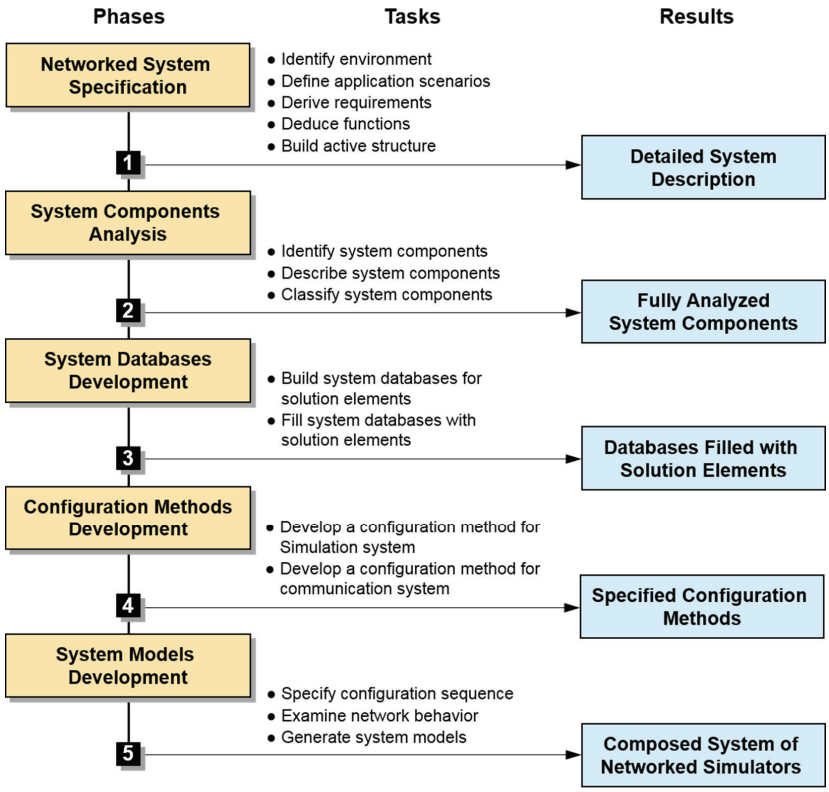


Figure 4-2: A procedure model to develop application-oriented system models for platforms of networked driving simulation

The first development phase within the procedure model is the **networked system specification**. In this phase, networked driving simulation is addressed as a complex system that is principally composed of further complex mechatronic systems. Therefore, CONSENS is utilized in this phase as a well-established specification technique for complex mechatronic systems [ABD+14, p. 119]. The specifications of the networked driving simulation system are determined and analyzed in the form of several coherent partial models. The partial models together form a holistic description for the system of networked driving simulation. The result of this phase represented as a comprehensive system description is used as solid baseline for the subsequent development phases. The second development phase within the procedure model is the **system components analysis**. In this development phase, all possible constituent systems and building components of the networked driving simulation system are identified, described, and classified. Furthermore, the roles of these system components within the entire SoS are described. The results of this phase lead to a comprehensive understanding of the networked driving simulation by highlighting its possible constituent systems and building components together with their categorization. This detailed analysis is necessary for the subsequent development phases towards system concretization. The third development phase within the procedure model is the **system databases development**. In this development phase, system databases are designed in accordance with the identification and classification of the constituent systems and building components in the previous phase. Moreover, entries of available components are deployed in these databases. The results of this development phase lead to a foundation for the selection and decision-making processes in the next development phase. The fourth development phase within the procedure model is the **configuration methods development**. Particular methods are developed in this phase for system configuration. Specifically, the constituent systems and building components can be selected using these methods after determining the necessary complexity grades and capabilities in accordance with the requirements of the concerned application scenarios. The result of this development phase is a set of concretized configuration approaches for the networked driving simulation system. The fifth development phase within the procedure model is the **system models development**. In this development phase, constituent systems and building components are selected based on a particular configuration sequence. Accordingly, a system model is developed and generated. This system model gives an overview of the system and summarizes the characteristics and capabilities of all selected constituent systems and building components. That is, the system model presents a concretized description of the target networked driving simulation system that is tailored specifically for the concerned application scenario. The included constituent systems and building components (i.e. solution elements) fulfill the application requirements to the best possible extent. The eligibility extent of the system is demonstrated transparently to users with the help of comprehensible charts within the system model. The following sections provide a thorough discussion of the whole development phases and the involved tasks together with the corresponding results.

4.3 Phase 1 – Networked System Specification

The objective of this phase is to provide a clear understanding of the networked driving simulation system by building a holistic system description. Principally, this description combines various development aspects of the target system. A multidisciplinary expertise must be involved during the design and the realization of platforms for networked driving simulation as a complex SoS. However, the available system architecting and description techniques for SoS to date are still in their infancy [Jam08a]. Moreover, these techniques usually focus on specific aspects of the SoS. For instance, some architecting techniques concentrate on the synergy of the constituent systems. Others focus on the communication between the constituent systems, with the argument that this particular aspect is common for all SoS types. However, a broader consideration is necessary for the successful design of complex systems [ABD+14, p. 117], [GGT13]. Therefore, the utilization of a well-established domain-spanning conceptual design method is necessary in this work for the specification of networked driving simulation systems. The CONSENS specification technique is adopted in this phase [GFD+09]. The term "CONSENS" is an English acronym standing for Conceptual Design Specification Technique for the Engineering of Complex Systems. In general, this specification technique was discussed thoroughly in Section 2.4.2 to show its potential for the design of complex mechatronic systems. It mitigates the design complexity by describing the various aspects of multidisciplinary systems using a set of coherent partial models. In particular, the effective usability of the CONSENS specification technique for the domain of driving simulators was validated in Reference [Has14]. Furthermore, the essential CONSENS partial models in this regard were determined and structured in a specific workflow. However, since the CONSENS specification technique is open for the conceptual design of newly emerging complex systems, it undergoes some minor modifications in this work for the development of networked driving simulation systems. The following subsection discusses the CONSENS workflow for the specification of networked driving simulation systems.

4.3.1 CONSENS Workflow for Networked Driving Simulation

The outcome of the CONSENS specification technique is represented as a principle solution that is described by eight interrelated partial models. Specifically, these partial models are: environment, application scenarios, requirements, functions, active structure, system of objectives, shape, and behavior [ABD+14, p. 119], [GFD+09]. Each partial model describes a specific aspect of the target system [GFD+09]. To build a coherent system of systems model in this work, the focus is given to the first five partial models in particular [VG12, p. 2]. The relevant partial models in this work are: environment, application scenarios, requirements, functions, and active structure. The system of objectives, shape, and behavior partial models are not considered within the scope of this work. Figure 4-3 shows the workflow of these partial models along with their summarized outcomes.

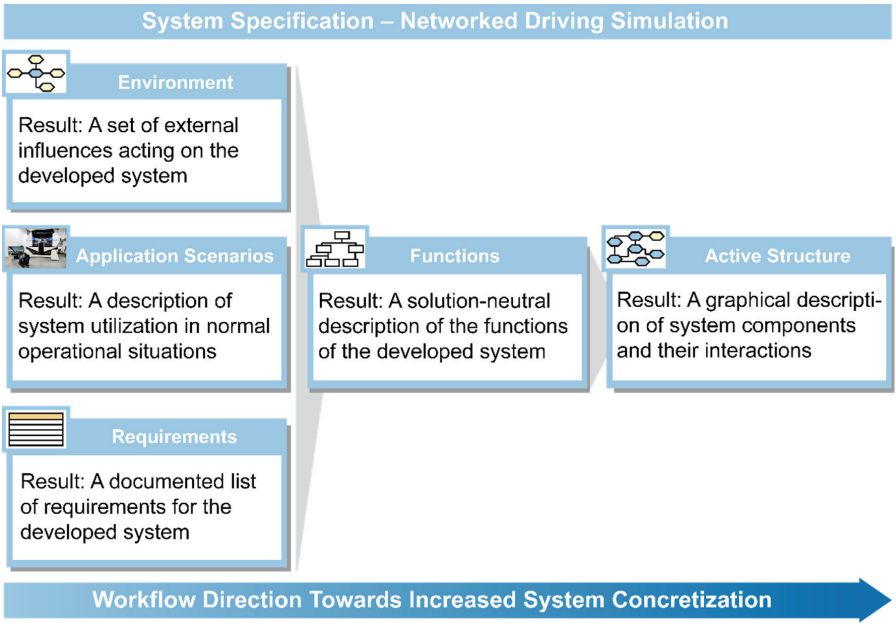


Figure 4-3: CONSENS workflow for networked driving simulation development

In this work, the CONSENS workflow is divided into three steps towards an increased system concretization as shown in Figure 4-3. The first step includes three partial models: environment, application scenarios, and requirements. The second step depends on the outcomes of the first step to create a hierarchy of functions for the entire system. In the third step, an active structure is built based on the results of the previous steps. The system specification process is often carried out during expert workshops. Experts of various disciplines, such as mechanical engineering, electrical engineering, communication engineering, requirements engineering, collaborate to specify the different aspects of the target system. The following subsections present these aspects and their results with respect to the networked driving simulation system.

4.3.2 Identify Environment

The environment partial model defines the possible external influences that can affect the networked driving simulation system. These external influences are environment elements or disturbance variables [GFD+09]. The networked driving simulation system is considered as a black box within the environment partial model. That is, the internal structure and the constituent systems are not visible in this partial model. In this work, the environment partial model is determined based on the acquired experiences with networked driving simulation and the analysis of typical driving simulation facilities. Specifically, this basis results in the identification of five main environment elements. The following is a description of the five identified environment elements:

- **Drivers:** The human drivers represent a crucial environment element. These drivers use the input devices within the driving platforms of the participating driving simulators to control a simulated vehicle in a virtual environment. The main input signals are: acceleration pedal position, brake pedal position, gear selector position, and steering wheel angle. The drivers receive feedback from their respective driving simulators in the form of motion or vibration, as well as visual and acoustic information. Basically, the motions and vibrations are generated by the eventually utilized motion platforms. In addition, some input devices, such as active steering wheels, can deliver relative motions for drivers. The visual feedback is represented with virtual scenes displayed to the drivers via the visualization systems. The acoustic feedback is delivered by the acoustic systems as sound effects that accompany the 3D models. The visual and acoustic signals are generated often together with the visualization software.
- **Simulation operator:** This can be a technician or a laboratory engineer, who is responsible for the general operation of the facility of networked driving simulation. Eventually, the simulation operator can be a domain-specific engineer or developer, who wants to conduct some experiments using the facility of networked driving simulation. The simulation operator can control the scenario by setting some simulation parameters. The networked driving simulation system returns simulation signals to the simulation operator for monitoring purpose.
- **Energy source:** This can be a wall outlet that provides electrical energy to the constituent systems and building components of the networked driving simulation system. Eventually, some components may require power supplies to convert the electrical power of the wall outlet to the levels suitable for their circuitry.
- **Ground:** This is the physical base of the networked driving simulation system. Dynamic forces exist between the ground and the networked driving simulation system as actions and reactions, specially, when the participating driving simulators are equipped with motion platforms.
- **Environment:** The surrounding environment affects the networked driving simulation system through disturbing influences, such as humidity, dirt, light, and temperature. The networked driving simulation system affects the surrounding environment through the produced heat and operation noise. Basic considerations regarding the influences of the environmental conditions on system reliability are presented in Reference [VDI4005].

Figure 4-4 shows a typical example environment partial model of a networked driving simulation system. The environment elements are illustrated as yellow hexagons, while the networked driving simulation system is represented as a blue hexagon in the center of the model.

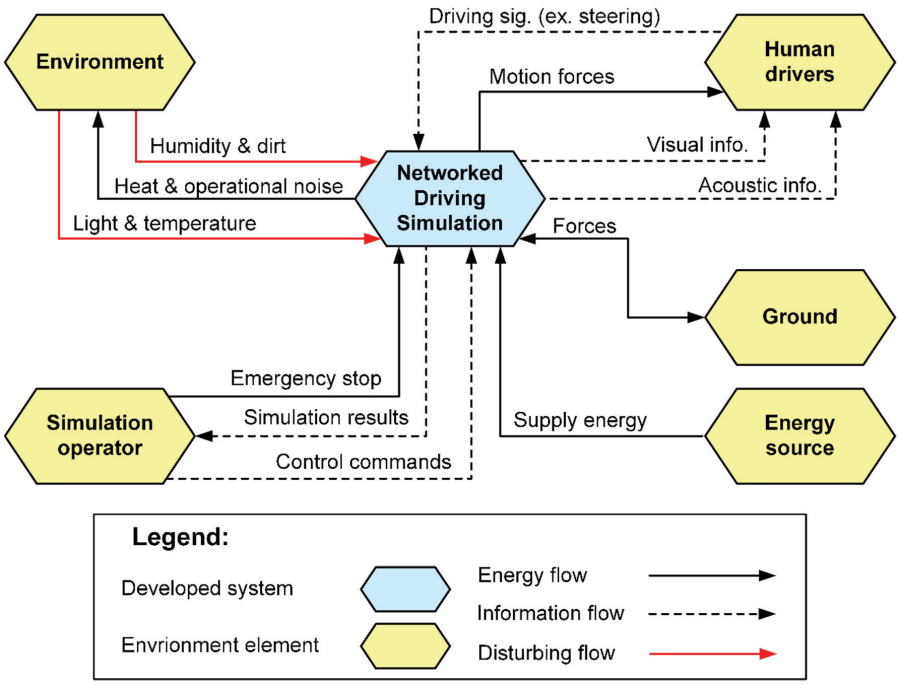


Figure 4-4: Environment partial model of a networked driving simulation system

The interrelations between the networked driving simulation system and the main environment components are shown in Figure 4-4. These interrelations are categorized mainly as information, energy, and disturbing flows. The objective of the established environment partial model is to ensure that all system surroundings are considered in a very early development phase. Hence, external causes of eventual future malfunctions can be easily determined and mitigated. The following subsection presents the application scenario partial model and its results with respect to the networked driving simulation system.

4.3.3 Define Application Scenarios

This partial model specifies the potential application scenarios of the networked driving simulation system. Each application scenario describes the target system with respect to the aim of use, operation modes, and the primary constituent systems and building components utilized for this particular application scenario. Specifically, the application scenarios are modeled using the so-called profile pages. Each profile page contains characterizing information about a particular scenario, such as the title, ID, and modification date. Moreover, each profile page provides a textual description of the application scenario [ABD+14, 120], [GFD+09]. Eventually, a sketch or a schematic can be added to provide better understanding of the application scenario. However, this sketch

shall not be confused with the shape partial model that provides different perspectives of the system and more details, such as the dimensions. Table 8 shows the profile page[1] of an example application scenario of a networked driving simulation system.

Table 8: Example application scenario of a networked driving simulation system

<table>
<tr><td colspan="2">Application scenario
Multi-driver training in driving schools</td><td>Status
10/19/2017</td><td>AS 1</td><td>Page 1</td></tr>
<tr><td colspan="5">Description: A driving instructor at a modern driving school handles two trainees simultaneously in realistic, multi-interactive traffic scenarios. The trainees share a virtual traffic environment. They have to react to each other and adapt their driving behavior interactively.

Simulation system
<table>
<tr><td colspan="2">Simulation entities</td><td>Supplementary components</td></tr>
<tr><td>Driving simulator 1</td><td>Driving simulator 2</td><td>Workstation</td></tr>
<tr><td>Purpose of use:
Trainee 1 uses this driving simulator to learn how to handle traffic situations in a virtual multi-interactive environment.</td><td>Purpose of use:
Trainee 2 uses this driving simulator to learn how to handle traffic situations in a virtual multi-interactive environment.</td><td>Purpose of use:
The driving instructor uses the workstation to control and monitor the training session.</td></tr>
</table>
Communication system
<table>
<tr><td>Communication technology</td><td>Communication architecture</td></tr>
<tr><td>General description:
A feasible communication technology is utilized. It shall ensure the data exchange with little delay and loss rates.</td><td>General description:
To maintain the solution feasibility for driving schools, no communication architecture is utilized in this application scenario.</td></tr>
</table>
Sketch
  </td></tr>
</table>

[1] The structure of the profile pages of the application scenarios is modified in this work. It differs from the standard structure of the CONSENS profile pages to become eligible for the specification of networked driving simulation as a system of independent and heterogeneous systems.

As a system of independent and heterogeneous systems, the description of the application scenarios of networked driving simulation is modified in this work. An overall application scenario is described for the whole networked driving simulation system. This description highlights principally the ultimate goal of the developed system of systems. In addition, a purpose of use for each anticipated constituent simulation system is described. Two or more driving simulators and eventually a traffic simulator represent a typical set of constituent simulation systems. Principally, the defined purposes of use can be used as basis to determine and document a set of requirements for each anticipated constituent system separately. The following subsection presents the requirements partial model and its results with respect to networked driving simulation systems.

4.3.4 Derive Requirements

This partial model specifies the list of requirements of the networked driving simulation system. Principally, this list can include functional and non-functional requirements [ABD+14, p. 121], [GFD+09]. The individual items of requirements can be denoted as demands or wishes (D/W) [PBF+07]. Table 9 shows an excerpt of an example list of requirements[2] of a networked driving simulation system.

Table 9: List of requirements of a networked driving simulation system (excerpt)

ID	No.	Requirements for networked driving simulation system	D/W
1	Requirements for driving simulator 1		
	1	Scene simulation system	
	1.1	It shall cover a 120° horizontal field of view	D
	…	…	
	2	Motion simulation system	
	2.1	It shall provide three degrees of freedom	D
	…	…	
2	Requirements for driving simulator 2		
	1	Scene simulation system	
	…	…	

[2] The structure of the list of requirements is modified in this work. It differs from the standard structure of the CONSENS list of requirements to become eligible for the specification of networked driving simulation as a system of independent systems.

As a system of independent systems, the structure of list of requirements of networked driving simulation has a modified form in this work. A separate set of requirements is defined for each independent constituent system and component that is denoted by a unique ID. The different sets of requirements form together the overall requirements of the networked driving simulation system. The following subsection presents the functions partial model and its results with respect to networked driving simulation systems.

4.3.5 Deduce Functions

The individual functions within the networked driving simulation system are deduced based on system requirements and application scenarios. Interactive simulation, traffic simulation, operation management, and data collection, and network communication are the fundamental system functions according to the problem analysis and the acquired experiences in the field of networked driving simulation. Each of these defined functions may undergo further top-down hierarchical subdivisions [ABD+14, p. 123], [GFD+09]. Discussions and catalogues to support the composition of hierarchical function subdivisions are presented in References [Bir80] and [Lan00]. Figure 4-5 shows example functions and sub-functions of a networked driving simulation system.

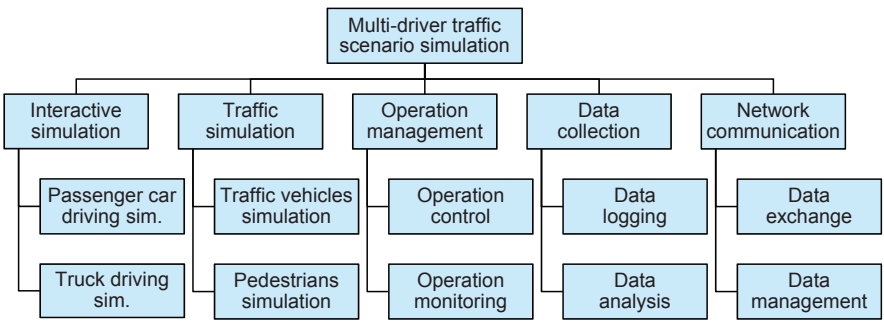


Figure 4-5: Functions partial model of a networked driving simulation system

The defined system functions are realized by solution patterns and concretizations. For instance, the interactive simulation function can be carried out by two specific driving simulators of different or equal complexity grades. The simulation of traffic vehicles and pedestrians can be performed with an independent traffic simulator. A workstation can provide a capability for operation control and monitoring. A database console can carry out the data logging function and serve for subsequent simulation session analysis. A communication technology, for instance, such as Ethernet, can carry out the data exchange between the constituent systems and building components. Data management can be achieved by communication architecture for networked simulation, such as High-level architecture [WYX+09]. As a lot of solutions may be available, a classification scheme (morphological box) can be utilized to facilitate the systematic combination of available solutions [PBF+07]. According to the adopted definition for systems of sys-

tems, networked driving simulation systems are composed of further heterogeneous constituent systems and building components. Therefore, combining only compatible solutions does not apply in this context in contrast to the design of typical mechatronic systems [PBF+07]. The following subsection presents the active structure partial model and its results with respect to networked driving simulation systems.

4.3.6 Build Active Structure

The active structure partial model is created based on the defined system functions and the possible constituent systems and building components of the networked driving simulation. In contrast to the environment partial model that considers the whole system as a black box, the active structure partial model concretizes the system by illustrating its internal structure in more details [ABD+14, p. 124]. Specifically, it shows the main system components and their primary interrelationships in the form of information and energy flows [GFD+09]. Figure 4-6 shows an example active structure of a networked driving simulation system including all possible (yet not all necessary) constituent systems and building components.

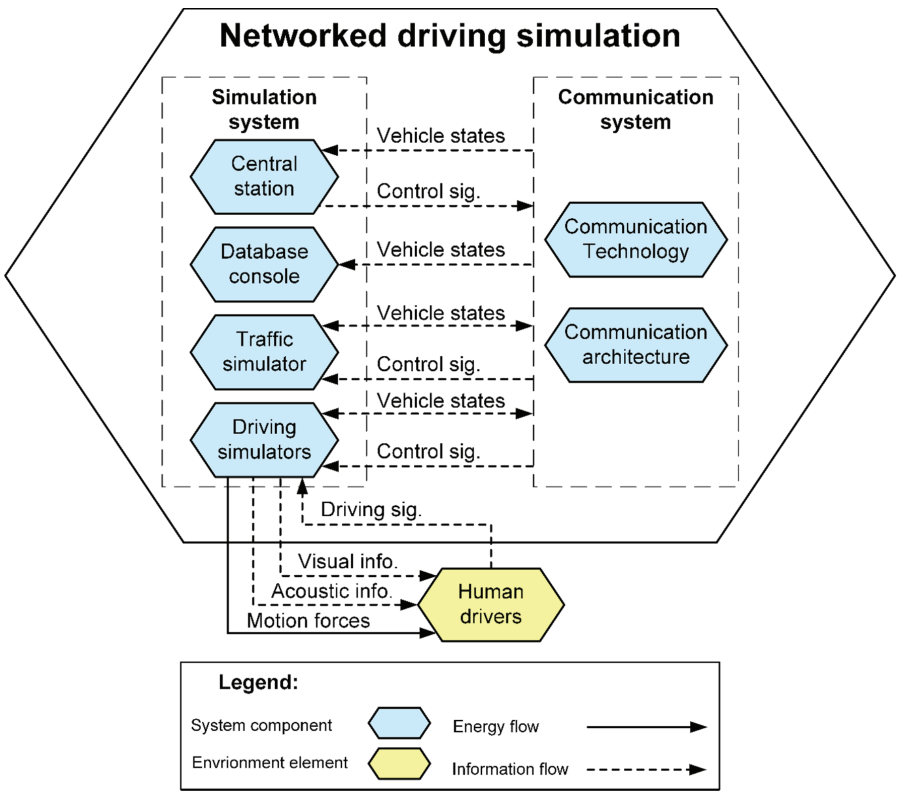
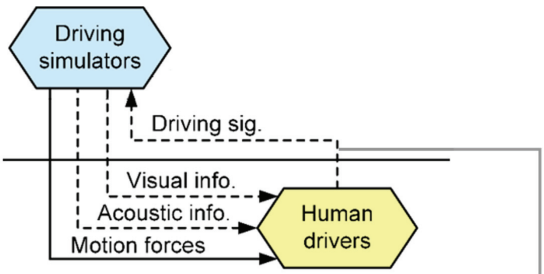


Figure 4-6: Active structure partial model of a networked driving simulation system

The constituent systems and building components within the active structure shown in Figure 4-6 are identified based on the particular system functions defined in the previous subsection. Specifically, the presented active structure partial model includes six constituent systems and building components that belong to two different member groups: **simulation system group** and **communication system group**. On the one hand, the driving simulators and the traffic simulator are constituent systems that belong to the simulation system group. Similarly, the workstation and database station are considered as supplementary system components that belong to the simulation system group. On the other hand, the communication technology and the communication architecture belong to the communication system group. Moreover, Figure 4-6 depicts the drivers that represent a crucial environment element to illustrate the interaction with the driving simulators, which act as central constituent systems of the networked driving simulation system. In this work, a modification is made on the representation of information flows between the constituent systems and building components through the communication system, as well as the information flows between the networked driving simulation system and its environment elements. Specifically, information flow lines of signal buses are used instead of separate information flow lines of individual signals. That is, each signal bus combines a group of information flow signals. The signal buses are described with signal tables, where the signals are characterized by attributes, such as name, unit, and description. This particular modification is necessary for the convenience of graphical demonstration of the active structure and environment partial models. A similar modification was adopted in Reference [Has14] – nonetheless, on the information flows between the internal building components of the driving simulator that represents the main system under development. Figure 4-7 shows an example signal bus table of the information flow between the drivers and the driving simulators.



Simulator	Signal name	Unit	Description
1	Acceleration	%	Acceleration pedal kick
1	Brake	%	Brake pedal kick
1	Steering	Deg.	Steering wheel angle
1	…	…	…
2	Acceleration	%	Acceleration pedal kick
2	…	…	…

Figure 4-7: Excerpt of the active structure showing an example signal bus table for the information flow between the human drivers and the simulators

The collective results of the specified partial models form a principle solution that acts as a communication and cooperation basis between experts of the involved development domains [ABD+14, p. 119]. Specifically, this basis is used for the subsequent design and development phases of the networked driving simulation system in this work.

4.4 Phase 2 – System Components Analysis

The objective of the second development phase is the identification, description, and classification of the components of the networked driving simulation system. This development phase depends mainly on the results of the preceding phase. Nonetheless, before the identification of the system components, a clear distinction must be made between the **constituent systems** and the **building components**. On the one hand, the constituent systems are independent participants of the networked driving simulation as a system of systems. That is, they can carry out meaningful tasks of their own, even if they are not networked to an entire system. On the other hand, the building components can present services to the system of systems. However, they cannot carry out meaningful tasks of their own, more specifically, when they are not networked to one or more constituent systems. The following subsection identifies the system components of the networked driving simulation system.

4.4.1 Identify System Components

The active structure partial model revealed initially the possible five system components of the networked driving simulation system. These system components can be identified further according to the presented distinction between the constituent systems and building components of the networked driving simulation. On the one hand, the driving and traffic simulators are constituent systems. They can be used separately for useful, independent purposes. On the other hand, the workstations, the database consoles, and the communication system are building components. They present services to the entire system, but they are not useful if utilized independently without constituent systems. Table 10 presents the five system components and the clear distinction between the constituent systems and the building components.

Table 10: Distinction between constituent systems and building components

Distinction	System components of networked driving simulation				
	Driving simulators	Traffic simulators	Work-stations	Database consoles	Communication systems
Constituent system	x	x			
Building component			x	x	x

Based on the results of the identification task, the following subsection describes the five system components by elaborating their role within the networked driving simulation as a system of systems.

4.4.2 Describe System Components

The five main identified system components of the networked driving simulation system can be characterized as **essential** and **optional** according to their roles. Essential system components are vital to achieve the central purpose of networked driving simulation: multi-driver traffic scenario simulation. However, the system of networked driving simulation can operate principally without the optional components and still achieve this central purpose. The following is a concise description and role characterization of each component of the networked driving simulation system from a solution-neutral perspective.

- **Driving simulators**

Driving simulators are operated by human drivers to control the respective simulated vehicles. The driving simulators can be of different types, such as a passenger car simulator or a truck simulator. Moreover, driving simulators of different complexity grades can principally participate within the networked driving simulation system. By any means, the participation of at least two driving simulators is necessary not only to achieve the central purpose, but also to establish a system of networked driving simulation. Hence, driving simulators are characterized as essential constituent systems in general. If a third driving simulator is added to the environment, one driving simulator can be considered eventually as an optional component for the realization of the networked driving simulation system.

- **Traffic simulators**

Traffic simulators generate traffic participants, such as programmed vehicles and pedestrians, to add more complexity to the multi-driver traffic scenario. One traffic simulator is often sufficient for the system of networked driving simulation. However, more than one traffic simulator can be integrated within the system to provide different traffic simulation granularity levels, such as macroscopic and microscopic traffic flows [Pot11], [HK12]. The system of networked driving simulation can operate principally without the utilization of traffic simulators. In this case, the multi-driver traffic scenario simulation depends only on the involved driving simulators. Hence, traffic simulators are characterized as optional constituent systems.

- **Workstations**

A workstation is utilized to provide control and monitoring operations on the system of networked driving simulation. That is, the simulation operator can make commands to stop and start the entire system or particular building components. Moreover, the simu-

lation operator can monitor various signals that give indications about the operation and performance of the whole system and its building components. Principally, the system of networked driving simulation can operate without the use of a workstation. Hence, the workstation is characterized as an optional building component.

- **Database consoles**

A database console is utilized to capture and save the simulation data. Moreover, operators and developers can conduct simulation analysis or generate review reports. However, the system of networked driving simulation can operate without a database console. Hence, the database console is characterized as an optional building component.

- **Communication systems**

In this work, the communication system includes two categories of building components: communication technologies and communication architectures. These building components are described as follows:

Communication technologies: This category includes different communication technologies responsible for information exchange, such as Ethernet, CAN, and FlexRay. These communication technologies differ mainly through the provided networking characteristics. The system of networked driving simulation cannot operate without a communication technology. Hence, the communication technologies are characterized as essential building components.

Communication architectures: This category includes different communication architectures responsible for networked simulation management, such as Distributed Interactive Simulation (DIS) [XSC+11]. These communication architectures differ mainly through the provided functions and services for networked simulation. Unlike the communication technologies, the system of networked driving simulation can operate without a communication architecture. Hence, the communication architectures are characterized as optional building components.

Table 11 shows the identified and described system components together with their role significance within the networked driving simulation system.

Table 11: Role significance of constituent systems and building components

Role significance	System components for networked driving simulation					
	Constituent systems		Building components			
	Driving simulators	Traffic simulators	Work-stations	Database consoles	Communication systems	
					Comm. technologies	Comm. architectures
Essential component	x				x	
Optional component		x	x	x		x

The identification and description of system components provided more understanding towards system concretization. Using the results of the identification and description tasks, the following subsection classifies the five system components into two main categories as an essential preparation step for the subsequent development phases.

4.4.3 Classify System Components

The system components can be classified generally into two main groups based on the functions and active structure partial model established in the previous development phase: **simulation system group** and **communication system group**. This particular classification conforms to the two main system layers of the reference architecture presented in Section 3.5. The classification is concretized further in this task. On the one hand, the simulation system group is divided into simulation entities and supplementary components. The different driving and traffic simulators are assigned to the simulation system group under the simulation entities. Moreover, individual components of driving simulators, such as visualization systems and motion platforms, belong to the simulation entities as well. Comprehensive identification, description, and classification of the individual components of driving simulators are presented in References [Neg07] and [Has14]. The different workstations and database consoles can be assigned to the simulation system group. However, these belong to the supplementary components to indicate a relative unimportant role within the system of networked driving simulation. On the other hand, the communication system group includes the various communication technologies and communication architectures.

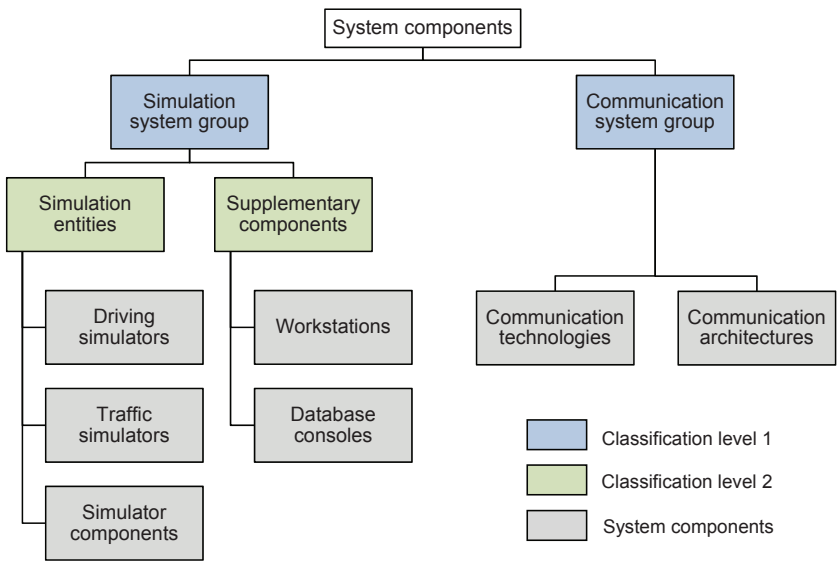


Figure 4-8: Classification of networked driving simulation system components

This particular classification reflects the general role of the five main system components within the networked driving simulation as a system of systems. It is used as basis for the next development phases. The following section presents the development of system databases and the deployment of solution elements.

4.5 Phase 3 – System Databases Development

The third development phase depends on the results of the preceding phases and presents another step towards system concretization. The objective of this development phase is to build system databases, which contain entries of the analyzed system components that are concretized through solution elements. While system components are solution-neutral, the solution elements represent concrete products of the system components from different developers and manufacturers. In addition to building system databases, an approach to fill the system databases with entries of the solution elements is presented in this development phase. The system databases are accessible and editable through the SoS configuration software. This is necessary for the subsequent development phases that address the configuration of the networked driving simulation system and the generation of system models. The following subsection elaborates the types and structure of the system databases.

4.5.1 Build System Databases for Solution Elements

In this task, system databases are developed based on the system components classification presented in the previous phase. More specifically, two system databases are built in accordance with the two main groups of system components: **simulation system database** and **communication system database**.

On the one hand, the simulation system database includes only solution elements of components related to the simulation task of the overall system of networked driving simulation. The simulation system database has four tables representing the four component categories that belong to the simulation system group: driving simulators, traffic simulators, workstations, and database consoles. These four database tables are filled with entries of the corresponding solution elements. A database for the solution elements of the individual driving simulator components has been created and filled within HASSAN'S method [Has14]. This particular database is merged with the simulation system database developed in this work. It has tables for solution elements of three categories of driving simulator components: hardware, software, and resources. The entries of the solution elements in these tables will be used eventually during the next phase of system configuration.

On the other hand, the communication system database includes solution elements of components related to the communication task of the overall system of networked driving simulation. The communication system database has two tables representing the two

components that belong to the communication system group: communication technologies and communication architectures. Similarly, these two database tables are filled with entries of the corresponding solution elements. Figure 4-9 depicts mainly the two developed system databases and the associated tables.

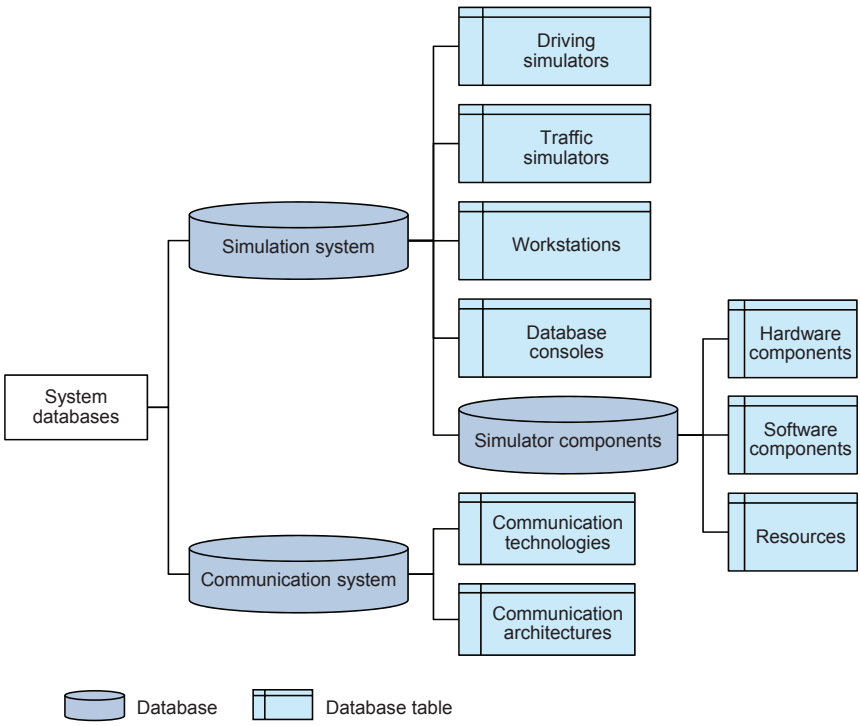


Figure 4-9: Developed system databases and their main tables

The system databases can be implemented with different database development tools. However, the selected database development tool and the implementation approach must allow the fundamental database operations [Ste08]: *C*reate, *R*ead, *U*pdate, and *D*elete. These basic database operations are summarized commonly with the acronym *CRUD* according to the first letters of their names respectively. Providing these basic operations is necessary to make the system databases accessible and editable from the SoS configuration software. The operations are described briefly as follows:

- Create: It is a database operation to create new table entries. The database shall be extendable to allow the insertion of entries of new solution elements.
- Read: It is a database operation to read data from the database tables. The database tables and the individual entries of the solution elements shall be readable from the SoS configuration software.

- Update: It is a database operation to edit the data held in the database tables. The database shall allow the update operations, so that the entries of the solution elements can be changed if necessary.
- Delete: It is a database operation to remove particular data from the database tables. The database shall allow the delete operations, so that the entries of unavailable solution elements can be removed.

The following subsection specifies the main attributes of the database tables and explains how to fill these database tables with solution elements accordingly.

4.5.2 Fill System Databases with Solution Elements

Apart from the database tables used in HASSAN'S methods, six database tables were identified in the previous development task to include solution elements of six categories of system components: **driving simulators, traffic simulators, workstations, database consoles, communication technologies, and communication architectures**. These database tables must be specified further with attributes before registering entries of corresponding solution elements. Table 12 shows the database table created to include entries of existing driving simulators as available solution elements. The shown table contains entries of three exemplary driving simulators developed at the Heinz Nixdorf Institute in Paderborn, Germany. The ID represents the key attribute of this table, where a unique identification number is assigned to each driving simulator entry.

Table 12: Database table of driving simulators (solution elements), where the individual building components are specified with fidelity levels

ID	Name	Visualization system					Motion system					Acoustic system			Driver platform			
		Eye distance (A)	Field of view (B)	Rear-view mirrors (E)	Continuity (F)	Resolution (G)	Motion standard (L)	Motion < 6 DOF (K)	Motion > 6 DOF (M)	Vehicle dynamics (N)	Tire model (O)	Primary sound (P)	Auxiliary sound (Q)	Sound system (R)	Mock-up (S)	HMI (T)	Steering (U)	Pedals set (V)
1	ATMOS simulator	2	3	2	3	2	–	3	–	3	3	1	1	1	4	2	2	1
2	Airmotion _ride	2	1	3	1	2	1	–	–	2	2	1	1	1	1	1	1	1
3	HNI PC simulator	1	1	3	1	1	–	–	–	2	2	1	–	1	1	1	1	1

In addition to the ID and name attributes, this database table has four more attributes in accordance with the four driving simulator components identified in NEGELE'S guidelines [Neg07, p. 22-60]: visualization system, motion system, acoustic system, and driver's platform. These attributes are described as follows:

- **Visualization system**

This attribute is composed of five sub-attributes: eye distance, field of view, rear-view mirrors, continuity, and resolution [Neg07, p. 37]. These sub-attributes are described together with the possible entry options as follows:

Eye distance: This sub-attribute represents the eye accommodation and can have one of three entry options: 'A1', 'A2', or 'A3'. These entry options correspond to three eye-scene distance ranges: < 0.8, 2.5–3, and > 5 meter respectively. The accommodation is a physiological function that governs the ability of the eye to focus and to view clear scenes [Kim05]. The shorter the eye-scene distance, the greater the effort required from the eye muscles for the accommodation. Short eye-scene distances increase eye discomfort [Kim05].

Field of view: This sub-attribute can have one of three entry options: 'B1', 'B2', 'B3', or 'B4'. An entry of 'B1' denotes a field of view of 40°–60° for essential minimum view coverage. An entry of 'B2' denotes a field of view of 120°–140° for partial view coverage. An entry of 'B3' denotes a field of view of 180°–220° for complete view coverage. An entry of 'B4' denotes a field of view of 360° for surround view coverage.

Rear-view mirrors: This sub-attribute can have one of three entry options: 'E1', 'E2', or 'E3'. An entry of 'E1' denotes the use of the optical mirrors of the vehicle without any modifications. An entry of 'E2' denotes the use of small display screens to represent the mirrors. An entry of 'E3' denotes the use of virtual mirrors within the front or side display systems.

Continuity: This sub-attribute can have one of three entry options: 'F1', 'F2', or 'F3'. An entry of 'F1' means that the front field of view is divided into sectors by physical vertical boundaries. An entry of 'F2' means that there are no physical vertical boundaries between the different visualization channels – however, virtual separation edges are visible. An entry of 'F3' means that there are no virtual separation edges due to the overlapping scenes of the different visualization channels.

Resolution: This sub-attribute can have one of three entry options: 'G1', 'G2', or 'G3'. These entry options correspond to three arc minutes ranges: > 6, 2–3, and < 1 arc minutes/pixel respectively. The arc minute is a unit of angular measurement that equals to 1/60 of a degree. A smaller value of arc minutes/pixel refers to a higher resolution of the display system [War12].

- **Motion system**

This attribute is composed of five sub-attributes: motion standard, motion < 6 DOF, motion > 6 DOF, vehicle dynamics, and tire model [Neg07, p. 49]. These sub-attributes are described together with the possible entry options as follows:

Motion standard 6 DOF: This sub-attribute refers to standard hexapod (Gough-Stewart) motion platforms that offer 6 DOF. It can have one of four entry options: 'L1', 'L2', 'L3', or 'L4'. An entry of 'L1' refers to a motion platform of maximum frequency of 1.5 Hz, ±25° amplitude, and 1.5 ton workload. An entry of 'L2' refers to a motion platform of maximum frequency of 30 Hz and 20-50 mm amplitude. An entry of 'L3' refers to a motion platform of maximum of 1.5 Hz frequency and 2.5 ton workload. An entry of 'L4' refers to a motion platform of maximum of 1.5 Hz frequency, but more than 2.5 ton workload. While the first two entry options refer to platforms that do not carry a visualization system, the latter two entry options refer to platforms, on which the visualization system is mounted.

Motion < 6 DOF: This sub-attribute refers to non-standard motion platforms that offer less than 6 DOF. It can have one of three entry options: 'K1', 'K2', or 'K3'. An entry of 'K1' refers to a motion platform of 1 DOF translation motion that has a maximum frequency of 30 Hz and 20–50 mm amplitude. An entry of 'K2' refers to a motion platform of 3 DOF (1 DOF translation motion + 2 DOF rotational motion) that has a maximum of 1.5 Hz frequency and ±10° amplitude. An entry of 'K3' refers to a motion platform of 3 DOF (1 DOF translation motion + 2 DOF rotational motion) that has a maximum of 1.5 Hz frequency and ±25° amplitude.

Motion > 6 DOF: This sub-attribute refers to standard hexapod (Gough-Stewart) motion platforms that can offer more than 6 DOF with additional complementary systems. It can have one of four entry options: 'M1', 'M2', 'M3', or 'M4'. An entry of 'M1' refers to a motion platform of 7 DOF (6 DOF standard motion + 1 DOF translation motion) that has a maximum frequency of 1.5 Hz and a translation motion path of several meters. An entry of 'M2' refers to a motion platform of 8 DOF (6 DOF standard motion + 2 DOF translation motions) that has a maximum of 1.5 Hz frequency and translation motion paths of several meters. An entry of 'M2' refers to a motion platform of 9 DOF (6 DOF standard motion + 2 DOF translation motions + 1 DOF rotational motion) that has a maximum of 1.5 Hz frequency, translation motion paths of several meters, and an unlimited rotational motion course.

Vehicle dynamics: This sub-attribute can have one of three entry options: 'N1', 'N2', or 'N3'. While an entry of 'N1' denotes a single-track vehicle model, an entry of 'N2' denotes double-track vehicle model. An entry of 'N3' refers to a multi-body system model.

Tire model: This sub-attribute can have one of four entry options: 'O1', 'O2', 'O3', or 'O4'. An entry of 'O1' denotes a tire model without combined longitudinal and lateral

dynamics. An entry of ‘O2’ denotes a tire model with combined longitudinal and lateral dynamics. An entry of ‘O3’ denotes a half empirical tire model. An entry of ‘O4’ denotes a 3D finite-element tire model.

- **Acoustic system**

This attribute is composed of three sub-attributes: primary sound, auxiliary sound, and sound system [Neg07, p. 52]. These sub-attributes are described together with the possible entry options as follows:

Primary sound: This sub-attribute can have one of three entry options: ‘P1’, ‘P2’, or a combination of both entry options. An entry of ‘P1’ refers to the sound associated with the ego-vehicle, such as motor sound. An entry of ‘P2’ refers to more advanced sound effects, such as the sound reflected when the ego-vehicle passes by trees.

Auxiliary sound: This sub-attribute can have one of three entry options: ‘Q1’, ‘Q2’, or a combination of both entry options. An entry of ‘Q1’ refers to auxiliary sound associated with the ego-vehicle, such as the sound of the turn indicator and gearbox. An entry of ‘Q2’ refers to the sound associated with the traffic vehicles that depends on their distance to the ego-vehicle.

Sound system: This sub-attribute can have one of three entry options: ‘R1’, ‘R2’, or ‘R3’. While an entry of ‘R1’ refers to a two-channel sound system, an entry of ‘R2’ refers to a sound system of more than three channels. An entry of ‘R3’ refers to a sound system with a pressure transducer that generates bass effects.

- **Driver’s platform**

This attribute is composed of four sub-attributes: mock-up, HMI, steering, and pedals set [Neg07, p. 56]. These sub-attributes are described together with the possible entry options as follows:

Mock-up: This sub-attribute can have one of four entry options: ‘S1’, ‘S2’, ‘S2’, or ‘S4’. An entry of ‘S1’ refers to a simple driving seat. An entry of ‘S2’ refers to a partial vehicle with apparent body modifications. An entry of ‘S3’ refers to a complete vehicle, where the body modifications are not apparent only during driving. An entry of ‘S4’ refers to a complete vehicle, where no modifications are apparent at all.

HMI: This sub-attribute can have one of three entry options: ‘T1’, ‘T2’, or ‘T3’. While an entry of ‘T1’ refers to a primary HMI, an entry of ‘T2’ refers to a complete HMI. An entry of ‘T3’ refers to a complete HMI with reconfigurable displays.

Steering: This sub-attribute can have one of three entry options: ‘U1’, ‘U2’, or ‘U3’. An entry of ‘U1’ refers to a steering moment that is proportional to the steering angle. An entry of ‘U2’ refers to an electrically generated steering moment that depends on the speed and the steering angle. An entry of ‘U3’ refers to a similar steering moment, however, with higher frequency (> 200 Hz).

Pedals set: This sub-attribute can have one of three entry options: ‘V1’, ‘V2’, or ‘V3’. An entry of ‘V1’ refers to a pedals set that delivers passive pressure. An entry of ‘V2’ refers to a pedals set that delivers an adaptive pressure by means of a characteristic curve. An entry of ‘V3’ refers to a pedals set that delivers an adaptive pressure, where the characteristic curve is modifiable.

Other auxiliary attributes can be added to the database table of driving simulators, such as the manufacturer, space, and electrical energy requirements of each available driving simulator.

To conform to the SoS definition adopted in this work, traffic simulation is considered as a separate task that is independent of the driving simulators in contrast to both NEGELE’S guidelines and HASSAN’S method. The generation of programmed traffic participants still may be required in addition to the networked interactive driving simulators to increase the complexity of the traffic scenarios. There are myriad alternatives of traffic simulators in the market. Developers of traffic simulators compete to provide different options and traffic characteristics. Table 13 shows the database table created to include entries of existing traffic simulators as available solution elements[3]. The table contains entries of three exemplary traffic simulators. The ID represents the key attribute of this table, where a unique identification number is assigned to each traffic simulator entry.

Table 13: Database table of traffic simulators (solution elements)

ID	Name	Developer	Environment database		Objects simulation		Granularity level			Visualization type	
			Database type (W)	District type (Y)	General vehicles (Z)	Special objects (AA)	Macroscopic	Microscopic	Mesoscopic	Two-dimensional	Three-dimensional
1	SUMO	DLR	W2	Y1	Z1/ Z2	AA1/ AA2	No	Yes	No	Yes	No
2	AIMSUN	TSS	W3	Y3	Z1/ Z2	AA1	Yes	Yes	Yes	Yes	Yes
3	HNI-Traffic	HNI	W2	Y3	Z2	AA1	No	Yes	No	Yes	No

Basically, this database table is constructed in accordance with the definition and classification of simulator components in NEGELE’S guidelines. It combines characteristics

[3] The values of the characteristics presented in this table may differ in future as the considered exemplary traffic simulators are subjected to continuous changes and updates by their respective developers.

related the environmental objects, such as road network and buildings, and the simulated traffic objects, such as programmed vehicles and pedestrians. Specifically, in addition to the ID, name, and developer attributes, this database table has four more attributes: environment database, objects simulation, granularity level, and visualization type[4]. The following is a description of these attributes and their entry options.

- **Environment database**

This attribute is inherited from NEGELE'S guidelines with respect to the features of the environmental objects database [Neg07, p. 59]. It is composed of two sub-attributes: database type and district type.

Database type: This sub-attribute can have one of three entry options: 'W1', 'W2', and 'W3'. An entry of 'W1' means that the available set of environmental objects, such as the road networks and landscape, is predetermined and cannot be adjusted. An entry of 'W2' means that the available set of environmental objects is modular and can be configured offline. An entry of 'W3' means that the available set of environmental objects is modular, configurable, and can be automatically adjusted according to the driving course and behavior.

District type: This sub-attribute can have one of three entry options: 'Y1', 'Y2', and 'Y3'. An entry of 'Y1' means that the environmental objects, such as buildings, tree, vegetation, and mountains, can be generated only in the near simulation field. An entry of 'Y2' means that these environmental objects can be generated only in the far simulation field. An entry of 'Y3' means that the generation of these environmental objects is adjustable.

- **Objects simulation**

This attribute is inherited from NEGELE'S guidelines with respect to the features of the traffic objects simulation [Neg07, p. 59]. It is composed of two sub-attributes: general vehicles and special objects.

General vehicles: This sub-attribute can have one of three entry options: 'Z1', 'Z2', or a combination of both entry options. An entry of 'Z1' means that traffic vehicles are generated randomly and follow the traffic rules. An entry of 'Z2' means that the traffic density can be adjusted together with some global driving parameters, such as maximum speed and acceleration, for all traffic vehicles.

Special objects: This sub-attribute can have one of four entry options: 'AA1', 'AA2', 'AA3', or any combination of the three entry options. An entry of 'AA1' means that the behavior of traffic vehicles can be automatically adjusted according to triggering events. An entry of 'AA2' means that special traffic participants, such as bicyclists, pedestrians,

[4] This database table can be eventually extended to include attributes for further characteristics of traffic simulators. This aspect conforms to the sixth requirement presented in Section 2.6.2.

or animals, are available and their behavior can be automatically adjusted according to triggering events. An entry of ‘AA3’ means that at least one traffic vehicle can be controlled manually by the simulation operator.

- **Granularity level**

This attribute is composed of three sub-attributes in accordance with three main granularity levels of traffic simulation: macroscopic, microscopic, and mesoscopic [HK12]. Each of the three sub-attributes can have one of two entry options: ‘Yes’ or ‘No’.

Macroscopic: Macroscopic traffic simulation is concerned with collective measureable quantities, such as average speed, flow rate, and density. That is, this traffic simulation type considers the traffic flow from a global perspective and do not distinguish the behavior of individual vehicles [HK12]. Simulation time and memory requirements are not restricted by the number of simulated vehicles. Therefore, macroscopic traffic simulators are suitable for faster than real-time traffic simulations.

Microscopic: Microscopic traffic simulation is concerned with the individual characteristics of traffic vehicles, such as speed, acceleration, and lane change. Each traffic vehicle is assigned to a type, such as passenger car, truck, or bus. The traffic vehicle type determines the vehicle characteristics. Moreover, each traffic vehicle is assigned to a driver behavior, such as aggressive or cautious. This gives each vehicle a unique performance to maintain while traveling through the entire road network. Hence, this traffic simulation type permits more detailed analysis of the interactions between the simulated vehicles [HK12].

Mesoscopic: Mesoscopic traffic simulation can be considered as a compromise between the features of the macroscopic and microscopic traffic simulations. In other words, the mesoscopic traffic simulation describes the traffic flow in greater detail than the macroscopic models, but in lesser detail than the microscopic models. In the mesoscopic traffic simulation, groups of traffic vehicles are formed to represent single units of different features [HK12]. For instance, a group of vehicle can be generated to travel on a highway as a single unit with the same driving style or characteristics.

- **Visualization type**

This attribute is composed of two sub-attributes: two dimensional and three dimensional. These sub-attributes can have one of two entry options: ‘Yes’ or ‘No’. Apart from the utilization with driving simulators, traffic simulators in general offer 2D and/or 3D visualization of the traffic scenarios for design, monitoring, and analysis purposes. While the traffic flow is observed from a top-view with the 2D visualization option, it is observed from a perspective view with the 3D visualization option. Although the 3D visualization option delivers impressive scenes, it may impose special requirements on the graphic cards.

The third member of the simulation system database is the workstations table. This database table includes entries of workstations of different capabilities or specifications. The set of attributes can include the ID, name, manufacturer, number of monitors, and computer specifications, etc. The fourth member of the simulation system database is the database consoles table. Similarly, this database table includes entries of database consoles of different capabilities or specifications. The set of attributes can include the ID, name, developer, design software, interfaces, storage capacity, and any other computer specifications.

With respect to the database tables of the individual driving simulator components, some of the solution elements can be characterized according to the features identified in NEGELE'S guidelines [Neg07, pp. 22–59]. Specifically, nine driving simulator components were identified in HASSAN'S method. Figure 4-10 shows these driving simulator components and the main classification.

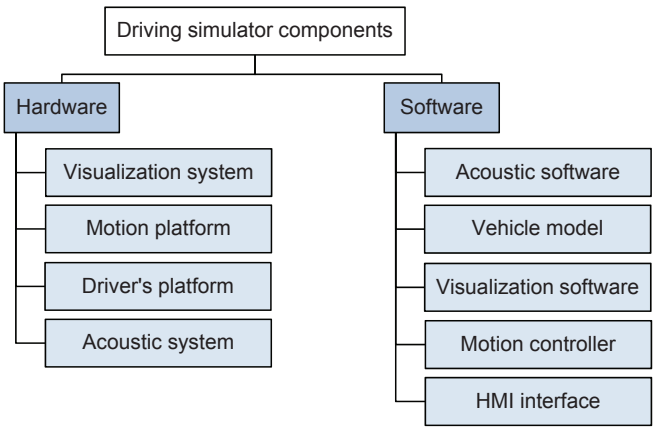


Figure 4-10: The driving simulator components as identified and classified in HASSAN'S method – reconstructed according to Reference [Has14, p. 89]

The visualization software, motion platform controller, and HMI interface in HASSAN'S method do not have corresponding features in NEGELE'S guidelines. Hence, the remaining six driving simulator components from HASSAN'S method are correlated to features and fidelity levels from NEGELE'S guidelines in this work. This correlation step represents the link between HASSAN'S method and NEGELE'S guidelines. Table 14 shows a complementary database table for the database designed in HASSAN'S method and integrated with the simulation system database in this work. The table includes six exemplary solution elements representing six driving simulator components from HASSAN'S method specified with feature fidelity levels from NEGELE'S guidelines.

Table 14: Complementary database table for the solution elements of HASSAN'S method specified with feature fidelity levels of NEGELE'S guidelines

ID	Solution elements	Visualization system					Motion platform			Driver platform				Acoustic system	Acoustic software		Vehicle model	
		Eye distance (A)	Field of view (B)	Rear-view mirrors (E)	Continuity (F)	Resolution (G)	Motion standard (L)	Motion < 6 DOF (K)	Motion > 6 DOF (M)	Mock up (S)	HMI (T)	Steering (U)	Pedals set (V)	Sound system (R)	Primary sound (P)	Auxiliary sound (Q)	Vehicle dynamics (N)	Tire model (O)
1	ATMOS visual. system	2	3	2	3	2												
2	Festo motion platform						1	–	–									
3	Smart vehicle									4	2	2	1					
4	Desktop speakers 2.0													1				
5	HNI acoustic module														1	–		
6	ASM dSPACE																3	3

The ID attribute contains the ID numbers of the driving simulator components, to which the solution elements belong. In addition to the name as key attribute, this database table has six more attributes representing the identified driving simulator components from HASSAN'S method: visualization system, motion platform, driver's platform, acoustic system, acoustic software, and vehicle model. The attributes are composed of sub-attributes, which contain the correlated features from NEGELE'S guidelines. Thereby, all available solution elements are characterized by specific fidelity levels.

With respect to the communication system, Table 15 shows the database table created to include entries of all available communication technologies as solution elements. These communication technologies differ mainly through the network characteristics. The shown table contains entries of three exemplary communication technologies[5]. The ID represents the key attribute of this table, where a unique identification number is assigned to each communication technology entry.

Table 15: Database table of communication technologies (solution elements)

ID	Name	Network characteristics							
		Bandwidth (Mbps)	Latency (ms)	Jitter (µs)	Packet loss rate (%)	Determinism	Error ratio	Transmission mode	Segment length (m)
1	Ethernet (10Base-T)	10	0.123	1	5	No	10^{-9}	Half	100
2	FireWire 400 (IEEE 1394)	400	0.041	$5*10^{-4}$	0.5	Near	10^{-12}	Full	4.5
3	Ethernet (100Base-T)	100	1.122	1	3	No	10^{-9}	Half	100

In addition to the ID and name attributes, this database table has one main attribute representing significant network characteristics of the communication technologies[6] [Mir06]. For example, this attribute can be divided into eight sub-attributes: bandwidth, latency, jitter, packet loss rate, determinism, error rate, transmission mode, and segment length. These sub-attributes are described as follows:

- **Bandwidth:** It represents the data transfer rate in the context of computer networks. Specifically, it is the theoretical amount of data that can be transferred from one point to another along a network segment in a given time period. The bandwidth is usually expressed in bits per second (bps).
- **Latency:** In the context of computer networks, it is the time required for a data packet to travel from one point to another. The latency is expressed in units of time.

[5] The values of the network characteristics presented in this table serve the explanation purpose only. The considered example communication technologies may be subjected to changes by the different developers and vendors. Hence, the actual values of their network characteristics may differ.

[6] This attribute can be eventually extended to include sub-attributes for further characteristics of the communication technologies. This aspect conforms to the sixth requirement presented in Section 2.6.2.

- **Jitter:** It is the latency variation of packets traveling from one point to another. The jitter is expressed as a fraction of unit interval (UI) or in time units, such as picosecond or microsecond.
- **Packet loss rate:** Packet loss is typically caused by network congestions. The packet loss rate is expressed as the percentage of the lost packets to the total sent packets during a specified time interval.
- **Determinism:** It is the successful and predictable delivery of data packets through a network. Some communication technologies offer a capability of deterministic data delivery within specified time periods.
- **Error ratio:** The bit errors represent the number of received bits that have been altered due to noise, distortion, or interference. The bit error ratio is the number of bit errors to the total transferred bits during a specified time interval.
- **Transmission mode:** It is the direction of data packets flow between the communicating nodes. There are three types of transmission mode: simplex mode, half duplex mode, and full duplex mode. In the simplex mode, data packets can be sent only in one direction. In half duplex mode, data packets can be sent in both directions between the communicating nodes, however, only one direction is considered at a time. In full duplex mode, data packets can be sent in both directions between the communicating nodes simultaneously.
- **Segment length:** The segment is a portion of the network connecting two communicating devices or systems. This characteristic is based on the signal attenuation in terms of dB loss per unit length for each meter of the transmission medium. Specifically, after a certain length of the transmission medium, the signal strength falls below a critical threshold and the delivery of data packets becomes unreliable. The attenuation rate depends mainly on the material of the transmission medium.

In addition to the communication technologies, Table 16 shows the database table created to include entries of all available communication architectures as solution elements. These communication architectures differ mainly through the provided functions and services for networked simulation. The shown table contains entries of three exemplary communication architectures[7]. The ID represents the key attribute of this table, where an identification number is assigned to each communication architecture entry. In addition to the ID and name attributes, this database table has one main attribute representing significant characteristics, functions, and services of the communication architectures[8] [XSC+11], [WYX+09].

[7] The assigned values in this table may differ in future as the considered exemplary communication architectures may be subjected to modifications by their respective standard developers.

[8] This attribute can be eventually extended to include sub-attributes for further functions or services of the communication architectures. This aspect conforms to the sixth requirement presented in Section 2.6.2.

Table 16: Database table of communication architectures (solution elements)

ID	Name	Functions and services				
		Central management	Specified data packet layout	Data packet delivery	Distribution management	Time management
1	HLA	Yes	No	BE/RE	Yes	Yes
2	DIS	No	Yes	BE	No	No
3	TENA	Yes	Yes	RE	No	No

The main attribute can be divided into six sub-attributes for example: central management, specified data packet layout, data packet delivery, data distribution management, declaration management, and time management. These sub-attributes are described as follows.

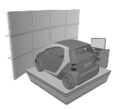
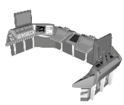
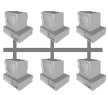
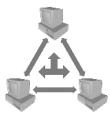
- **Central management:** This sub-attribute can have one of two entry options: 'Yes' or 'No'. An entry of 'Yes' means that a central management unit is responsible for the information exchange. An entry of 'No' means that the information exchange occurs directly between the participating networked systems.
- **Specified data packet layout:** This sub-attribute can have one of two entry options: 'Yes' or 'No'. An entry of 'Yes' means that a data packet layout is predetermined. An entry of 'No' means that the applications can determine data packet layout arbitrarily.
- **Data packet delivery:** This sub-attribute can have one of three entry options: 'BE', 'RE', or a combination of both entry options. An entry of 'BE' refers to a best-effort data packet delivery, where there is no guarantee about data delivery or quality of service. An entry of 'RE' refers to the support of reliable data packet delivery.
- **Distribution management:** This sub-attribute can have one of two entry options: 'Yes' or 'No'. An entry of 'Yes' means that a service of data distribution management is available to provide an active mechanism for data routing between the participating networked systems [SZ14]. An entry of 'No' means that the data distribution management service is not supported.
- **Time management:** This sub-attribute can have one of two entry options: 'Yes' or 'No'. An entry of 'Yes' means that a service of time management is available to provide a mechanism to control time advancement of the participating systems during the simulation. An entry of 'No' means that the time management service is not supported.

Further attributes can be added to the database table of communication architectures, such as the provider and the version of the underlying standard. This means that there would be multiple entries of the same communication architectures representing different providers and standard versions. The following section discusses a central development phase of the system-level design framework. Specifically, this phase determines the methods used to configure the networked driving simulation system.

4.6 Phase 4 – Configuration Methods Development

The previous phases provided a comprehensive understanding of the networked driving simulation system. The system components were identified, described, and classified. Moreover, system databases were developed in accordance with the results of the system components analysis. Table 17 shows a morphological box that includes the identified system components together with exemplary solution elements (symbolic).

Table 17: Morphological box of the networked driving simulation system (excerpt)

System components	Solution elements		
	1	2	3
Driving simulators			
Traffic simulators			
Workstations			
Database consoles			
Communication technologies			
Communication architectures			

In general, the morphological box approach introduced by ZWICKY represents a well-established formation scheme that can be used when it comes to system composition [Zwi69], [Zwi89], [PBF+07]. In this work, the entries of system components and the corresponding available solution elements are inserted into the rows of a morphological matrix as shown in Table 17. While the first four system components belong to the simulation aspect of the networked driving simulation system, the latter two system components belong to the communication aspect. The solution elements of system components must be combined systematically to obtain an overall solution. The networked driving simulation system is composed of independent, heterogeneous systems. Therefore, the combination of solution elements is not governed by their consistency or compatibility in this work in contrast to HASSAN'S method for example [Has14]. The solution elements are selected according to the offered capabilities and functionalities with respect to the requirements of the concerned application scenarios. The following subsection presents a configuration method for the simulation aspect of the networked driving simulation system.

4.6.1 Simulation System Configuration Method

The simulation system aspect in this work includes four system components: driving simulators, traffic simulators, workstations, and database consoles. The application-oriented selection of the participating driving simulators is the central task of the simulation system configuration. However, available driving simulators must be classified in accordance with their capabilities to account for the subsequent phase of application-oriented system model configuration and generation.

- **Selection approach for available driving simulators**

Classifying driving simulators into three categories (low-level, mid-level, and high-level) collectively is not practical [Jam11]. A driving simulator may have high capability for one particular component and low capability for other components according to the purpose of use [CJ11]. Hence, driving simulators are classified in this work according to the 15 application classes defined in NEGELE'S guidelines [Neg07, pp. 94–98]. These application classes give more insight into the fidelity levels of the individual driving simulator components. While seven application classes are considered as not common or practical, eight application classes are fully specified in NEGELE'S guidelines [Neg07, pp. 99–114]. Nonetheless, the specifications of available driving simulators may not exactly fulfill the whole requirements of the application classes. Therefore, a cost function must be defined to give a quantified indication of the deviations.

The relative significances of the individual driving simulator components are specified for each application class in NEGELE'S guidelines [Neg07, pp. 99–114]. For instance, the motion system is more significant than the visualization system for the application class 1a (skill-based responses and vehicle stabilization). In this work, the relative significance is quantified, where each driving simulator component takes a unique integer

number from 1 to 4. Higher numbers mean higher relative significances. With respect to the application class 1a for example, the significance numbers: 4, 3, 1, and 2 are assigned to the simulator components: motion system, visualization system, acoustic system, and driver's platform respectively. Based on the specified and quantified relative significances, Equation 1 presents the cost function developed in this work to give an indicative measure of the deviation between the specifications of the available driving simulators and the requirements of the application classes:

$$Fidelity\ level\ deviation = \frac{1}{4} \times \left(\sum_{m=1}^{4} \left(\frac{Significance_m}{N} \times \sum_{n=1}^{N} |Level_{Req(n)} - Level_{Spec(n)}| \right) \right)$$

Equation 1: A cost function for determining the deviation between the specifications of driving simulators and the requirements of the application classes

Where:

m:	Designation of the driving simulator component m
$Significance_m$:	Specified relative significance value of a component m
n:	Designation of the component feature
N:	Maximum number of component features
$Level_{Req}$:	Requirement feature fidelity level of a component
$Level_{Spec}$:	Specification feature fidelity level of a component

This cost function is applied to each available driving simulator with respect to each application class. The minimum deviation value among all available driving simulators with respect to a particular application class indicates a best possible match between the specifications and the requirements. With respect to any particular application class, Equation 2 presents a simple function used to find the minimum deviation value among all driving simulators.

$$Best\ matching\ simulator = Min(Deviation_1, Deviation_2, Deviation_3, \dots, Deviation_x)$$

Equation 2: A simple minimum deviation function for determining the best matches between driving simulators and the application classes

Where x is the number of available driving simulators.

The resulting minimum deviation value is used to select the driving simulator, whose specifications best match the requirements of a concerned application class. Further cost functions can be eventually developed, provided that they can give unique selections of driving simulators. The developed cost function has been applied to the three exemplary driving simulators of Table 12 and showed very good results in terms of the provided

unique selections. Table 18 shows these results with respect to the eight specified application classes.

Table 18: The results of the developed cost function for three exemplary driving simulators

Driving simulators	Specified driving simulator application classes							
	1a	2b	3b	3c	4b	4c	5b	5c
ATMOS driving simulator	2.04	2.15	3.22	3.22	2.00	**1.81**	2.00	2.00
Airmotion_ride driving simulator	3.26	2.23	1.48	1.48	**1.38**	2.38	1.80	1.80
HNI PC-based driving simulator	3.15	2.30	**1.30**	**1.30**	1.96	2.96	1.78	1.78

Exemplary minimum deviation values are highlighted in Table 18 supposing that the database table contains entries of only three driving simulators. For instance, the HNI PC-based driving simulator can be used for the application class 3b (navigation driving tasks with rule-based responses) and the application class 3c (navigation driving tasks with knowledge-based responses). The Airmotion_ride driving simulator can be used for the application class 4b that addresses secondary driving tasks with rule-based responses. The ATMOS driving simulator can serve the application class 4c that addresses secondary driving tasks with knowledge-based responses. If other entries of driving simulators are added to the database table and the cost function is applied, the results of the minimum deviation function with respect to each application class will change.

- **Selection approach for further available simulation system components**

The selection of solution elements of traffic simulators, workstations, and database consoles is more convenient and straightforward due to the limited number of characterizing features. A similar approach can be applied to these simulation system components, however, without the use of a particular predetermined significance factor. That is, the deviation between the specifications of the available solution elements and the requirements of the application scenarios can be determined through simple cost functions or comparison tables.

- **Selection approach for available driving simulator components**

The available driving simulators may not be satisfactory if the deviations between their specifications and the requirements of the concerned application classes are large. In such cases, new driving simulators can be built through combining solution elements of the different driving simulator components. HASSAN'S method can be utilized for this particular purpose [Has14]. In this regard, the previous development phase presented an approach to assign feature fidelity levels from NEGELE'S guidelines to the solution ele-

ments of the driving simulator components specified in HASSAN'S method. Moreover, a complementary database table has been created to include these fidelity assignments.

According to NEGELE'S guidelines, driving simulator components are characterized by a set of features. For instance, the visualization system is characterized by five features: eye distance, field of view, rear-view mirrors, continuity, and resolution. Nonetheless, features of an available solution element may not have together the exact fidelity levels that fulfill the corresponding requirements of a particular application class. Therefore, a cost function must be defined to give a quantified indication of the deviation. No relative significances for the features of the individual driving simulator components are specified in NEGELE'S guidelines. Hence, no particular significance factor is used within the cost function. Equation 3 presents the cost function developed in this work to give an indicative measure of the deviation between the specifications of individual driving simulator components and the corresponding requirements of the application classes:

$$Fidelity\ level\ deviation = \sum_{y=1}^{Y} |Level_{Req(y)} - Level_{Spec(y)}|$$

Equation 3: A cost function for determining the deviation between the specifications of individual simulator components and the corresponding requirements of the application classes

Where:

y:	Designation of the feature of a driving simulator component
Y:	Maximum number of features of a driving simulator component
$Level_{Req}$:	Requirement feature fidelity level of a simulator component
$Level_{Spec}$:	Specification feature fidelity level of a simulator component

This cost function is applied to all available solution elements of each driving simulator component with respect to the corresponding requirements of each application class. Among all solution elements of a particular driving simulator component, the minimum deviation value indicates a best possible specifications match to the corresponding requirements of the concerned application class. With respect to any particular driving simulator component and an application class, Equation 2 presents a simple function used to find the minimum deviation value among all solution elements.

$$Best\ matching\ component = Min(Deviation_1, Deviation_2, Deviation_3, \dots, Deviation_g)$$

Equation 4: A simple minimum deviation function for determining the best matches between individual simulator components and the application classes

Where g is the number of available solution elements of any particular driving simulator component.

This approach complements the process of driving simulator configuration introduced in HASSAN'S method [Has14]. Specifically, the selection of solution elements of driving simulator components is governed by their capability to fulfill the corresponding requirements of the concerned application scenarios. The following subsection presents a configuration method for the communication aspect of the networked driving simulation system.

4.6.2 Communication System Configuration Method

The communication system aspect in this work includes two system components: communication technologies and communication architectures. While using a communication technology is essential for the operation of the networked driving simulation system, the communication architectures are classified as optional building components. There are a lot of solution elements for both building components from different providers. Moreover, the available solution elements are usually subjected to continuous development to establish variants of different characteristics and functions. A careful selection of the communication system is necessary to reach the expected outcomes of the networked driving simulation system.

- **Determining priority of communication characteristics and functions**

Communication technologies are characterized typically by a set of network characteristics, such as bandwidth and latency. However, no particular communication technology can provide the best possible specifications regarding all network characteristics [DT95]. Therefore, it may be difficult to find an absolute optimal solution element for all application scenarios due to the presence of various conflicting network characteristics and myriad choices of available communication technologies.

This challenge leads to a typical multi-criteria decision-making problem [Tri00]. Thereby, the network characteristics represent the criteria and the communication technologies represent the alternatives. In general, there are different methods in the literature to handle multi-criteria decision making problems [Tri00]. Nonetheless, some of these methods require a prior assignment of priority weights to the different criteria. This is necessary to ultimately reach a compromised solution. The compromised solution in this regard refers to a choice that satisfies the most important criteria to a far extent while partially satisfying the less important criteria.

Hence, the network characteristics in this context must be prioritized according to the requirements of the concerned application scenarios. Table 19 shows an excerpt of a priority analysis matrix that can be used to assign priority weights to the network characteristics of the communication technologies. At this stage, the priority weighting process is solution-neutral, i.e., no particular communication technologies are considered.

Table 19: Excerpt of a priority analysis matrix with example communication characteristics

Priority analysis matrix								
Priority scheme 1 = equally important 5 = more important 10 = much more important 0.2 = less important 0.1 = much less important		Network characteristics					Weights sum	Priority weight (%)
		Bandwidth	Packet loss rate	Determinism	Latency	Segment length		
Network characteristics	Bandwidth		5	0.2	1	10	16.2	29.35
	Packet loss rate	0.2		0.2	1	5	6.4	11.59
	Determinism	5	5		5	10	25	45.29
	Latency	1	1	0.2		5	7.2	13.04
	Segment length	0.1	0.2	0.1	0.2		0.4	0.730

The relevant network characteristics are listed vertically and horizontally in the shown priority analysis matrix. Based on the requirements of a concerned application scenario, a relative priority weight is assigned to each pair of different network characteristics according to a priority scheme. The priority scheme used in this work includes five levels as shown in Table 19. Exemplary relative priority weights are presented in Table 19 for a set of five network characteristics. For instance, the bandwidth can be less important than the determinism, but much more important than the segment length for a particular application scenario. The overall relative priority weight of each network characteristic is calculated as a sum of weights. Ultimately, the final priority weighting percentages of all network characteristics are calculated according to Equation 6.

$$Priority\ weighting_n\ (\%) = \ (Weights\ sum_n \times 100) \ / \left(\sum_{m=1}^{M} Weights\ sum_m \right)$$

Equation 5: Calculation of the priority weighting percentages of the individual network characteristics

Where:

n and m: Designations of the network characteristics n and m

M: Total number of network characteristics

The priority weighting percentages reflect the unique significances of the network characteristics for a concerned application scenario. Although the shown example priority analysis matrix includes only five network characteristics, it can be extended vertically and horizontally to include more network characteristics as desired.

Similarly, communication architectures are characterized typically by a set of functions for networked simulation. However, no particular communication architecture can provide the best possible specifications regarding all functions. Hence, the communication functions must be prioritized with respect to the application scenarios to reach a compromised solution. With the same approach used for the communication technologies, priority weights can be assigned to the functions of the communication architectures according to the requirements of the concerned application scenarios.

- **Selection approach for available communication systems**

After determining the relative priorities of the communication characteristics and functions, a decision-making method is required to avoid an exhaustive and impractical search among all available solution elements of the communication technologies and communication architectures. To select a suitable decision-making method, the following set of requirements has been determined based on the overall requirements of the system-level design framework:

Extensibility: The decision-making method shall allow the consideration of new solution elements that are eventually added in future.

Automation: The decision-making method shall be automatable. This is necessary in order to embed the method within the SoS configuration software and provide automatic selections or suggestions for non-expert users.

Convenience: The decision-making method shall consider the user priorities. In other words, the selection process depends principally on the preference of the users according to the requirements of the concerned application scenarios. However, non-expert users shall not be burdened with intensive analysis or manual comparison processes.

Different scales: The decision-making method shall be able to consider criteria of different scales. For instance, while some criteria can be described by cardinal numbers, other criteria can be specified as only ‘Yes’ or ‘No’ options.

Unique selections: The decision-making method shall be able to provide unambiguous selections. In other words, non-expert users shall not have to conduct additional analysis to select between available solution elements.

There are various decision-making methods in the literature [Tri00]. In this work, a selection of four common methods are considered for comparison and evaluation: analytic hierarchy process, ideal point method, outranking method, and cost-benefit analysis [Völ12]. In the analytic hierarchy process, each two criteria are compared together to assign respective priorities. Then, each two alternatives are compared together with re-

spect to each criterion. Finally, the available alternatives are ranked for the final selection. The ideal point method has various variants. In the compromise programming variant for instance, a performance matrix is created between the available alternatives and the criteria. The user has to define an ideal point for each criterion assuming that all criteria can have only monotonically increasing values. Then, the available alternatives are rated with respect to the defined ideal points. The final selection is the alternative of the best overall score. The outranking method has also various variants. In the ELECTRE[9] variant for instance, a performance matrix is created between the available alternatives and the determined criteria. The user has to define a threshold for each criterion. Then, each two alternatives are compared together with respect to each criterion and its threshold. The result is represented as a set of dominated and non-dominated alternatives. Finally, the set of non-dominated alternatives are considered for further filtration. In the cost-benefit analysis method, the available alternatives are listed against the criteria. The alternatives are then evaluated according to their fulfillment of all criteria. Table 20 presents an evaluation of the aforementioned decision-making methods according to the determined requirements.

Table 20: Evaluation of different decision-making methods from the literature using the determined set of requirements

Analyzed and evaluated decision-making methods	Requirements of decision-making methods				
	Extensibility	Automation	Convenience	Different scales	Unique selections
Analytic hierarchy process	Yes	Partial	Partial	Yes	Yes
Ideal point method	Yes	Yes	No	No	Yes
Outranking method	Yes	Partial	No	No	No
Cost-benefit analysis	Yes	Yes	Yes	Yes	Yes

The evaluation shows that the cost-benefit analysis exclusively fulfills all determined requirements of the decision-making methods. Thus, this particular decision-making method is adopted in this work to facilitate the selection of the communication systems. For instance, Table 21 shows an assessment of three exemplary communication tech-

[9] ELECTRE is an acronym that stands for *EL*imination *E*t *C*hoice *T*ranslating *RE*ality.

nologies with respect to five network characteristics based on the cost-benefit analysis method.

Table 21: Assessment of example communication technologies according to the cost-benefit analysis method (excerpt)

Network characteristics	Priority weight (%)	Communication technologies					
		Ethernet 10 Mbps		CAN Bus		InfiniBand	
		Fulfillment (%)	Partial assessment (%)	Fulfillment (%)	Partial assessment (%)	Fulfillment (%)	Partial assessment (%)
Bandwidth	29.35	80	23.5	20	5.87	100	29.4
Packet loss rate	11.59	10	1.16	100	11.6	100	11.6
Determinism	45.29	00	0.00	100	45.3	00	0.00
Latency	13.04	00	0.00	100	13.0	00	0.00
Segment length	0.730	100	0.73	50	0.37	30	0.22
Final assessment (%)		25.39		76.14		41.22	

The network characteristics are listed vertically and the available communication technologies are listed horizontally within the shown assessment matrix. Principally, the assessment uses the results of the priority analysis scheme presented earlier in this subsection. More specifically, each network characteristic is assigned to its priority weighting percentage that is calculated using the priority analysis scheme. The priority weighting percentages differ according to user preferences for the concerned application scenarios. Moreover, for each communication technology, the extent of fulfillment of each network characteristic is determined using an explicit requirement value given by the user. Consequently, all available communication technologies are assessed partially with respect to the individual network characteristics using Equation 6.

$$Partial\ assessment\ (\%) = \frac{Priority\ weight\ (\%)\ \times\ Fulfillment\ (\%)}{100}$$

Equation 6: Partial assessment of available communication technologies

The final assessment of each communication technology is calculated as the summation of all partial assessments according to Equation 7.

$$Final\ assessment_n\ (\%) = \sum_{m=1}^{M} Partial\ assessment_m\ (\%)$$

Equation 7: Final assessment of available communication technologies

Where:

n:	Designation of the communication technology n
m:	Designation of the network characteristic m
M:	Total number of network characteristics

Equation 8 presents a simple function that is used to find the best matching communication technology, which has the maximum final assessment.

$$Best\ matching\ CT = Max(Final\ assessment_1, Final\ assessment_2 \ldots, Final\ assessment_s)$$

Equation 8: A simple maximum function for determining the best matching communication technology (CT)

Where s is the number of available communication technologies.

A similar approach can be utilized for the communication architectures. That is, the user prioritizes the functions and services according to the concerned application scenarios to calculate priority weighting percentages. The available communication architectures are assessed according to the prioritized functions and services. Consequently, a simple function can be used to select the best matching communication architecture that has the maximum final assessment.

4.7 Phase 5 – System Models Development

The previous development phases and their tasks focused on the comprehensive analysis of the whole system and the development of selection approaches for its simulation and communication aspects. The outcomes shall be embedded in the SoS configuration software discussed in the next chapter to save efforts and time of non-expert system users and domain-specific developers. The last development phase is concerned with the actual creation of application-oriented system models based on the outcomes of the previous phases. The following subsection specifies the system configuration sequence to compose application-oriented system models for networked driving simulation.

4.7.1 Specify Configuration Sequence

The configuration sequence of system components is specified in this task. The system components have been classified into two main groups in the second development phase of the procedure model: simulation system group and communication system group. Basically, the components of the simulation system group are selected before the components of the communication system group during the configuration sequence. Figure 4-11 illustrates the determined configuration sequence of networked driving simulation as a system of systems.

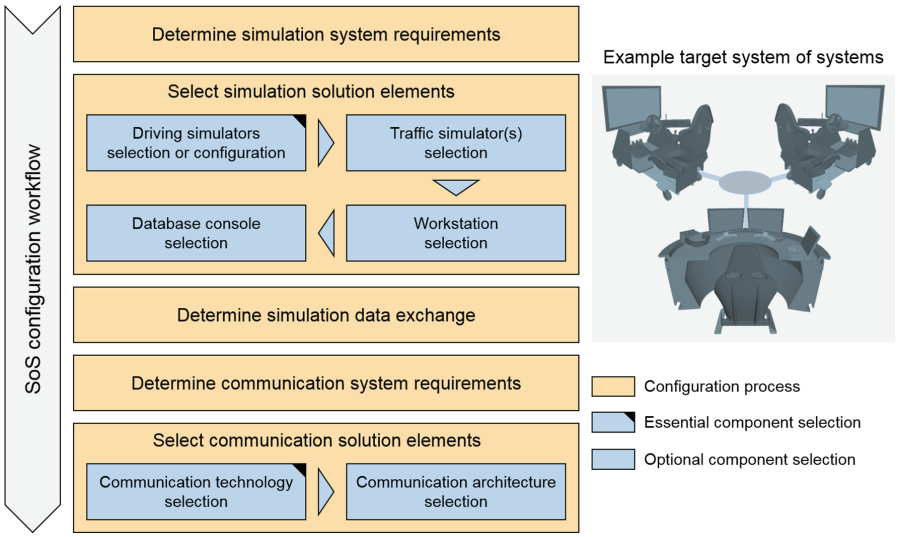


Figure 4-11: Configuration workflow for the networked driving simulation system

- **Determine simulation system requirements**

The application scenario is defined as a start point for the configuration process. This leads mainly to a decision about the number of participating driving simulators. Based on the overall application scenario, a purpose of use is defined for each participating driving simulator. This in turn leads to determining the respective driving tasks and responses [Neg07]. Consequently, the SoS configuration software determines the application class of each participating driving simulator. The specified application classes lead to the derivation of the requirements of the corresponding driving simulators [Neg07]. In addition to the driving simulators, the defined overall application scenario helps to decide about the eventual utilization of other system components, i.e., traffic simulators, workstations, and database consoles, and facilitates the derivation of their requirements.

- **Select simulation solution elements**

Driving simulators selection or configuration: There are two distinct ways to conclude the driving simulators that can fulfill the derived requirements. If the selection of

available driving simulators is preferred, the SoS configuration software finds the best matching driving simulator for each concerned application class from the entries of the corresponding database table. If configuring new driving simulators is preferred, the SoS configuration software finds the best matching simulator components for each concerned application class from the entries of the corresponding database table. In the latter case, subsidiary configuration processes are directed by the SoS configuration software to select the individual components while assuring their compatibility and consistency [Has14].

Traffic simulator(s) selection: The SoS configuration software finds the best matching traffic simulator for each concerned application class from the entries of the corresponding database table. Yet one traffic simulator is chosen typically in accordance with the most significant application class of the concerned application scenario. For example, if the application scenario includes an instructor driving simulator and a trainee driving simulator, the traffic simulator of the trainee application class is chosen. Nonetheless, the selection of traffic simulators can be eventually skipped as they are optional components for the operation of the networked driving simulation system.

Workstation selection: A workstation is suggested by the SoS configuration software based on the specifications of available workstations compared to the corresponding requirements. However, the selection of the workstation can be eventually skipped as its utilization is optional for the operation of the networked driving simulation system.

Database console selection: A database console is suggested by the SoS configuration software based on the specifications of available database consoles and the corresponding requirements. Nonetheless, this selection step can be eventually skipped as the utilization of a database console is optional for the operation of the networked driving simulation system.

- **Determine simulation data exchange**

This is an intermediate step before handling the communication system aspect. The system user determines the information exchange between the selected simulation solution elements. Each selected simulation solution element is specified with the simulation data it sends and/or receives. The number of the selected simulation system components, and hence, the determined simulation data exchange can affect the determination of some network requirements, such as the bandwidth.

- **Determine communication system requirements**

The requirements of the communication system are determined in this step. There are two sets of requirements in this regard: requirements of the communication technology and requirements of the eventually utilized communication architecture. The requirements of the communication technology specify the desired values for the concerned network characteristics. However, the values of some particular network characteristics can be initially determined only by assuming a worst-case scenario. For instance, the

required bandwidth depends on the number and size of the exchanged data packets, which in turn depend on the communication technology itself and its network protocol. Several network analyzers are available in the market [ORB+06a]. For example, Wireshark is a widely-known network analyzer that can be used for the determination of the required network bandwidth [ORB+06b]. Thereby, the network analyzer captures the data packets sent by each involved application and calculates the number of bits per second. The utilization of Ethernet with TCP/IP can be assumed as it involves almost the largest packet overhead among other communication technologies, and hence, can serve the purpose of worst-case scenario calculation. The protocol overhead is a part of the data packets used for handshaking, encoding/decoding, or addressing purposes. Consequently, a part of the theoretical bandwidth is consumed and cannot be used to transfer useful data. The required network bandwidth is the summation of the number of bits per second of all involved applications.

After determining the requirements of the communication technology, the network characteristics are prioritized in accordance with their significance to the concerned application scenario. Similarly, if the application scenario necessitates the utilization of a communication architecture, the corresponding requirements specify the support of particular functions and services. These are prioritized also in accordance with their significance to the concerned application scenario.

- **Select communication solution elements**

Communication technology selection: The SoS configuration software finds a best matching communication technology in accordance with the specified and prioritized requirements regarding the network characteristics. The result is an initial proposed communication technology with the best matching network characteristics.

Communication architecture selection: The SoS configuration software finds a best matching communication architecture in accordance with the specified and prioritized requirements regarding the communication functions and services. Nonetheless, this selection step can be eventually skipped as the utilization of a communication architecture is optional for the operation of the networked driving simulation system.

The system user is guided by the SoS configuration software through this configuration sequence. The system user can navigate back and forth arbitrarily along the sequence to modify the selections. The result of each step within the whole configuration process is affected by three particular factors:

1. The entries of available solution elements within the corresponding database table designed and built in the third phase of the procedure model.
2. The corresponding component selection approach as determined in the fourth phase of the procedure model.

3. The preferences of system user, who still can override the selections or suggestions provided by the SoS configuration software.

Before the construction and generation of a system model, the following subsection presents a crucial task that must be carried out to assure the eligibility of the initially selected communication technology.

4.7.2 Examine Network Behavior

In this task, the initially selected communication technology is examined to make sure that the eventually estimated requirements of network characteristics still can guarantee proper system operation. A network simulator can be utilized for this purpose as a supplementary software [WGG10]. Principally, network simulators are used to simulate the network behavior using mathematical formulas and models of network protocols. There are commercial network simulators, such as OPNET and QualNet. Other network simulators are available as free and open source packages, such as NS2, NS3, J-Sim, SSFNet, and OMNeT++.

The Automotive Network Diagnoser (ANDi) from Technica Engineering GmbH is utilized particularly in this work. It is a user-friendly test and simulation environment that supports various communication technologies and operation systems. Using this network simulator, users can construct a virtual network by creating nodes connected by network segments replicating a desired real network topology. The characteristics of the initially selected communication technology are provided along with the estimated amount of exchanged data packets to start a simulation. Based on the observed network behavior, the initial requirements of network characteristics are eventually modified. In this case, the communication technology selection step is recalled to find a more appropriate solution element. This process is repeated till the simulated network behavior exactly meets or comes very close to the requirements of the concerned application scenario regarding the communication technology. The following task discusses the final creation of a system model after finishing the configuration process and assuring the eligibility of the network characteristics.

4.7.3 Generate System Models

The objective of this task is to generate a system model using the SoS configuration software based on the results of the component selection process. Specifically, the system model can be considered as a comprehensive report that contains concise information about the system and the selected solution elements.

In general, model-based systems engineering relies on models to progress from the level of requirements to the level of system realization [ADK13]. That is, it follows a model-centric approach rather than a traditional document-centric approach. In practice, the SoS design and modeling cannot be carried out through a conventional system devel-

opment process [ADK13]. The complexity of the SoS design can be reduced considerably by following a model-centric approach. Agent-based modeling can provide a practical tool in this regard [ADK13]. It is a relatively new approach for modeling complex systems that are composed of further interacting systems. The agent-based modeling method considers the roles of all system components seen from a bottom-up perspective [BLM15]. The resultant agent-based model can describe the emerging system as a whole based on the roles and interactions of the constituent systems and building components. In particular, the application-oriented modeling of networked driving simulation in this work is concerned particularly with the system components, characteristics, and their relationships. The agent-based modeling technique is adopted in this task as it provides the required foundation principles for modeling system of systems [ADK13]. Comprehensive discussions about the principles of agent-based modeling are presented in References [RG1] and [ADM05]. However, there is no universal agreement in the literature on a precise definition for the term “agent”. In this work, an agent can be a constituent system or a building component as identified, described, and classified in the second development phase of the procedure model. The agent exchanges information and/or carries out specific tasks within the networked driving simulation system. A formal visual scheme is typically required to apply the agent-based modeling principles.

The Unified Modeling Language (UML) is a graphical visualization means that can facilitate system analysis and development [32]. Specifically, UML uses a set of well-defined elements that are independent of programming languages to construct various types of comprehensible UML diagrams. These diagrams can represent the system from different perspectives and provide different modeling levels. There are two main categories of UML diagrams: structure diagrams and behavior diagrams [DWT15, p. 35]. The UML structure diagrams include different types: class, object, package, deployment, components, composite structure design, and profile diagrams. Similarly, the UML behavior diagrams have different types: sequence, behavioral state machine, activity, communication, interaction overview, timing, behavioral state machine, protocol state machine, and use-case diagrams. The UML diagrams can be utilized during different phases of system development, such as analysis, design, and implementation. The same diagram type can be eventually used during the whole development process [DWT15, p. 35]. In this case, the diagram continues to evolve by including more details that can ultimately lead to writing software code or building physical models. In general, the simplicity and consistent notation of the UML diagrams make the utilization of UML very practical for analysts and developers of different fields. In particular, four UML diagrams can serve practically the agent-based modeling principles: sequence, state, activity, and class diagrams [Ber12]. The sequence diagrams can be used if the sequential interactions of the agents over time represent a significant design aspect. The state diagrams can be used if one is interested in the changes of the internal states of the agents. The activity diagrams are similar to the traditional flow charts. They can be used if it is important to analyze the activity progress of system components. The class dia-

grams consist typically of classes and different types of relationships, such as association, composition, and inheritance [Ber12]. They can be used when the focus is given to the system analysis and design and the relationships between its components. A comprehensive discussion about the elements of the UML class diagrams is presented in Reference [Rum16]. The UML class diagram concepts, notations, and elements are utilized in this task as basis for the construction of system models [Ber12]. The UML class diagrams can demonstrate the modeling level of detail, which conforms to the system-level design addressed in this work. In summary, the entire system model in this work consists of three different parts: UML class diagram, specification/requirement radar charts, and a list of exchanged simulation data.

Part I – UML class diagram: This part contains an UML class diagram that gives a holistic overview about the configured networked driving simulation system. The UML class diagram is divided into further two groups, which are called packages according to the UML notations [Rum16]. While the first package addresses the simulation system aspect, the second package addresses the communication system aspect. Each package encompasses various blocks. The parent blocks are classes that represent the utilized system components. The parent blocks within the simulation system package have inheritance relationships with an upper parent simulation system class. Similarly, the parent blocks within the communication system package have inheritance relationships with an upper parent communication system class. The child blocks are instances that represent the selected solution elements. The instances have abstraction relationships to the corresponding classes. Each block within the class diagram is typically composed of three compartments stacked vertically. The top compartment includes the name. The middle compartment lists the significant specifications and characteristics. The bottom compartment lists the main operations performed by the system component or the solution element.

Part II – Specification/requirement radar charts: This part consists of two sections containing several radar charts. Generally, the radar chart is a demonstration method for the visualization of data sets that include multiple quantitative variables. These variables are represented on axes that start from an origin point [Wil05]. Particularly, the first section of this part contains radar charts illustrating the deviation between the specifications of each selected simulation solution element and the corresponding application requirements. In a similar way, the second section contains radar charts illustrating the percentage or binary fulfillment of requirements by the corresponding specifications of each selected communication solution element. This part gives an overview of the eligibility of each selected solution element with respect to the concerned application scenario. The specification/requirements radar charts provide users with a visual demonstration of the eligibility extent of the selected solution elements. Hence, users can easily decide whether to proceed or to navigate back to select other alternatives.

Part III – List of exchanged simulation data: This part contains a list of the simulation data that each selected solution element sends and/or receives. The simulation data

are characterized by different attributes, such as the sender, unit, and type. This list of exchanged simulation data can be used during the preparation of the constituent systems and building components for system realization. Moreover, it can be used as basis if a communication architecture is utilized to particularly provide a data distribution management service [SZ14]. The Extensible Markup Language (XML) is chosen for the construction of this part. The XML defines a set of rules to encode the information in an easy format that can be understood by humans as well as software programs. Furthermore, the hierarchical structure supported by the XML format makes it convenient to trace and find the concerned information. A comprehensive discussion about the XML and its rules is presented in Reference [Ram02].

The discussed three parts of the system model are constructed automatically by the SoS configuration software according to the results of the configuration process. The user can still navigate back to alter particular selections when necessary based on the information illustrated through the created system model. Finally, the user can save the created system model and/or print it to begin with system realization. The generated system model is used also to communicate information between the domain-specific experts and users about the system that shall be built. Three system models are shown together with example application scenarios in Chapter 5.

The presented development phases and tasks can be specified and eventually modified by system developers, who have expertise in system engineering and system of systems engineering. Moreover, rigorous experiences in the field of driving simulation in general and networked driving simulation in particular are necessary. However, the outcomes of these development phases and tasks are embedded in the accompanying SoS configuration software in order to enable domain-specific developers and non-expert users to easily design and generate application-oriented system models of networked driving simulation. The following chapter presents the underlying design concepts of the SoS configuration software. In addition, validation examples are presented to acknowledge the usability of the developed framework for the design and generation of application-oriented system models of networked driving simulation.

5 Implementation Prototype and Validation

The previous chapter presented the procedure model of the design framework. The first four development phases within this procedure model focus on the comprehensive analysis of the whole system and the development of configuration approaches for its simulation and communication aspects. Yet the outcomes of these development phases shall be embedded in a system of systems configuration software in order to save efforts and time of non-expert system users and domain-specific developers. The last development phase addresses mainly the actual creation of application-oriented system models based on the outcomes of the preceding development phases. The creation of the system models is carried out by the system of systems configuration software. This chapter presents an implementation prototype of the system of systems configuration software and a comprehensive validation of the developed design framework. Specifically, Section 5.1 presents the main operations, the architecture, and an implementation prototype of the system of systems configuration software. As a usability validation, Section 5.2 provides the generation of system models for three example application scenarios for networked driving simulation. Finally, Section 5.3 presents further validation for the overall design framework based on the set of requirements determined in Chapter 2.

5.1 SoS Configuration Software Development

The procedure model represents the first essential part of the design framework. However, users and domain-specific developers don't have to study the whole procedure model in order to create application-oriented system models for networked driving simulation. Therefore, the SoS configuration software is developed to embed the approaches of the procedure model and represents thereby the second essential part of the design framework. The concept and the main operations of the SoS configuration software are described in Subsection 5.1.1. The architecture of the SoS configuration software is described in Subsection 5.1.2. Subsection 5.1.3 presents an implementation prototype of the SoS configuration software and the design of its graphical user interfaces.

5.1.1 Main Operations

In general, configuration management is a discipline that addresses not only technical but also administrative aspects to handle system rearrangement [Wat09, pp. 1–12]. In the technical context particularly, configuration management focuses on maintaining the functionality and performance so that the configured system conforms to a defined set of requirements, design constraints, and/or operation conditions. The general concepts of configuration management are adopted in this work to determine the necessary operations of the SoS configuration software [Has03]. The configuration management in this work is applied on a complex system that consists of further independent systems and components. The focus is given to the system level design and configuration of the net-

worked driving simulation as a system of systems. That is, the internal operation of the constituent systems and building components is not considered.

The main objective of the SoS configuration software is to enable users and domain-specific developers to design and rearrange a model for a networked driving simulation system in accordance with the requirements of the concerned application scenarios. Comprehensive system analysis and deep technical knowledge are not required to model the system using the SoS configuration software. This particular feature represents the main advantage of the SoS configuration software presented in this work.

The SoS configuration software uses and extends the principles and approaches adopted for an analogous software discussed in References [Has14] and [HGA+15]. The general, fundamental requirements of the SoS configuration software were discussed in Subsection 2.6.2. However, a set of functional and non-functional requirements are determined and discussed in this subsection to provide more understanding of the operations and benefits of the SoS configuration software. This set of requirements was derived based on the requirements of the whole design framework. It is used as basis to implement a prototype of the SoS configuration software.

Functional requirements: They describe the main operations carried out by the SoS configuration software.

1. The SoS configuration software shall have access to the system databases designed in the third phase of the procedure model. Thereby, system users shall be able to view the entries of the available solution elements and their properties. Moreover, system users shall be able to add entries of new solution elements and modify or delete existing entries of solution elements through the SoS configuration software.

2. The SoS configuration software shall ensure that the selection of system components conforms to the determined requirements of the concerned application scenarios to the best possible extent. This feature shall be realized by utilizing the configuration methods introduced in the fourth development phase of the procedure model.

3. The SoS configuration software shall guide system users through the configuration process specified in the last phase of the procedure model. Moreover, the SoS configuration software shall allow system users to navigate back and forth through the configuration process when necessary.

4. The SoS configuration software shall ensure the transparency of the whole configuration process. That is, system users shall be provided with a sort of graphical demonstration that compares the specifications and requirements of each suggested system component during the configuration process.

5. The SoS configuration software shall allow system users to override the suggestions regarding the selections of system components during the configuration process.

6. The SoS configuration software shall create system models in accordance with the selected system components. These system models shall follow the design specified in the last phase of the procedure model.

7. The SoS configuration software shall allow system users to save and/or print the created system models.

Non-functional requirements: They specify particular demands and constraints for the SoS configuration software. In other words, they are concerned with the properties of the SoS configuration software as a whole package while disregarding the individual functionalities. The non-functional requirements play a considerable role while determining the implementation concepts and architecture of the SoS configuration software.

1. The SoS configuration software shall have a user-friendly graphical interface to ensure its convenient usability. System users shall not be overwhelmed by the underlying configuration methods or the structure of the associated system databases.

2. The SoS configuration software shall have a modular and extendable structure. The eventual future modification of the software shall be provided with minimal programming efforts. Similarly, the associated system databases shall be extendable. This feature permits the insertion of entries of future solution elements. Moreover, columns for further specifications or characteristics of available solution elements can be easily added.

3. The SoS configuration software shall be platform-independent. Specifically, it shall be compiled in an executable file that is independent of the operating system.

The architecture of the SoS configuration software is presented in the next subsection based on the defined set of functional and non-functional requirements.

5.1.2 Adopted Architecture

The modularity and extensibility are two crucial requirements of the SoS configuration software as discussed in Chapter 2 and emphasized in the previous subsection. The separation of concerns principle is adopted in this work in response to these particular requirements [VAC+11, pp. 118–139]. The separation of concerns is a well-known software engineering principle that is used for software architecting. It can be applied by the identification of concrete parts of the software system that are responsible for specific aspects or task. These concrete parts are encapsulated as separate building blocks within the software system. The main purpose is to break down the complexity of the software system into individual manageable parts. The resulting modular and extensible structure of the software system is the most significant benefit of the separation of concerns principle in software architecting [VAC+11, p. 127].

The separation of concerns principle is realized in this work by applying the Model-View-Controller (MVC) approach [Bou92]. This approach considers the software sys-

tem in three distinct parts: model, view, and controller. The model part deals with the information or data handled by the software system. The view part deals with the presentation of the application to the users. The controller part deals with the interaction between the users and the software system to carry out the application objectives.

Applying the MVC approach for the development of the SoS configuration software results in the following three modules:

1. **SoS configuration software model:** This module includes the associated system databases as well as the configuration methods addressed within the procedure model. The system databases include the specifications of the available solution elements. The configuration methods use these specifications together with the provided application requirements to find out the best possible matching solution elements for the concerned application scenarios.
2. **SoS configuration software view:** This module includes the graphical user interface of the SoS configuration software. System users handles the SoS configuration software only through this module. Specifically, system users can perform operations on the associated system databases and go through a configuration process to create system models for networked driving simulation.
3. **SoS configuration software controller:** This module includes callback functions that connect the view and model parts together. Specifically, when system users perform actions through the view module, the controller module invokes corresponding operations on the associated databases or calls corresponding configuration methods.

Figure 5-1 illustrates the three modules of the SoS configuration software and their interactions based on the MVC approach.

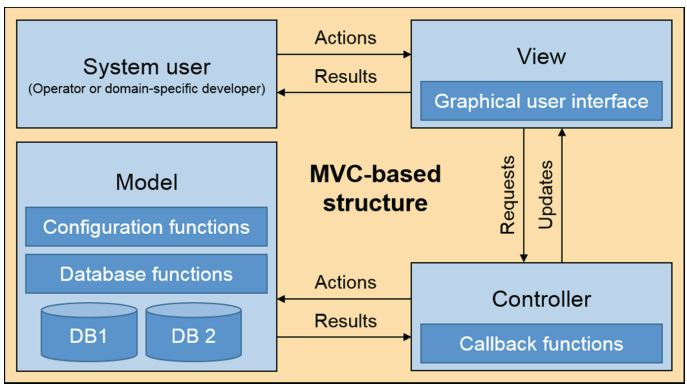


Figure 5-1: MVC modules of the SoS configuration software and their interactions

The following subsection presents an implementation prototype of the SoS configuration software.

5.1.3 Implementation Prototype

In this work, a prototype of the SoS configuration software has been implemented. Two software tools were utilized to implement this prototype: Microsoft Office Excel and MATLAB. This subsection describes the fundamental components of the SoS configuration software.

- **Associated system databases**

The system databases associated with the SoS configuration software belong to the model part of the MVC approach as illustrated Figure 5-1 [Bou92]. These system databases were implemented in accordance with the design presented in the third phase of the procedure model. The structure of these system databases follows the relational model, where the data is registered in tables associated with particular relations. The relational database model provides a very good level of simplicity and ease of data handling. The data can be accessed data without navigating a rigid pathway through a tree or hierarchy. For the prototype version of the SoS configuration software, the system databases were implemented using Microsoft Office Excel. Database tables can be created easily with Microsoft Office Excel. Furthermore, it is convenient to access the created tables from different software applications.

- **Configuration functions**

The functions of the SoS configuration software belong to the model part of the MVC approach as illustrated in Figure 5-2 [Bou92]. These functions were implemented in accordance with the approaches presented in the fourth phase of the procedure model. For the prototype version of the SoS configuration software, the MATLAB was used to realize the configuration approaches of the procedure model and carry out operations on the associated system databases. The matrix-based MATLAB language facilitates the implementation and debugging of complex functions.

- **Graphical user interface**

The graphical user interface of the SoS configuration software belongs to the view part of the MVC approach as illustrated in Figure 5-2 [Bou92]. It was implemented using the MATLAB GUIDE (GUI development environment) component. Using the MATLAB GUIDE Layout Editor, graphical user interfaces can be intuitively designed. Furthermore, the editor generates a MATLAB code containing callback functions automatically after the construction of the user interface. These callback functions can be easily modified to program the behavior of the individual user interface elements. That is, the user interface elements can invoke particular configuration functions through their callback functions. This role represents the controller part of the MVC approach [Bou92]. Figure 5-2 shows a screen shot for the start window of the SoS configuration software prototype.

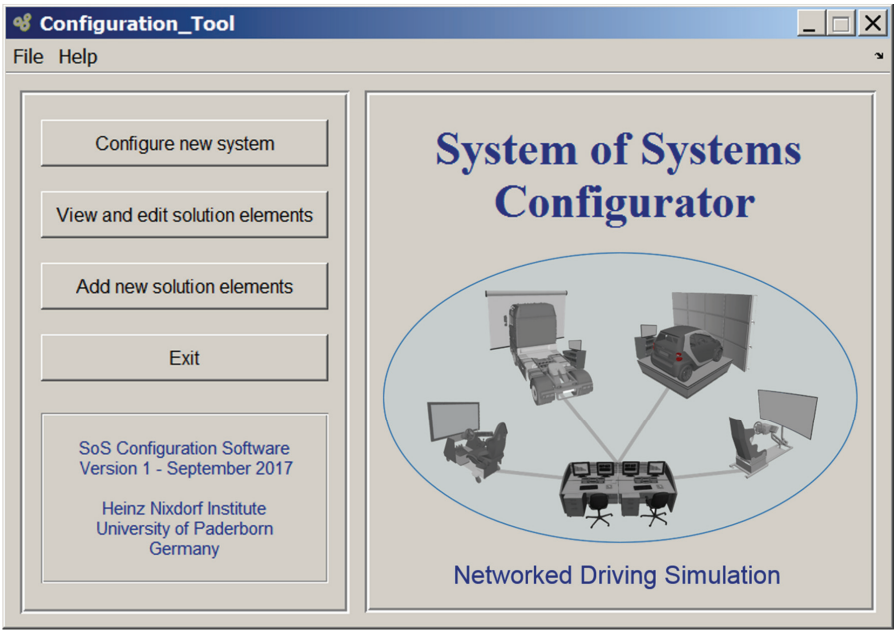


Figure 5-2: Start window of the SoS configuration software prototype

More illustration of the SoS configuration software prototype is presented in the appendix. As a usability validation, the following section explains the creation of system models for three application scenarios for networked driving simulation.

5.2 Example Application Scenarios and System Models

Three example multi-interactive application scenarios are presented in this section in order to validate the design method and its associated SoS configuration software. Each validation example begins with a motivation and a concrete definition for the application scenario. The application scenario and the requirements are formalized based on the specification technique adopted in the **first phase of the procedure model**. In accordance with the specific application requirements, the major target is to compose a networked driving simulation system from the components analyzed in the **second phase of the procedure model** and registered in the databases designed in the **third phase of the procedure model**. To that end, a configuration process is conducted based on configuration approaches developed in the **fourth phase of the procedure model** and embedded in the **SoS configuration software**. The configuration process results in a concretized system model for networked driving simulation. Specifically, this system model is constructed based on the modeling techniques adopted in the **fifth phase of the procedure model**. The system model is generated using the developed **SoS configuration software**. Finally, a platform of networked driving simulation is built in accord-

ance with the generated system model. The environment, application scenario, functions, and active structure partial models of the validation examples are reduced versions of the example partial models presented in Section 4.3.

In addition to the validation purpose, the presented application scenarios deliver an example line of thoughts to show non-expert users and domain-specific developers how to utilize the developed method and the SoS configuration software for creating application-oriented system models. Nonetheless, the actual application requirements and the user's preferences play considerable roles during the configuration process. Moreover, the entries of available solution elements within the system databases affect the outcomes of the configuration process. Hence, the example system models created in this section shall not be considered as absolute standard solutions for the presented application scenarios.

5.2.1 Scenario 1 – Advanced Training with ADAS

Motivation and scenario definition: Although ADAS are designed to reduce the burden on drivers, the complexity of user interface grows with increasing the number of automated functionalities. This characteristic demands some of driver's attention and introduces much cognitive load. Therefore, conventional training with driving simulators must be adapted for more immersion and a capability for interactive supervision and instruction. The application scenario of this validation example is to perform multi-interactive, supervised training sessions to learn ADAS functions and avoid the eventual overestimation of their capabilities.

Configuration process: A simulation environment consisting mainly of two driving simulators is defined based on the described application scenario. While one driving simulator is used by a trainee, the other driving simulator is used by a training instructor. The trainee is introduced to various ADAS functions, such as, e.g., blind spot and congestion assistance systems, while driving through an unfamiliar road network [LH13], [MZ04]. The trainee activates and handles the settings of the ADAS functions and responds to the dashboard indicators. This purpose of use falls within the driving simulator application class 4c that addresses knowledge-based responses and secondary driving tasks. The training instructor has a simple navigation driving task. The aim is to drive interactively within the same virtual environment to observe the traffic situation and to eventually introduce unexpected driving maneuvers. The response of the training instructor is not of concern as a matter of course. The purpose of use can be carried out by the driving simulator application classes 3a, 3b, and 3c that address different driving responses with navigation driving tasks. The application class 3c is chosen for convenience in this validation example. The determined application classes of the two participating driving simulators are used together as the first actual input to the SoS configuration software. Among the available driving simulators, the SoS configuration software

suggests two particular driving simulators that best fulfill the requirements of the application classes 4c and 3c.

In this application scenario, no traffic simulator is chosen as the trainee has to react only to the maneuvers introduced by the training instructor. No workstation is required for this platform as the training instructor already participates interactively within the virtual traffic scenario, and hence, there is no need for external monitoring or control processes. A database console can be used in this application scenario to capture, save, and replay the simulation data for analysis and conducting after-action reviews.

A lot of data must be exchanged between the participating simulators. Not only position and orientation data of the simulated vehicles are exchanged through the network, but also additional information for better immersion and engaging training sessions, such as, e.g., front and rear lamps state, turning indicator lamps state, and front wheels orientation. A bandwidth of 10 Mbps is required initially according to a worst-case analysis and calculation of the number of exchanged data packets [MS13], [ORB+06a], [ORB+06b]. Lower priority levels are assigned to the other network characteristics, such as, real-time data delivery or data loss rate. Accordingly, a 10 Mbps Ethernet with User Datagram Protocol is suggested by the SoS configuration software for this application scenario. The simulated network behavior using the ANDi software confirmed the eligibility of this selection. No standard architecture for networked simulation is selected as networked simulation management functions are not required for this application scenario.

Generated system model: A system model is created by the SoS configuration software in accordance with the application requirements and the results of the conducted configuration process. Figure 5-3 shows a simplified version of the simulation package of the UML class diagram that contains representations of the selected simulation system components.

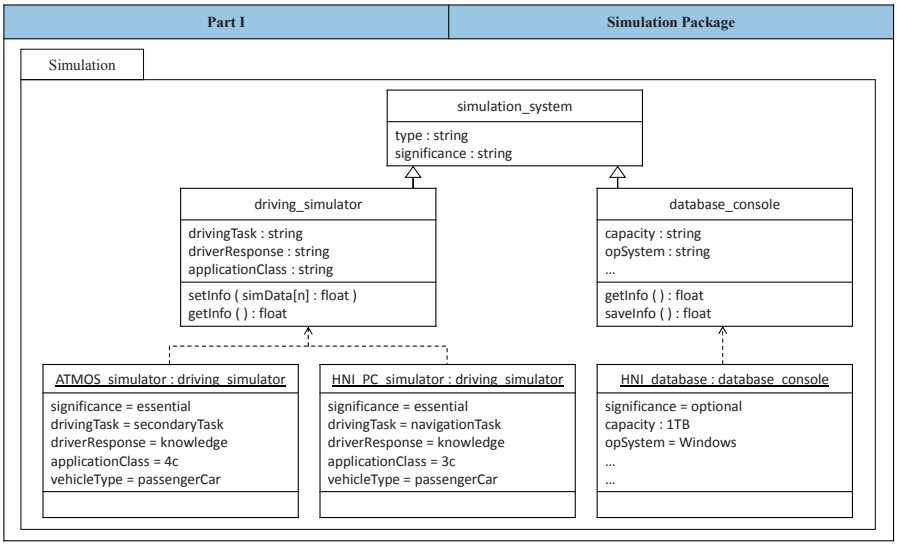


Figure 5-3: UML class diagram of the selected simulation system components – package 1

The simulation package includes a main class named: *simulation_system*. Two classes inherit the *simulation_system* class: *driving_simulator* and *database_console*. The driving_simulator class has two instances representing the two selected driving simulators: *ATMOS_simulator* and *HNI_PC_simulator*. The database_console class has one instance representing the selected database console: *HNI_database*. Figure 5-4 shows a simplified version of the communication package of the UML class diagram that contains representations of the selected communication system components.

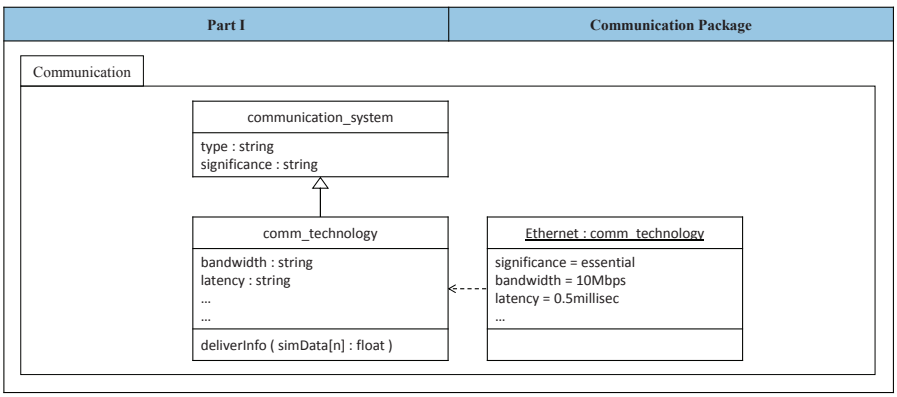


Figure 5-4: UML class diagram of the selected communication system components – package 2

The communication package includes a main class named: *communication_system*. There is only one class that inherits the *communication_system* class: *comm_technology*. It has one instance representing the selected communication technology. Figure 5-5 shows the first section of the second part of the generated system model. This section contains the specification/requirement radar charts of the two selected driving simulators. The Appendix provides tables that facilitate the interpretation of the features and fidelity levels within the radar charts. Moreover, these tables can be displayed by the SoS configuration software upon user's request.

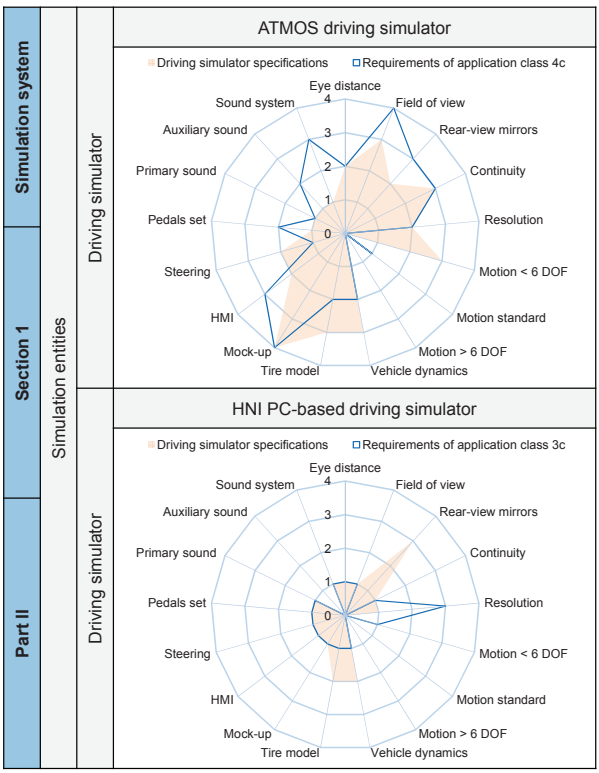


Figure 5-5: Specification/requirement radar charts of the selected driving simulators

Figure 5-6 shows the second section of the second part of the generated system model. This section contains the specification/requirement radar charts of the selected communication system components. In this application scenario, a communication technology was selected as a matter of course, whereas the utilization of a communication architecture was excluded. The radar chart illustrates the percentage fulfillment of requirements by the corresponding specifications of the selected solution element. Section 4.5 provides descriptions for the network characteristics shown within the radar chart. Moreover, the descriptions of these network characteristics can be displayed by the SoS configuration software upon user's request.

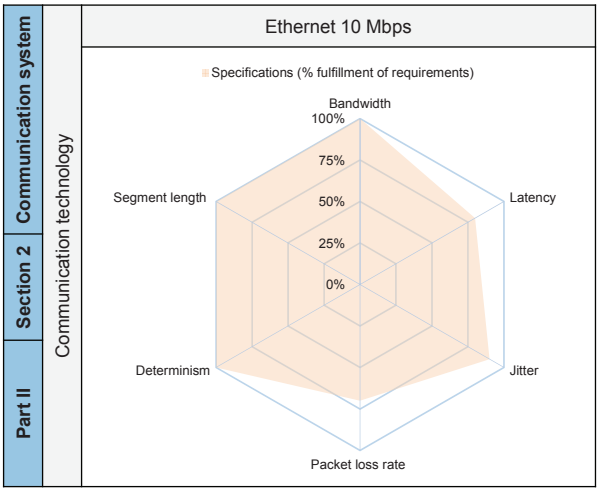


Figure 5-6: Specification/requirement radar chart of the selected communication system component

Established platform: A platform of networked driving simulation was built in accordance with the generated system model for the first example application scenario. Figure 5-7 shows the two networked driving simulators of the developed platform.

Figure 5-7: Developed platform of networked driving simulation for the first example application scenario

The HNI PC-based driving simulator shown at the left side of Figure 5-7 has no motion platform. This driving simulator has a commercial wheel-transmission-pedals set and a simple driving seat. It utilizes a 60-inch high-definition screen. The HNI PC-based driving simulator is operated by a software package developed with MATLAB/Simulink at the Heinz Nixdorf Institute [AHG+14a]. The ATMOS driving simulator shown at the right side of Figure 5-7 has a complex motion platform consisting of two dynamical components. The motion platform provides five DOF to fully simulate vehicle lateral and longitudinal accelerations. The ATMOS driving simulator has an eight-channel cylindrical projection system. It is powered by eight LCD-projectors to cover a horizontal field of view of 240 degrees. In addition, three small displays are used to simulate the side and rear mirror views. The ATMOS driving simulator is operated by a software package developed by dSPACE. The following subsection presents a second validation example.

5.2.2 Scenario 2 – Demonstration of Autonomous Driving

Motivation and scenario definition: Demonstrating the capabilities of autonomous driving contributes significantly to raising the awareness about its benefits and attracting more customers [Pla07]. Automotive manufacturers organize demonstration events to show the magnificent features and enable broader audience to be familiar with the new technologies. System demonstration with the help of drives on test roads may be permitted and it can deliver impressive experience to potential customers. However, driving with other traffic participants is typically not permitted to date. Networked driving simulation can complement the demonstration purpose by adding an interactive factor to the simulated traffic environment. This characteristic can deliver a comprehensive experience with the capabilities as well as the limitations of the autonomous driving system. The application scenario of this validation example is to interactively demonstrate different autonomous driving technologies to customers.

Configuration process: A simulation environment consisting mainly of two driving simulators is defined based on the described application scenario. One driving simulator is equipped with a simulation model for an autonomous driving system. It is used by customers to interactively experience and test the system in a safe simulation environment. It is assumed that the customers are introduced theoretically to the autonomous driving system in advance. That is, they know about the features of the system and how to handle its settings. This purpose of use falls within the application class 4b that addresses rule-based responses and secondary driving tasks. The second driving simulator is used by a marketing representative, who has a simple navigation driving task. The aim is to drive interactively within the same virtual environment to subject the autonomous driving system to different traffic conditions, such as, e.g., car following or emergency brake scenarios. The response of the marketing representative is not of concern as a matter of course. This purpose of use falls within the application classes 3a, 3b, and 3c that address different driving responses with navigation driving tasks. The application

class 3c is chosen for convenience in this validation example. Similar to the previous example application scenario, the determined application classes of the two participating driving simulators are used together as the first actual input to the SoS configuration software. Among the available driving simulators, the SoS configuration software suggests two particular driving simulators that best fulfill the requirements of the application classes 4b and 3c.

For this application scenario, no traffic simulator is chosen so that customers are not overwhelmed or confused through the interaction with other programmed traffic participants at this system introductory level. No workstation is required for this platform as it used for simple demonstration purposes in exhibitions. Similarly, no database console is used as typically no analysis process is required after the demonstration sessions.

Only position and orientation data of the simulated vehicles must be exchanged for the simple demonstration purpose of this application scenario. No high priority is given to the bandwidth of the communication system. A bandwidth of 1 Mbps (Megabit per second) is required initially according to a worst-case analysis and calculation of the number of exchanged data packets [MS13], [ORB+06a], [ORB+06b]. The simulation model of the autonomous driving system incorporates sub-models of various sensors [AHB+15]. Another sub-model takes decisions for rapid actions that influence the vehicle dynamics, such as, e.g., acceleration, braking, and steering. Hence, deterministic data exchange between the driving simulators is essential for reliable system operation. Other network characteristics are less relevant, such as, e.g., secure data exchange, or length of transmission medium. Accordingly, the CAN bus technology is suggested by the SoS configuration software for this application scenario as a deterministic communication technology [Vos15]. If the utilization of the CAN bus technology is expensive and relatively complex as it requires special network cards, the user can alternatively go for the FireWire (IEEE 1394). The FireWire (IEEE 1394) is a serial bus for high-speed communication that can be utilized as a more feasible alternative [And98]. Although the IEEE-1394 standard is used typically to connect computers to peripherals, such as, e.g., digital cameras and external hard drives, it can be used to carry network data as well. Using the FireWire technology with the Internet Protocol provides a very near deterministic data delivery [And98]. The simulated network behavior using the ANDi software confirmed the eligibility of this alternative selection. Similar to the previous validation example, no standard architecture for networked simulation is used as networked simulation management functions are not required in this application scenario.

Generated system model: A system model is created by the SoS configuration software in accordance with the application requirements and the results of the conducted configuration process. Figure 5-8 shows a simplified version of the simulation package of the UML class diagram that contains representations of the selected simulation system components.

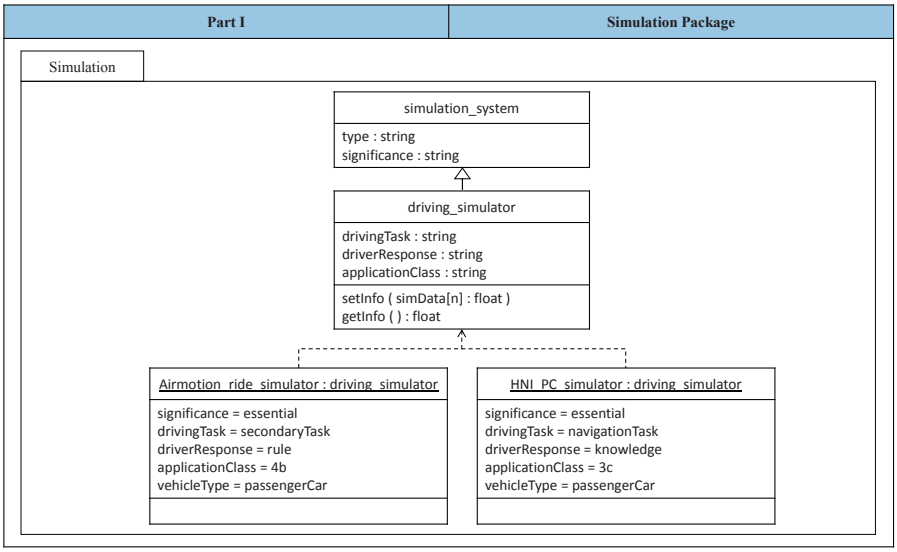


Figure 5-8: UML class diagram of the selected simulation system components – package 1

The simulation package includes a main class named: *simulation_system*. There is only one class that inherits the *simulation_system* class: *driving_simulator*. The *driving_simulator* class has two instances representing the two selected driving simulators: *Airmotion_ride_simulator* and *HNI_PC_simulator*. Figure 5-9 shows a simplified version of the communication package of the UML class diagram that contains representations of the selected communication system components.

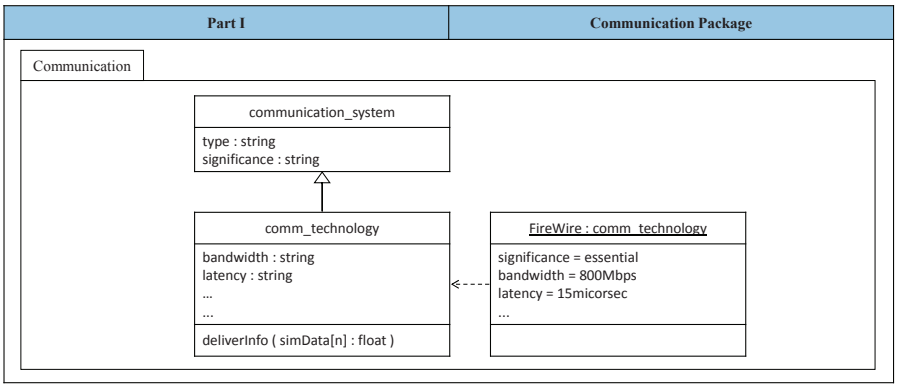


Figure 5-9: UML class diagram of the selected communication system components – package 2

The communication package includes a main class named: communication_system. There is only one class that inherits the *communication_system* class:

comm_technology. It has one instance representing the selected communication technology. Figure 5-10 shows the first section of the second part of the generated system model. This section contains the specification/requirement radar charts of the two selected driving simulators. The Appendix provides tables that facilitate the interpretation of the features and fidelity levels within the radar charts. Moreover, these tables can be displayed by the SoS configuration software upon user's request.

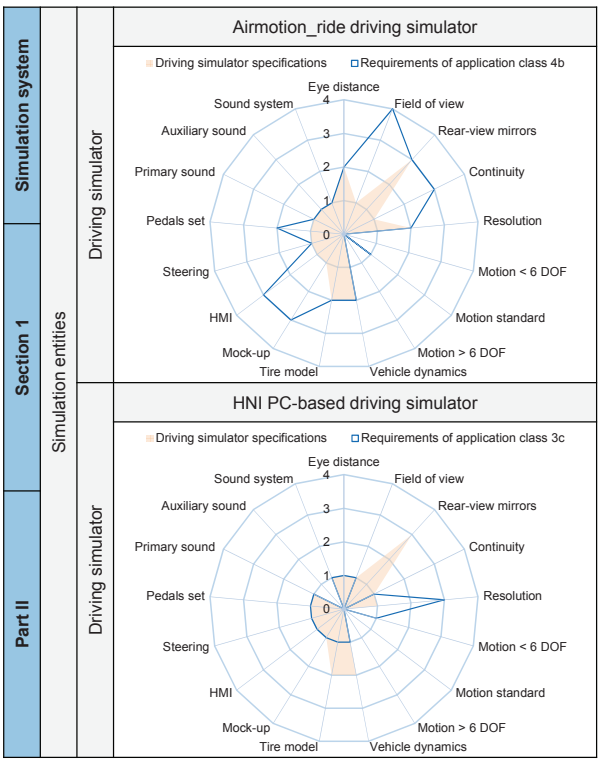


Figure 5-10: Specification/requirement radar charts of the selected driving simulators

Figure 5-11 shows the second section of the second part of the generated system model. This section contains the specification/requirement radar charts of the selected communication system components. Similar to the previous application scenario, a communication technology was selected as a matter of course, however, the utilization of a communication architecture was excluded. The radar chart illustrates the percentage fulfillment of requirements by the corresponding specifications of the selected solution element. Section 4.5 provides descriptions for the network characteristics shown within the radar chart. Moreover, the descriptions of these network characteristics can be displayed by the SoS configuration software upon user's request.

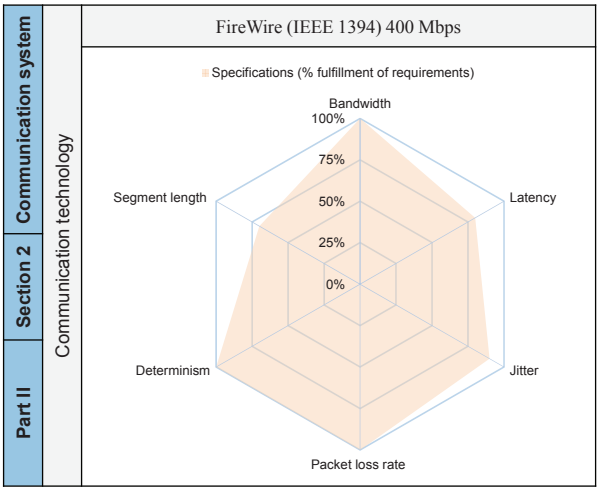


Figure 5-11: Specification/requirement radar chart of the selected communication system component

Established platform: A platform of networked driving simulation was built in accordance with the generated system model for the second example application scenario. Figure 5-12 shows the two networked driving simulators of the developed platform.

Figure 5-12: Developed platform of networked driving simulation for the second example application scenario

The HNI PC-based driving simulator is utilized in the established platform as shown at the left side of Figure 5-12. It is the same simple driving simulator variant of the previous example application scenario. The Airmotion_ride driving simulator shown at the right side of Figure 5-12 is utilized in the established platform. It has a pneumatic motion platform consisting of an inverted hexapod system. A simple motion controller regulates the extension and contraction of six pneumatic elastic tubes based on the position and orientation of the simulated vehicle. Similar to the HNI PC-based driving simulator, the Airmotion_ride driving simulator has a 60-inch high-definition screen and a low-cost commercial wheel-transmission-pedals set. It is operated by a software package developed with MATLAB/Simulink at the Heinz Nixdorf Institute [AHG+14a].

5.2.3 Scenario 3 – Analysis of Advanced Traffic Strategies

Motivation and scenario definition: Promising applications are emerging with the utilization of Vehicle-to-Infrastructure (V2I) and Vehicle-to-Vehicle (V2V) communication technologies. The common target of these technologies is to improve traffic efficiency while reducing the probability of collisions [WTB15]. Yet analyzing different strategies is important to compare the efficiency and benefits, as well as to early detect the possible shortcomings of these technologies. Microscopic and macroscopic traffic modeling and simulation are effective tools to study the causes of traffic problems in general and to evaluate the solutions of potential traffic strategies in particular [Bar10]. However, human drivers still represent an important factor. They may be assisted by various connected systems offering different information and service levels [SBK13]. Drivers may take different decisions based on the received information, such as, e.g., changing the route. Adding the factor of human drivers to the simulation environment helps to conduct analysis in a multi-interactive traffic environment, and hence, to deliver more substantial results. The application scenario of this validation example is to analyze different traffic strategies while taking the human driver factor into consideration.

Configuration process: A simulation environment consisting mainly of two driving simulators is defined based on the described application scenario. Both driving simulators are used by test persons, who are familiar with the simulated road network and the features of the addressed connected vehicle technology. The test persons have to plan the route and drive from a start to a target location. They can change the planned route based on the received information about the current traffic situation. This purpose of use falls within the application class 3b that addresses rule-based responses and navigation driving tasks. Similar to the previous example application scenarios, the determined application classes of the two participating driving simulators are used together as the first actual input to the SoS configuration software. Among the available driving simulators, the SoS configuration software suggests a particular driving simulator that best fulfills the requirements of application class 3b.

For this validation example, a traffic simulator is chosen according the application class 3b as suggested by the SoS configuration software. The utilization of a traffic simulator is necessary in order to provide suitable traffic complexity and density for this particular application scenario. The utilized traffic simulator allows for changing the traffic density and defining the behavior of the individual traffic participants. More information about this particular traffic simulator is presented in Reference [AHB+16]. A workstation is required for this scenario to perform monitoring and control operations. Moreover, a database console is necessary to capture, save, and replay the simulation data for analysis and conducting after-action reviews.

In this application scenario, a considerable amount of data must be exchanged through the communication system. Not only position and orientation data of simulator vehicles are exchanged, but also those of each programmed traffic participant. Moreover, data messages of the addressed connected vehicle technology are exchanged between the simulator vehicles. Hence, this application scenario is concerned particularly with the bandwidth for data exchange. The other characteristics of the communication technology have lower priority levels, such as, real-time data delivery or data loss rate. A bandwidth of 1 Gbps is required initially based on a first worst-case analysis and calculation of the number of exchanged data packets [MS13], [ORB+06a], [ORB+06b]. Accordingly, a 1 Gbps Ethernet with User Datagram Protocol is suggested by the SoS configuration software for this application scenario. The simulated network behavior using the ANDi software confirmed the eligibility of this selection.

More driving simulators may be added to the system to increase the complexity of traffic scenarios. In some scenarios, one of the driving simulators may be used to represent a special-purpose vehicle, such as, e.g., an ambulance or an emergency vehicle. This special-purpose simulated vehicle may have different requirements regarding the type of exchanged data. Consequently, it may be necessary to declare the generated and required data separately for each traffic participant. Therefore, a standard architecture for networked simulation is required unlike the previous validation examples. HLA standard is suggested by the SoS configuration software for this application scenario, especially, due to the provided declaration management function [AGG+17].

Generated system model: A system model is created by the SoS configuration software in accordance with the application requirements and the results of the conducted configuration process. Figure 5-13 shows a simplified version of the simulation package of the UML class diagram that contains representations of the selected simulation system components.

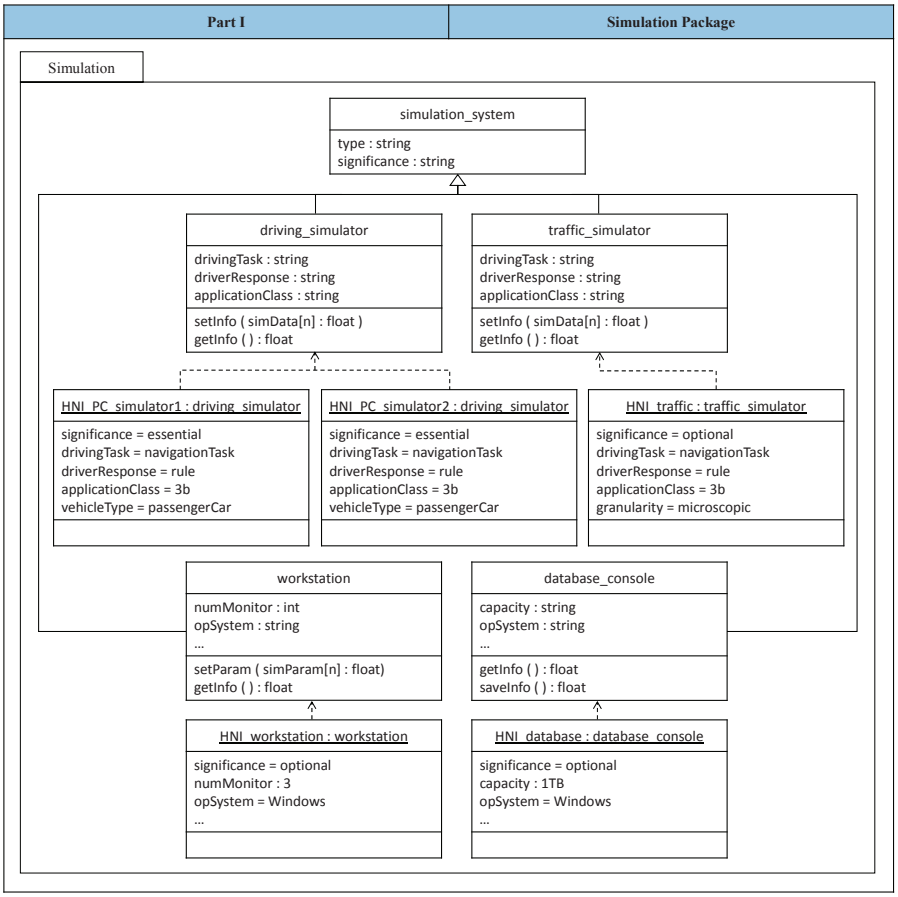


Figure 5-13: UML class diagram of the selected simulation system components – package 1

The simulation package includes a main class named: simulation_system. Four classes inherit the *simulation_system* class: *driving_simulator*, *traffic_simulator*, *workstation*, and *database_console*. The *driving_simulator* class has two instances representing the two selected driving simulators: *HNI_PC_simulator1* and *HNI_PC_simulator2*. The *traffic_simulator* class has one instance representing the selected traffic simulator: *HNI_traffic*. Similarly, the *workstation* and the *database_console* classes have instances representing the respective selected solution elements. Figure 5-14 shows a simplified version of the communication package of the UML class diagram containing the selected communication system components.

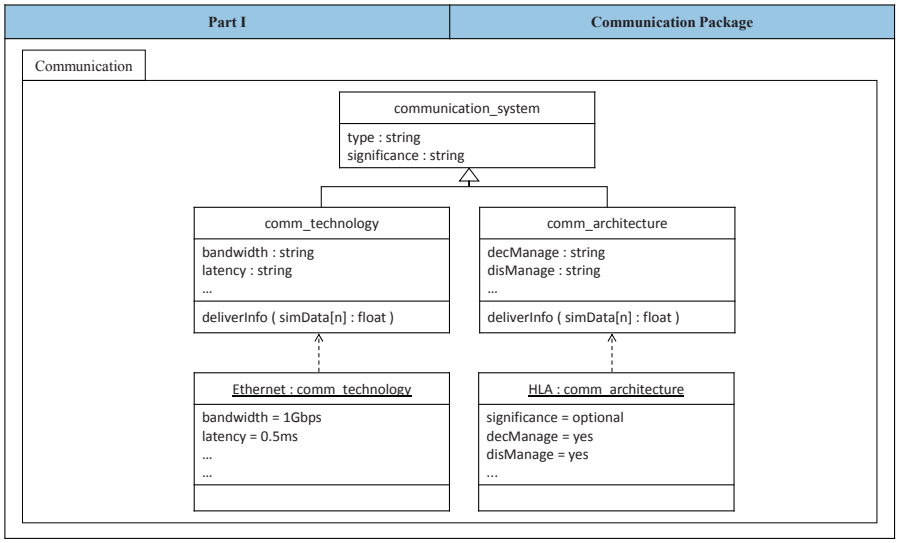


Figure 5-14: UML class diagram of the selected communication system components – package 2

The communication package includes a main class named: *communication_system*. Two classes inherit the *communication_system* class: *comm_technology* and *comm_architecture*. Each of these classes has an instance representing the selected solution element. Figure 5-15 shows the first section of the second part of the generated system model. This section contains the specification/requirement radar charts of the two selected driving simulators as well as the selected traffic simulator. The Appendix provides tables that facilitate the interpretation of the features and fidelity levels within the radar charts. Moreover, these tables can be displayed by the SoS configuration software upon user's request.

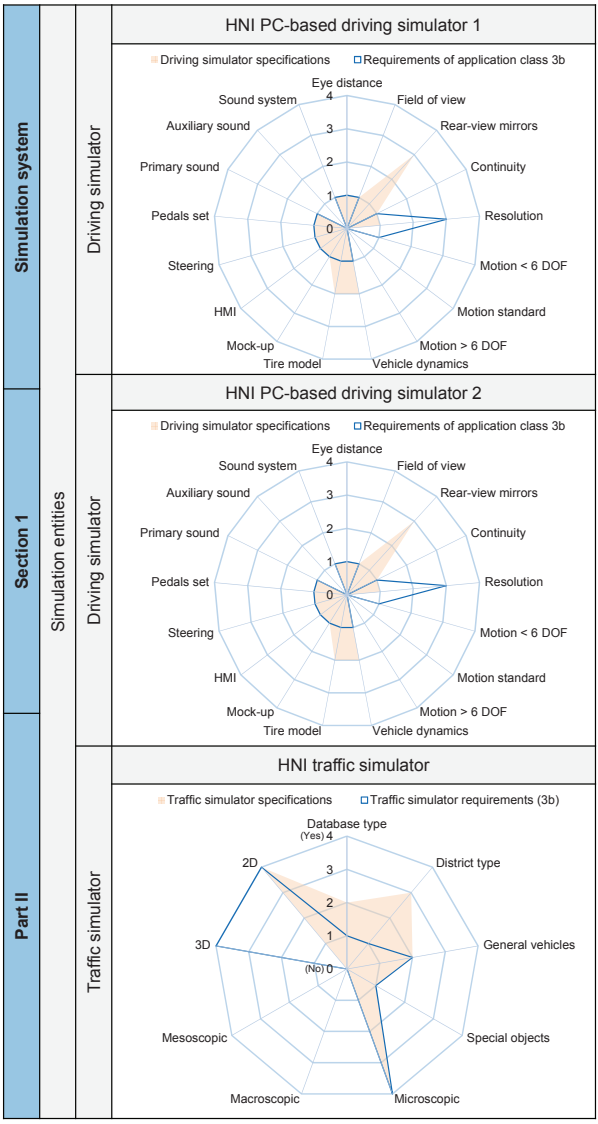


Figure 5-15: Specification/requirement radar charts of the selected simulation entities

Figure 5-16 shows the second section of the second part of the generated system model. This section contains the specification/requirement radar charts of the selected communication system components. Unlike the previous application scenarios, a communication architecture was selected in addition to the communication technology. The respective radar charts illustrate the percentage or binary fulfillment of requirements by the corresponding specifications of the selected solution elements. Section 4.5 provides descriptions for the network characteristics and the communication services shown

within the radar charts. Moreover, these descriptions can be displayed by the SoS configuration software upon user's request.

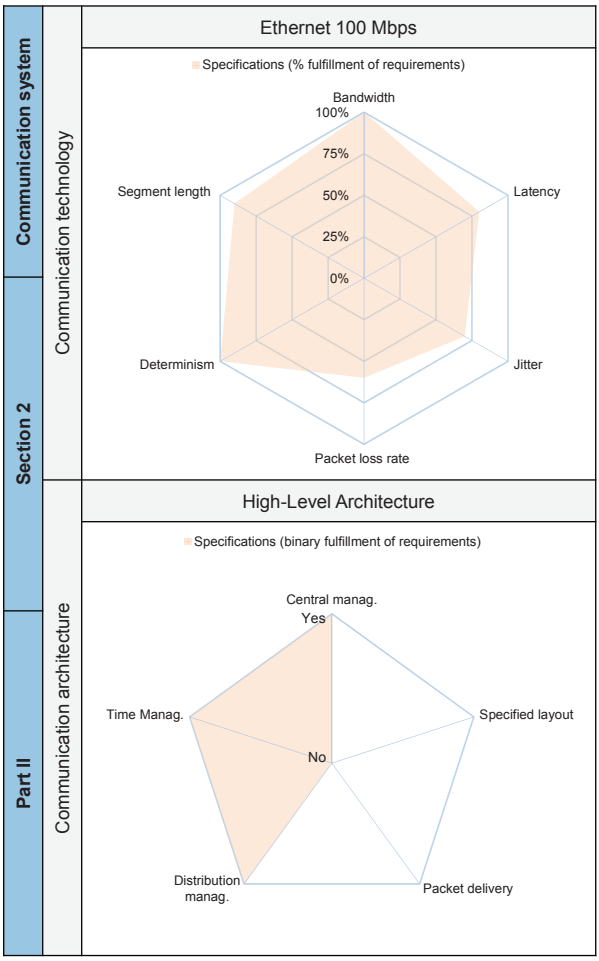


Figure 5-16: Specification/requirement radar charts of the selected communication components

Established platform: A platform of networked driving simulation was built in accordance with the generated system model for the third example application scenario. Figure 5-17 shows the two networked driving simulators of the developed platform.

Figure 5-17: Developed platform of networked driving simulation for the third example application scenario

The platform shown in Figure 5-17 utilizes two variants of the same driving simulator: HNI PC-based driving simulator. In this application scenario, the HNI PC-based driving simulator has a small screen and a normal seat unlike the configuration used within the previous two example application scenarios. Consequently, the shown platform does not impose special space requirements in comparison to the platforms of the previous validation examples. This platform has been demonstrated successfully during the FMB 2016 exhibition (German acronym that stands for Forum of Mechanical Engineering) in Bad Salzuflen in Germany. The exhibition visitors got insight into the developed platform of networked driving simulation and its intended application scenario.

Based on the requirements determined in Chapter 2, the following section provides a validation for the overall design framework presented in this work.

5.3 Design Framework Validation According to Requirements

The *System-Level Design Framework for Networked Driving Simulation* presented in this work depends on the model-based system engineering principles for building multi-disciplinary system models. Specifically, the presented design framework consists of a procedure model and a SoS configuration software for composing application-oriented system models of networked driving simulation. The requirements of the overall design framework were determined in Section 2.6 based on a comprehensive problem analysis.

These requirements have been divided principally into three sets (see Section 2.6). The first set includes four requirement items concerning the procedure model. The second set includes two requirement items for the developed SoS configuration software. The third set of requirements addresses the resultant networked driving simulation systems. It includes three requirement items inherited from the open system approach that is adopted by the system of systems engineering (see Subsection 2.4.3). The following is a validation of the overall design framework according to the three sets of requirements.

R1 – Systematic approach: The developed procedure model presents concrete phases for the systematic design of networked driving simulation. Each design phase within the procedure model consists of a group of tasks that include specific approaches towards increased system concretization. The design process can be used to create system models for networked driving simulation as a system of systems. These system models are tailored to the particular requirements of the concerned application scenarios.

R2 – Complexity mitigation: The developed procedure model reduces the design complexity of networked driving simulation. Specifically, the whole system has been fully analyzed with the help of a well-established specification technique (CONSENS). Moreover, system components have been identified, described, and classified. The approaches utilized within the procedure model save considerable time and effort while designing system models for networked driving simulation. Systems of networked driving simulation can be rapidly designed and established based on rigorous scientific foundation. Consequently, system users and domain-specific developers can concentrate on the actual operation objective of the networked driving simulation system.

R3 – Domain-spanning: The developed procedure model covers different aspects of the networked driving simulation that represents a typical multidisciplinary system. These aspects have been assigned to two main groups: simulation system and communication system. A set of requirements is defined for each aspect. The approaches utilized within the procedure model consider the defined sets of requirements to carry out a system-level design process.

R4 – Automation potential: The developed procedure model utilizes specific approaches to achieve the results of its different phases. The potential of automation was one of the crucial criteria for the selection of these particular approaches. Consequently, the adopted approaches have been embedded within the accompanying SoS configuration software to provide a system configuration process with a high extent of automation. Hence, system users and domain-specific developers do not have to perform exhaustive analysis during the configuration of the networked driving simulation system. Furthermore, no deep technical knowledge is required for selecting system components suitable for the concerned application scenarios.

R5 – Separation of concerns: The developed SoS configuration software follows the separation of concerns principle. There are mainly three separate aspects in this regard: user interface, configuration approaches, and associated system databases. With the

separation of concerns principle, system users have only to handle a very user-friendly interface during the configuration process. The underlying configuration approaches and the system databases were implemented in other embedded layers. The eventual future modification of any particular layer can be carried out with minimal change of other layers.

R6 – Extendable architecture: The SoS configuration software was implemented based on the model-view-controller approach. This resulted in a modular structure that provides a very good level of extensibility. New functions can be added in future and the embedded configuration approaches can be easily modified. Moreover, the tables of the associated system databases can be edited. Entries of existing solution elements can be modified and entries of new solution elements can be added to the tables. Furthermore, the database tables can be extended to add more attributes that characterize the solution elements.

R7 – Open interface principle: The presented example platforms of networked driving simulation follow the open interface principle. This feature was realized by using only standard and open communication technologies and architectures. Thereby, system capabilities can be easily extended by integrating additional components. Moreover, the exchange of simulation data can be easily modified when new components are added to the system.

R8 – Reconfiguration principle: The presented example platforms of networked driving simulation follow the reconfigurability principle. This feature was realized by utilizing reconfigurable driving simulators that have open structures. The individual components of the driving simulators can be easily exchanged. Thereby, the same networked driving simulation system can be easily modified to address new application scenarios.

R9 – Modularity principle: The presented example platforms of networked driving simulation follow the modularity principle. This feature was realized by utilizing system components that carry out independent, specific tasks. For instance, traffic simulation is considered in this work as a separate task independent of driving simulation. Thereby, the networked driving simulation system can be easily modified by changing only individual system components.

The following chapter outlines a summary and emphasizes the novelty of the presented work. In addition, future work is revealed to suggest possible enhancements.

6 Summary and Outlook

The automotive research and industry sectors have major concerns regarding the development and introduction of future autonomous and cooperative vehicle technologies. Driving simulation is an effective tool that can support the development of in-vehicle systems in a safe, cost-effective environment. However, with the introduction of autonomous and cooperative vehicle technologies, the traffic becomes more complex and interactive while the human drivers still represent an important factor. Conventional driving simulation does not provide an adequate representation of the multi-interactivity related to these advanced technologies. Yet networked driving simulation presents an intelligible solution to examine various aspects associated with the future traffic and transportation systems in a multi-interactive environment. Nonetheless, there are various components with different specifications for the development of networked driving simulation systems. Moreover, there are new multi-interactive application scenarios with different requirements, such as development of cooperative vehicle systems, group behavior analysis, and multi-driver training. According to the extensive literature review and analysis, there are no rigorous methodological approaches or tools to date for the application-oriented design of networked driving simulation systems.

The concept of networked driving simulation complies with the definition of system of systems (SoS). Specifically, two or more constituent systems exchange information and share the same virtual environment. A common system goal is accomplished through their collaboration within the system of systems environment: multi-interactive traffic scenario simulation. To establish an intelligible system of systems, the adopted structure must be able to evolve in order to accommodate potential changes of the application requirements. The evolution of a system of systems can be realized by integrating different constituent systems or modifying the characteristics of existing building components. However, this imposes a challenging demand on the architecting of system of systems to offer a capability for flexible adaptation. Classical system architecting is neither sufficient nor effective when it comes to heterogeneous networked systems. Architecting system of systems necessitates broader design considerations beyond the typical system development processes.

This work presented a new method for the systematic design of networked driving simulation as a typical system of systems. The key principles of systems engineering for an open system architecture, as well as the model-based system engineering measure for building multidisciplinary system models were considered in particular. The design method consists mainly of a procedure model accompanied by a SoS configuration software. With its concrete phases, the procedure model analyzes the system thoroughly and addresses all the necessary tasks for the system modeling process. The design process is embedded in the SoS configuration software to aid non-expert system users and domain-specific developers while selecting system components in accordance with the requirements of the concerned application scenarios. In particular, the design method

considers the whole system of networked driving simulation in two main aspects: **simulation system** and **communication system**. The novelty of the developed method can be emphasized by the following three concrete aspects:

- Combining and using two distinguished approaches from the literature for the selection of the simulation components of the networked driving simulation system.
- Utilizing a well-established decision-making method for the selection of the communication components of the networked driving simulation system.
- Creating system models for networked driving simulation in accordance with the principles of the agent-based modeling technique and the concepts of the UML class diagrams. The system model acts thereby as simple communication basis between users and domain-specific developers during the subsequent system realization step.

Together, these unique aspects form the first methodological work for networked driving simulation to date. The presented validation examples emphasized the flexible usability of the developed method by designing three different application-oriented system models. These system models make use of existing driving simulators in accordance with the requirements of the concerned application scenarios. The utilized driving simulators exhibit different complexity levels (mixed-fidelity levels). That is, they have different technical specifications and serve different purposes of use. However, new application scenarios were achieved by the integration in environments of networked driving simulation. Moreover, the utilized communication systems have different characteristics and capabilities. These were tailored to the data delivery requirements of the respective application scenarios. Three platforms of networked driving simulation have been built in accordance with the designed system models. With the accompanying SoS configuration software, non-expert users and domain-specific developers can make rely on the presented method to design further application-oriented system models of networked driving simulation. In summary, the developed *System-Level Design Framework for Networked Driving Simulation* enables the systematic development of resilient platforms of networked driving simulation.

Outlook

The developed *System-Level Design Framework for Networked Driving Simulation* presented a comprehensive procedure model that addresses all system aspects. Nonetheless, some enhancements can be carried out as future work with respect to the implementation prototype of the accompanying SoS configuration software:

- **SoS configuration software and Mechatronic Modeller interfacing**

In this work, the networked driving simulation system was specified exemplarily using the CONSENS specification technique in the first phase of the procedure model. This system specification phase was carried out without the support of the developed SoS configuration software. The Mechatronic Modeller is a software tool for the computer-

aided modeling of mechatronic systems using the CONSENS specification technique [GDN10]. Particularly, the Mechatronic Modeller can support users to create the partial models addressed in this work: environment, application scenarios, requirements, functions, and active structure. Furthermore, the Mechatronic Modeller can generate a comprehensive report containing the results of the system specification phase. The SoS configuration software can be developed further to able to load and read the reports generated by the Mechatronic Modeller. Accordingly, the configuration sequence can be adjusted at the beginning individually for each application scenario. Thereby, the user is guided only through the necessary configuration steps in accordance with the system components addressed within the partial models.

- **Further development of the graphical user interface**

In its current prototypical implementation, the graphical user interface of the SoS configuration software was developed with MATLAB. MATLAB was selected as it provides all the necessary functions that facilitate the implementation of the selection and decision-making processes addressed within the procedure model. The utilization of MATLAB functions is very convenient and does not require prior knowledge of special programming techniques. This aspect is useful when it comes to rapid software prototyping. However, MATLAB provides only primitive elements for the design of graphical user interfaces. The graphical user interface of the developed SoS configuration software can be reconstructed with more advanced elements using other programming tools or languages. For instance, the Qt C++ software can be utilized to establish a more user-friendly graphical user interface for the SoS configuration software. Qt C++ is a cross-platform application framework that can be used to develop software applications and graphical user interfaces

- **Further development of the system databases**

In its current prototypical implementation, the system components database was developed with Microsoft Office Excel. Microsoft Office Excel can be used to easily construct database tables, while the database operations are carried out by another external tool. In this work, MATLAB was used to carry out all typical database operations on the tables created with Microsoft Office Excel: create, read, update, and delete. This approach supported the rapid software prototyping purpose in this work. However, the database access time is relatively long using this approach. The system components database can be implemented using a better database language such as SQL. SQL is a standard language for handling data held in relational database systems.

Addressing these particular aspects in future work can increase the flexibility and convenience of the developed framework. The system usability can be enhanced thereby as a logical consequence.

In addition, further non-traditional application scenarios for driving simulation can be introduced and analyzed in future work. The purpose of these non-traditional applica-

tion scenarios is to keep up with the rapid advancements in the automotive field. The requirements of these application scenarios can be determined based on the demands of future autonomous and cooperative driving systems. These requirements can be used with the presented method and the accompanying SoS configuration software to create a ready set of system models for networked driving simulation. Users and domain-specific developers can use this set of system models to establish corresponding application-oriented platforms for networked driving simulation.

7 List of Abbreviations

2D	Two-Dimensional
3D	Three-Dimensional
ADAS	Advanced Driver Assistance Systems
ATMOS	Atlas Motion System
bps	bits per second
CAN	Controller Area Network
Ch.	Chapter
CONSENS	Conceptual Design Specification Technique for the Engineering of Complex Systems
CRDU	Create, Read, Update, and Delete
DB	Database
DiL	Driver-in-the-Loop
DIS	Distributed Interactive Simulation
DLR	Deutsches Zentrum für Luft- und Raumfahrt
DOF	Degrees of Freedom
e.g.	exempli gratia – for example
et al.	et alii – and others
etc.	et cetera – etcetera
Gbps	Gigabits per second
GUI	Graphical User Interface
HiL	Hardware-in-the-Loop
HLA	High-Level Architecture
HMI	Human-Machine-Interface
HNI	Heinz Nixdorf Institute
HW	Hardware
i.e.	id est – that is
ID	Identification, Identity, or Identifier

LCD	Liquid Crystal Display
Mbps	Megabits per second
MbSE	Model-based Systems Engineering
MVC	Model-View-Controller
NII	National Institute of Informatics
OSU	Oregon State University
SE	Systems Engineering
SiL	Software-in-the-Loop
SoS	System of Systems
SoSE	System of Systems Engineering
SQL	Structured Query Language
SUMO	Simulation of Urban Mobility
SW	Software
TCP/IP	Transmission Control Protocol/Internet Protocol
TENA	Test and Training Enabling Architecture
TSS	Transport Simulation Systems
UDP/IP	User Datagram Protocol/Internet protocol
UI	Unit Interval
UML	Unified Modeling Language
XML	Extensible Markup Language

8 Bibliography

[ABC+14] AUBERLET, J. M.; BHASKAR, A.; CIUFFO, B.; FARAH, H.; HOOGENDOORN, R.; LEONHARDT, A.: Data collection techniques. In: DAAMEN, W.; BUISSON, C.; HOOGENDOORN, S. P. (Ed.): Traffic Simulation and Data: Validation Methods and Applications. 1st edition, CRC Press, Taylor and Francis Group, Boca Raton, Florida, USA, 2014, pp. 7-29 – ISBN 9781482228700

[ABD+14] ANACKER, H.; BRENNER, C.; DOROCIAK, R.; DUMITRESCU, R.; GAUSEMEIER, J.; IWANEK; P.; SCHÄFER, W.; VAßHOLZ, M.: Methods for the Domain-Spanning Conceptual Design. In: GAUSEMEIER, J.; RAMMIG, F. J.; SCHÄFER, W. (Ed.): Design Methodology for Intelligent Technical Systems: Develop Intelligent Technical Systems of the Future. 1st edition, Springer-Verlag, Heidelberg, Germany, 2014, pp. 117-171 – ISBN 9783642454349

[ADK13] ACHESONA, P.; DAGLIA, C.; KILICAY-ERGIN, N.: Model Based Systems Engineering for System of Systems Using Agent-based Modeling. In: Proceedings of Conference on Systems Engineering Research (CSER'13), March 2013, Atlanta, GA, USA, Elsevier, ScienceDirect, Procedia Computer Science, Vol. 16, pp. 11-19 – DOI 10.1016/j.procs.2013.01.002

[ADM05] ARAI, K.; DEGUCHI, H.; MATSUI, H.: Agent-Based Modeling Meets Gaming Simulation. Vol. 2, Springer-Verlag Tokyo, Japan, 2005 – ISBN 9784431294269

[AN13] ARIOUI, H.; NEHAOUA, L.: Driving Simulation. 1st edition, Wiley-ISTE, New Jersey, USA, 2013 – ISBN 9781848214675

[And98] ANDERSON, D.: FireWire System Architecture: IEEE 1394A. 2nd edition, Addison-Wesley Professional, Boston, Massachusetts, USA, 1998 – ISBN 9780201485356

[ARC11] ALLEN, R.; ROSENTHAL, T.; COOK, M.: A Short History of Driving Simulation. In: FISHER, D. L.; RIZZO M.; CAIRD, J.; LEE, J. (Eds.): Handbook of Driving Simulation for Engineering, Medicine, and Psychology. 1st edition, CRC Press Taylor & Francis Group, USA, 2011, pp. 2.2-2.11 – ISBN 9781420061000

[Aza08] AZANI, C.: An Open Systems Approach to System of Systems Engineering. IN: JAMSHIDI, M. (Ed.): System of Systems Engineering: Innovations for the Twenty-first Century. 1st edition, John Wiley & Sons, New Jersey, USA, 2008, pp. 21-43 – ISBN 9780470195901

[BA05] BJÖRKLUND, G. M.; ABERG, L.: Driver behaviour in intersections: Formal and informal traffic rules. In: Journal of Transportation Research Part F – Traffic Psychology and Behaviour, May 2005, Elsevier, Amsterdam, The Netherlands, Vol. 8, Issue 3, pp. 239-253 – DOI 10.1016/j.trf.2005.04.006

[Bar10] BARCELO, J.: Fundamentals of Traffic Simulation. 1st edition, Springer Science & Business Media: New York, USA, 2010 – ISBN 9781441961419

[BAS17-ol] BAST FEDERAL HIGHWAY RESEARCH INSTITUTE: http://www.bast.de/, 2017

[BBC+99] BROGGI, A.; BERTOZZI, M.; CONTE, G.; FASCIOLI, A.: Automatic Vehicle Guidance – The Experience of the Argo Autonomous Vehicle. 1st edition, World Scientific Publishing, Singapore, 1999, pp. 34-37 – ISBN 9789810237202

[Bei14] BEIKER, S.: History and Status of Automated Driving in the United States. In: GEREON, M.; BEIKER, S. (Ed.): Road Vehicle Automation. 1st edition, Springer International Publishing, Switzerland, 2014, pp. 61-70 – ISBN 9783319059891

[Bel10] BEL GEDDES, N.: Magic motorways. Reproduction of a historical book version before 1923, Nabu Press historical reprints publisher, BiblioLabs LLC, Charleston, South Carolina, USA, 2010 – ISBN-13 9781171860020

[Ber12] BERSINI, H.: UML for ABM. In: Journal of Artificial Societies and Social Simulation, January 2012, Vol. 15, No. 1 – DOI 10.18564/jasss.1897

[BFF15] BARNARD, Y.; FISCHER, F.; FLAMENT, M.: Field Operational Tests and Deployment Plans. In: CAMPOLO, C.; MOLINARO, A.; SCOPIGNO, R. (Ed.): Vehicular ad hoc Networks: Standards, Solutions, and Research. 1st edition, Springer International Publishing, Switzerland, 2015, pp. 393-408 – ISBN 9783319154961

[BG98] BAHILL, A. T.; GISSING, B.: Re-evaluating systems engineering concepts using systems thinking. In: IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), November 1998, Vol. 28, Issue 4, pp. 516-527 – ISSN 10946977

[BGR06] BONOTTI, A.; GENOVALI, L.; RICCI, L.: DiVES: A distributed support for networked virtual environments. In: Proceedings of 20th International Conference on Advanced Information Networking and Applications (AINA 2013), April 18-20 2006, Gothenburg, Sweden – ISBN 0769524664

[BHM+04] BEASLEY, H. H.; HARDING, T. H.; MARTIN, J. S.; RASH, C. E.: Laboratory system for the evaluation of helmet-mounted displays. In: RASH, C. E.; REESE, C. E. (Ed.): Helmet- and Head-Mounted Displays IX – Technologies and Applications. In: Proceedings of SPIE, the International Society for Optical Engineering, Volume 5442, SPIE Publishing, Orlando, FL, USA, September 2004 – ISBN 9780819453655

[Bir80] BIRKHOFER, H.: Analyse und Synthese der Funktionen technischer Produkte. VDI-Verlag Fortschritts-Bericht VDI-Z, Reihe 1, Nr. 70, Düsseldorf, Germany, 1980

[BKG08] BERSSENBRÜGGE, J.; KREFT, S.; GAUSEMEIER, J.: Using a Virtual Reality-based Night Drive Simulator as a Tool for the Virtual Prototyping of an Advanced Leveling Light System. In: Proceedings of IDETC/CIE 2008 ASME 2008 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, August 3-8 2008, New York City, USA

[BLM15] BANOS, A.; LANG, C.; MARILLEAU, N.: Agent-Based Spatial Simulation with NetLogo. 1st edition, ISTE Press, Elsevier, London, UK, 2015, pp. 1-20 – ISBN 9781785480553

[Bou92] BOURNE, J. R.: Tools for Building Applications: The Model-View-Controller Paradigm and View Components. In: BOURNE, J. R. (Eds.): Object-Oriented Engineering: Building Engineering Systems Using Smalltalk-80. 1st edition, CRC Press, Taylor & Francis Group, Boca Raton, Florida, USA, 1992 – ISBN 9780256112108

[BP90] BLETHYN, S. G.; PARKER, C. Y.: Designing Information Systems. 1st edition, Elsevier, Amsterdam, The Netherlands, 1990 – ISBN 9780750610384

[BS12] BANDO, T.; SHIBATA, T.: Networked driving simulator based on SIGVerse and lane-change analysis according to frequency of driving. In: Proceedings of 15th International IEEE Conference on Intelligent Transportation Systems (ITSC), September 16-19 2012, Anchorage, AK, USA – ISBN 9781467330640

[BW14] BERNHART, W.; WINTERHOFF, M.: Autonomous Driving – Disruptive Innovation that Promises to Change the Automotive Industry as We Know It. In: LANGHEIM, J. (Ed.): Energy Consumption and Autonomous Driving. In: Proceedings of the 3rd CESA Automotive Electronics Congress, Paris, Springer International Publishing, Switzerland, 2014, pp. 3-10 – ISBN 9783319198187

[Cac04] CACCIABUE, P. C.: Guide to Applying Human Factors Methods: Human Error and Accident Management in Safety-critical Systems. 1st edition, Springer-Verlag, London, UK, 2004 – ISBN 9781849968980

[Cal90] CALANDRINO, L.: Telematics and Transportation Safety Management – Review of the EUREKA/PROMETHEUS Research Project Pro-Com. In: SOEKKHA, H.M.; BOVY, P.H.L.; DREWE, P.; JANSEN, G.R.M. (Ed.): Telematics – Transportation and Spatial Development. 1st edition, CRC Press, Taylor and Francis Group, Boca Raton, Florida, USA, 1990, pp. 173-183 – ISBN 9789067641258

[CDF+07] COLDITZ, F.; DRAGON, L.; FAUL, R.; MELJNIKOV, D.; SCHILL, V.; UNSELT, T.; ZEEB, E.: Use of Driving Simulators within Car Development. In: Proceedings of Driving Simulation Conference North America, September 12-14 2007, Iowa City, USA

[CF01] CARLOCK, P. G.; FENTON, R. E.: System of Systems (SoS) enterprise systems engineering for information-intensive organizations. In: DE WECK, O. L. (Ed.): Journal of Systems Engineering, John Wiley & Sons, Hoboken, New Jersey, USA, October 2001, Vol. 4, Issue 4, pp. 242-261 – e-ISSN 15206858

[CJ11] CARSTEN, O.; JAMSON, H.: Driving Simulators as Research Tools in Traffic Psychology. In: Porter, B. E. (Eds.): Handbook of Traffic Psychology. Academic press, Elsevier, Waltham, USA, 2011, pp. 87-96 – ISBN 9780123819840

[CLC10] CAI, H.; LIN, Y.; CHENG, B.: Mimicking human driving behaviour for realistic simulation of traffic flow. In: International Journal of Simulation and Process Modelling, October 2010, Vol. 6, No. 2, pp. 126-136 – DOI 10.1504/IJSPM.2010.036017

[CM11] CASTRONOVO, A.; MÜLLER, C.: A Schema of Possible Negative Effects of Advanced Driver Assistant Systems. In: Proceedings of the 6th International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design (Driving Assessment-2), June 27-30 2011, Lake Tahoe, California, USA, Public Policy Center, University of Iowa, 2011

[CMV+10] CLOUTIER, R.; MULLER, G.; VERMA, D.; NILCHIANI, R.; HOLE, E.; BONE, M.: The Concept of Reference Architectures. In: Systems Engineering Journal, Spring 2010, Vol. 13, Issue 1, pp. 14-27, Willy Online Library, John Wiley & Sons, Hoboken, New Jersey, USA, 2010 – DOI 10.1002/sys.20129

[Com17-ol] COMPUTER HISTORY MUSEUM: http://www.computerhistory.org/, 2017

[Cor11] CORKE, P.: Robotics, Vision and Control: Fundamental Algorithms in MATLAB. 1st edition, Springer-Verlag, Berlin, Germany, 2011, pp. 65-85 – ISSN 16107438

[DAG12] DUMITRESCU, R.; ANACKER, H.; GAUSEMEIER, J.: Design Framework for the Integration of Cognitive Functions into Intelligent Technical Systems. In: Proceedings of 1st Joint International Symposium on System-integrated Intelligence, New Challenges for Product and Production Engineering, 2012, June 17-19 2012, Hannover, Germany, pp. 17-20, 2012 – ISBN 9783943104592

[Del08] DELAURENTIS, D. A.: Understanding Transportation as a System of Systems Problem. IN: JAMSHIDI, M. (Ed.): System of Systems Engineering: Innovations for the Twenty-first Century. 1st edition, John Wiley & Sons, New Jersey, USA, 2008, pp. 520-541 – ISBN 9780470195901

[Dim10] DIMARIO, M. J.: System of Systems Collaborative Formation. 1st edition, World Scientific Publishing, Toh Tuck Link, Singapore, August 2010, pp. 5-22 – ISBN 9789814313889

[DK08] DAGLI, C. H.; KILICAY-ERGIN, N.: System of Systems Architecting. IN: JAMSHIDI, M. (Ed.): System of Systems Engineering: Innovations for the Twenty-first Century. 1st edition, John Wiley & Sons, New Jersey, USA, 2008, pp. 77-100 – ISBN 9780470195901

[DPV+11] DIAS, L.; PEREIRA, G.; VIK, P.; OLIVEIRA, J.: Discrete Simulation Tools Ranking– a Commercial Software Packages comparison based on popularity. In: Proceedings of 9th Industrial Simulation Conference ISC, June 6-8 2011, Venice, Italy

[DT95] DOGANATA, Y.N.; TANTAWI, A.N.: Analysis of communication requirements for intelligent transportation systems: methodology and examples. In: Proceedings of 45th IEEE Vehicular Technology Conference, July 25-28 1995, Chicago, IL, USA – ISBN 078032742X

[Dum11] DUMITRESCU, R.: Entwicklungssystematik zur Integration kognitiver Funktionen in fortgeschrittene mechatronische Systeme. Ph.D. Dissertation, Faculty of Mechanical Engineering, University of Paderborn, HNI Publication Series, Volume 286, 2011

[DWT15] DENNIS, A.; WIXOM, B. H.; TEGARDEN, D.: Systems Analysis and Design: An Object-Oriented Approach with UML. 5th edition, John Wiley & Sons, New Jersey, USA, 2015, pp. 1-36 – ISBN 9781118804674

[FCR+11] FISHER, D.; CAIRD, J.; RIZZO M.; LEE, J.: Handbook of Driving Simulation for Engineering, Medicine, and Psychology – An Overview. In: FISHER, D.; CAIRD, J.; RIZZO M.; LEE, J. (Eds.): Handbook of Driving Simulation for Engineering, Medicine, and Psychology. CRC Press Taylor & Francis Group, USA, 2011, pp. 1.1-1.16 – ISBN 9781420061017

[Fra17] FRANKE, U.: Autonomous Driving. In: LOPEZ, A. M.; IMIYA, A.; PAJDLA, T.; ALVAREZ, J. M. (Ed.): Computer Vision in Vehicle Technology – Land, Sea, and Air. 1st edition, John Wiley & Sons, New Jersey, USA, 2017, pp. 24-54 – ISBN 9781118868072

[FSA+11] FISCHER, M.; SEHAMMAR, H.; AUST, M. L.; NILSSON, M.; LAZIC, N.; WEIEFORS, H.: Advanced driving simulators as a tool in early development phases of new active safety functions. In: Proceedings of the 3rd International Conference on Road Safety and Simulation September 14-16 2011, Indianapolis, USA, 2011

[FVR+99] FARMER, E.; VAN ROOIJ, J.; RIEMERSMA, J.; JORNA, P.: Handbook of Simulator-Based Training. 1st edition, Routledge, Taylor & Francis Group, Abingdon, UK, 1999, Ch. 11 – ISBN 9780754611875

[GAC+13] GAUSEMEIER, J.; ANACKER, H.; CZAJA, A.; WAßMANN, H.; DUMITRESCU, R.: Auf dem Weg zu intelligenten technischen Systemen. In: GAUSEMEIER, J.; DUMITRESCU, R.; RAMMIG, F.-J.; SCHÄFER, W.; TRÄCHTLER, A. (Hrsg.): 9th Workshop on Design of Mechatronic Systems, April 18-19 2013, Paderborn, Germany, HNI Publication Series, Vol. 310, 2013, pp. 11-47 – ISBN 9783942647298

[Gau13a] GAUSEMEIER, J.: Strategische Planung und Entwicklung der Produkte von morgen. Publication series of Nordrhein-Westfälische Akademie der Wissenschaften, Ferdinand Schöningh Verlag, Paderborn, 2013

[Gau13b] GAUSEMEIER, S.: Ein Fahrerassistenzsystem zur prädiktiven Planung energie- und zeitoptimaler Geschwindigkeitsprofile mittels Mehrzieloptimierung. Ph.D. Dissertation, Faculty of Mechanical Engineering, University of Paderborn, Universitätsbibliothek Paderborn, Publikationsservice, 2013

[GB11] GREENBERG, J.; BLOMMER, M.: Physical Fidelity of Driving Simulators. In: FISHER, D.; CAIRD, J.; RIZZO M.; LEE, J. (Eds.): Handbook of Driving Simulation for Engineering, Medicine, and Psychology. CRC Press Taylor & Francis Group, USA, 2011, pp. 7.1-7.23 – ISBN 9781420061017

[GB14] GÄNG, J.; BERTSCHE, B.: An approach to describe interactions in and between mechatronic systems. In: MARTORELL, S.; SOARES, C. G.; BARNETT, J. (Ed.): Safety, Reliability and Risk Analysis: Theory, Methods and Applications. Vol. 3, CRC Press Taylor and Francis Group, Florida, USA, 2014, pp. 2233-2238 – ISBN 9781482266481

[GCW+13] GAUSEMEIER, J.; CZAJA, A. M.; WIEDERKEHR, O.; DUMITRESCU, R.; TSCHIRNER, C.; STEFFEN, D.: Studie: Systems Engineering in der industriellen Praxis. In: GAUSEMEIER, J.; DUMITRESCU, R.; RAMMIG, F.-J.; SCHÄFER, W.; TRÄCHTLER, A. (Hrsg.): 9th Workshop on Design of Mechatronic Systems, April 18-19 2013, Paderborn, Germany, HNI Publication Series, Vol. 310, 2013 – ISBN 9783942647298

[GCW+15] GAUSEMEIER, J.; CZAJA, A.; WIEDERKEHR, O.; DUMITRESCU, R.; TSCHIRNER, C.; STEFFEN, D.: Studie: Systems Engineering in der industriellen Praxis. In: MAURER, M.; SCHULZE, S. (Ed.): Tag des Systems Engineering, November 2013, Carl Hanser Verlag, Stuttgart, Germany, 2013, pp. 113–122 – ISBN 9783446439153

[GDN10] GAUSEMEIER, J.; DOROCIAK, R.; NYßEN, A.: The Mechatronic Modeller – A Software Tool for Computer-Aided Modeling of the Principle Solution of an Advanced Mechatronic System. In: Proceedings of the 11th International Workshop on Research and Education in Mechatronics, September 9-10 2010, Ostrava, the Czech Republic, 2010 – ISBN 8888412336

[GEK01] GAUSEMEIER, J.; EBBESMEYER, P.; KALLMEYER, F.: Produktinnovation – Strategische Planung und Entwicklung der Produkte von morgen. 1st edition, Hanser Fachbuch Verlag, München, Germany, 2001 – ISBN 9783446216310

[GFD+09] GAUSEMEIER, J.; FRANK, U.; DONOTH, J.; KAHL, S.: Specification technique for the description of self-optimizing mechatronic systems. In: Research in Engineering Design, November 2009, Vol. 20, Issue 4, Springer, London, 2009, pp. 201-223 – ISSN 09349839

[GGT13] GAUSEMEIER, J.; GAUKSTERN, T.; TSCHIRNER, C.: Systems Engineering Management Based on a Discipline-Spanning System Model. In: C.J.J. Paredis, C. Bishop, D. Bodner (Eds.): Conference on Systems Engineering Research (CSER'13), 19-22 March 2013, Atlanta, USA, 2013

[GM03] GAUSEMEIER, J.; MOEHRINGER, S.: NEW GUIDELINE VDI 2206 – A FLEXIBLE PROCEDURE MODEL FOR THE DESIGN OF MECHATRONIC SYSTEMS. In: Proceedings of 14th International Conference on Engineering Design (ICED), August 19-21 2003, Stockholm, Sweden – Paper No. DS31_1149FPB

[GNM+13] GAJANANAN, K.; NANTES, A.; MISKA, M.; NAKASONE, A.; PRENDINGER, H.: An Experimental Space for Conducting Controlled Driving Behavior Studies based on a Multiuser Networked 3D Virtual Environment and the Scenario Markup Language. In: IEEE Transactions on Human-Machine Systems, July 2013, Vol. 29, Issue 7, pp. 345-358 – DOI 10.1109/TSMC.2013.2265876

[GPN17] GRUBMÜLLER, S.; PLIHAL, J.; NEDOMA, P.: Automated Driving from the View of Technical Standards. In: WATZENIG, D; HORN, M. (Ed.): Automated Driving: Safer and More Efficient Future Driving. 1st edition, Springer International Publishing, Switzerland, 2017, pp. 29-40 – ISBN 9783319318936

[Has03] HASS, A. M. J.: Configuration Management Principles and Practice. 1st edition, Addison-Wesley Professional, Pearson PLC, Boston, USA, 2003 – ISBN 9780321117663

[Has14] HASSAN, B.: A Design Framework for Developing a Reconfigurable Driving Simulator. Ph.D. Dissertation, Faculty of Mechanical Engineering, University of Paderborn, HNI Publication Series, Volume 333, 2014

[HBK+12] HEESEN, M.; BAUMANN, M.; KELSCH, J.; NAUSE, D.; FRIEDRICH, M.: Investigation of Cooperative Driving Behaviour during Lane Change in a Multi-Driver Simulation Environment. In: Proceedings of Human Factors and Ergonomics Society (HFES) Europe Chapter Conference, October 10-12 2012, Toulouse, France – ISBN 9780945289449

[Heg10] HEGDE, G. S.: Mechatronics. 1st edition, Jones & Bartlett Learning, Burlington, Massachusetts, USA, 2010, pp. 1-34 – ISBN 9781934015292

[HK12] MANDIAU, R.; PIECHOWIAK, S.; DONIEC A.; ESPIE, S.: Agent-oriented Road Traffic Simulation. IN: HAMMADI, SLIM; KSOURI, M. (Ed.): Advanced Mobility and Transport Engineering. 1st edition, John Wiley & Sons, New Jersey, USA, 2012, pp. 1-26 – ISBN 9781848213777

[HLH+13] HU, X.; LIU, X.; HE, Z.; ZHANG, J.: On the use of OpenStreetMap data for V2X channel modeling in urban scenarios. In: Proceedings of IET International Conference on Smart and Sustainable City 2013 (ICSSC 2013), August 19-20 2013, Gothenburg, Sweden – ISBN 9781849197076

[HMJ+15] HURWITZ, D.; MONSERE, C.; JANNAT, M.; WARNER, J.; RAZMPA, A.: Towards Effective Design Treatment for Right Turns at Intersections with Bicycle Traffic. Final Report by Oregon State University and Portland State University for Oregon Department of Transportation, November 2015, Contract or Grant No. SPR 767, Report No. FHWA-OR-RD-16-06

[Hou08] HOUTENBOS, M.: Expecting the unexpected: a study of interactive driving behaviour at intersections. Ph.D. Dissertation, TU Delft, TRAIL Research School, 2008, ISBN 9789055840953

[HR03] HANCOCK, P.; RIDDER, S.: Behavioural accident avoidance science: understanding response in collision incipient conditions. In: Ergonomics Journal, October 2003, Vol. 46, No. 12, pp. 1111-1135 – DOI 10.1080/0014013031000136386

[HS11] HANCOCK, P.; SHERIDAN, T.: The Future of Driving Simulation. In: FISHER, D.; CAIRD, J.; RIZZO M.; LEE, J. (Eds.): Handbook of Driving Simulation for Engineering, Medicine, and Psychology. CRC Press Taylor & Francis Group, USA, 2011, pp. 4.1-4.10 – ISBN 9781420061000

[HTF96] HARASHIMA, F.; TOMIZUKA, M.; FUKUDA, T.: Mechatronics – "What Is It, Why and How" An editorial. In: IEEE/ASME Transactions on Mechatronics, Vol. 1, Issue 1, 1996, pp. 1-4 – ISSN 10834435

[Iee17-ol] IEEE SPECTRUM: http://www.spectrum.ieee.org/, 2017

[IOL+13] IMAMURA, T.; OGI, T.; LUN, E. T. C.; ZHANG, Z.; MIYAKE, T.: Trial Study of Traffic Safety Education for High School Students Using Driving Simulator. In: Proceedings of IEEE International Conference on Systems, Man, and Cybernetics (SMC), October 2013, Manchester, UK – ISBN 9781479906505

[Ise05] ISERMANN, R.: Mechatronic Systems: Fundamentals. 2nd edition, Springer-Verlag, London, UK, 2005 – ISBN 1852339306

[Jam05] JAMSHIDI, M.: System-of-systems engineering - a definition. In: Proceedings of IEEE/SMC International Conference on System of Systems, Big Island, Hawaii, USA, January 2005

[Jam08a] JAMSHIDI, M.: Introduction to System of Systems. IN: JAMSHIDI, M. (Ed.): System of Systems Engineering: Innovations for the Twenty-first Century. 1st edition, John Wiley & Sons, New Jersey, USA, 2008, pp. 1-20 – ISBN 9780470195901

[Jam08b] JAMSHIDI, M.: Systems of Systems Engineering: Principles and Applications. 1st edition, CRC Press Taylor and Francis Group, Florida, USA, 2008 – ISBN 9781420065886

[Jam11] JAMSON, H.: Cross-platform Validation Issues. In: FISHER, D.; CAIRD, J.; RIZZO, M.; LEE, J. (Eds.): Handbook of Driving Simulation for Engineering, Medicine, and Psychology. CRC Press Taylor & Francis Group, USA, 2011, pp. 12.1-8.13 – ISBN 9781420061000

[Jan12] JANSCHEK, K.: Mechatronic Systems Design: Methods, Models, Concepts. 1st edition, Spriner-Verlag, Berlin, Germany, 2012 – ISBN 9783642175305

[Joh08] JOHNSON, M. A.: From engineering to system engineering to system of systems engineering. In: Proceedings of World Automation Congress (WAC), October 2008, Hawaii, HI, USA – ISBN 9781889335384

[Joh08a] JOHNSON, M. A.: From engineering to system engineering to system of systems engineering. In: Proceedings of IEEE World Automation Congress (WAC), October 2008, Hawaii, HI, USA – ISBN 9781889335384

[Kaw90] KAWASHIMA, H.: Japanese perspective of driver information systems. In: Transportation Journal, September 1990, Vol. 17, Issue. 3, pp. 263-284 – DOI 10.1007/BF02127039

[Kea08] KEATING, C. B.: Emergence in System of Systems. IN: JAMSHIDI, M. (Ed.): System of Systems Engineering: Innovations for the Twenty-first Century. 1st edition, John Wiley & Sons, New Jersey, USA, 2008, pp. 169-190 – ISBN 9780470195901

[KGG+11] KREFT, S.; GAUSEMEIER, J.; GRAFE, M.; HASSAN, B.: Automatisierte Trassierung virtueller Straßen auf Basis von Geo-Informationssystemen. In: Gausemeier, J.; Grafe, M. (Eds.): 10. Paderborner Workshop "Augmented & Virtual Reality in der Produktentstehung", 19. - 20. Mai 2011, HNI-Verlagsschriftenreihe, Band 295, Paderborn, 2011, pp. 253-266 ISBN 9783942647144

[KH16] KÜHN, M.; HANNAWALD, L.: Driver Assistance and Road Safety. In: WINNER, H.; HAKULI, S.; LOTZ, F.; SINGER, C. (Ed.): Handbook of Driver Assistance Systems – Basic Infor-

mation, Components and Systems for Active Safety and Comfort. 1st edition, Springer International Publishing, Switzerland, 2016, pp. 69-90 – ISBN 9783319123516

[Kim05] KIM, G.: Output Display. IN: KIM, G. (Ed.): Designing Virtual Reality Systems. 1st edition, Springer-Verlag London, UK, 2005, pp. 77-113 – ISBN 9781852339586

[KRU+15] KEATING, C.; ROGERS, R.; UNAL, R.; DRYER, D.; SOUSA-POZA, A.; SAFFORD, R.; PETERSON, W.; RABADI, G.: System of Systems Engineering. In: DOOLEN, T.; VAN AKEN, E. (Ed.): Engineering Management Journal, Taylor and Francis Group, Boca Raton, Florida, USA, April 2015, Vol. 15, Issue 3, pp. 36-45 – DOI 10.1080/10429247.2003.11415214

[KSK11] KANDHAI, K.; SMITH, M.; KANNEH, A.: Immersive driving simulation for driver education and analysis. In: Proceedings of 16th International Conference on Computer Games (CGAMES), August 2011, Louisville, KY, USA – ISBN 9781457714511

[KW16] KOOPMAN, P.; WAGNER, M.: Challenges in Autonomous Vehicle Testing and Validation. In: SAE International Journal of Transportation Safety, Vol. 4, Issue 1, 2016, pp. 15-24 – DOI 10.4271/2016-01-0128

[KYB14] KRAUSE, M.; YILMAZ, L.; BENGLER, K.: Comparison of real and simulated driving for a static driving simulator. In: STANTON, N.; LANDRY, S.; DI, G. (Ed.): Advances in Human Aspects of Transportation: Part II. 1st edition, AHFE Conference 2014, Krakow, Poland, 2014, pp. 29-40 – ISBN 9781495120985

[Lan00] LANGLOTZ, G.: Ein Beitrag zur Funktionsstrukturentwicklung innovativer Produkte. Forschungsberichte aus dem Institut für Rechneranwendung in Planung und Konstruktion RPK der Universität Karlsruhe, Shaker Verlag, 2000

[LBJ+11] LORENZ, T.; BAUMANN, M.; JASCHKE, K.; KOSTER, F.: A Modular and Scalable Application Platform for Testing and Evaluating Its Components. In: Proceedings of 20th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), June 27-29 2011, Paris, France – ISBN 9781457701344

[LBJ+16] BILIK, I.; BIALER, O.; VILLEVAL, S.; SHARIFI, H.; KONA, K.; PAN, M.; PERSECHINI, D.; MUSNI, M.; GEARY, K.: Automotive MIMO radar for urban environments. In: Proceedings of IEEE Radar Conference (RadarConf), May 2-6 2016, Philadelphia, PA, USA – e-ISBN 9781509008636

[LH13] LEE, J.-D.; HUANG, K.-F.: A Warning System for Obstacle Detection at Vehicle Lateral Blind Spot Area. In: Proceedings of 7th Asia Modelling Symposium (AMS), July 23-25 2013, Hong Kong, China – ISSN 23761164

[LJX+16] LIHUA, L.; JING, C.; XIAOQUN, L.; ANXIN, Z.: Research and implementation of virtools based on 3D virtual teaching laboratory. In: Proceedings of 3rd International Conference on Computer Science and Network Technology (ICCSNT), October 12-13 2013, Dalian, China – ISBN 9781479905607

[LVG+10] LASCHINSKY, J.; VON NEUMANN-COSEL, K.; GONTER, M.; WEGWERTH, C.; DUBITZKY, R.; KNOLL, A.: Evaluation of an Active Safety Light using Virtual Test Drive within Vehicle in the Loop. In: Proceedings of IEEE International Conference on Industrial Technology (ICIT), March 14-17 2010, Vi a del Mar, Chile, – ISBN 9781424456956

[Man12] MANGEL, T.: Inter-Vehicle Communication at Intersections: An Evaluation of Ad-Hoc and Cellular Communication. Ph.D. Dissertation, Faculty of Informatics, Karlsruhe Institute of Technology, 2012, pp. 11-22 – ISBN 9783866448995

[MB03] MÜLLER, P.; BLESSING, L.: DEVELOPMENT OF PRODUCT-SERVICE-SYSTEMS – COMPARISON OF PRODUCT AND SERVICE DEVELOPMENT PROCESS MODELS. In: Proceedings of 16th International Conference on Engineering Design (ICED), August 28-31 2007, Paris, France, Paper No. DS42_P_547 – ISBN 1904670016

[MDR91] MACCONAILL, P. A.; DREWS, P.; ROBROCK, K.-H. (Eds.): Mechatronics and Robotics I. IOS Press, STM Publishing House, Amsterdam, The Netherlands, January 1991, Vol. 1 – ISBN 9789051990577

[Mey07] MEYWERK, M.: CAE-Methoden in der Fahrzeugtechnik. Springer-Verlag, Berlin, 2007 – ISBN 9783540498667

[MGL+16] MAURER, M.; GERDES, J. C.; LENZ, B.; WINNER; H.: Autonomous driving: Technical, Legal and Social Aspects. 1st edition, Springer-Verlag, Heidelberg, Germany, 2016 – ISBN 9783662488454

[Mic14] MICOUIN, P.: Model-based Systems Engineering: Fundamentals and Methods. 1st edition, John Wiley & Sons, New Jersey, USA, 2014 – ISBN 9781848214699

[Mir06] MIR, N. F.: Computer and Communication Networks. 1st edition, Prentice Hall, Upper Saddle River, New Jersey, USA, 2006 – ISBN 9780131389106

[MMM+11] MAAG, C.; MUHLBACHER, D.; MARK, C.; KRUGER, H.: Studying effects of Advanced Driver Assistance Systems (ADAS) on individual and group level using multi-driver simulation. In: Proceedings of IEEE Intelligent Vehicles Symposium, June 5-9 2011, Baden-Baden, Germany – ISBN 19310587

[MMM+12] MAAG, C.; MUHLBACHER, D.; MARK, C.; KRUGER, H. P.: Studying Effects of Advanced Driver Assistance Systems (ADAS) on Individual and Group Level Using Multi-Driver Simulation. In: IEEE Intelligent Transportation Systems Magazine, August, 2012, Vol. 4, Issue 3, pp. 45-54 – ISSN 19391390

[MS13] MEINEL, C.; SACK, H.: Internetworking: Technological Foundations and Applications. 1st edition, Springer-Verlag, Berlin, Germany, 2013 – ISBN 9783642353918

[MTK+09] MUSTAFEE, N; TAYLOR, S. J. E.; KATSALIAKI, K.; BRAILSFORD, S.: Speeding Up Decision Support: Investigating the Distributed Simulation of a Healthcare Supply Chain. In: KHOUMBATI, K.; DWIVEDI, Y. K.; SRIVASTAVA, A.; LAL, B. (Ed.): Handbook of Research on Advances in Health Informatics and Electronic Healthcare Applications – Global Adoption and Impact of Information Communication Technologies. 1st edition, Medical Information Science Reference, IGI Global, Pennsylvania, USA, 2009, pp. 255-273 – ISBN 9781605660301

[MZ04] MINDERHOUD, M. M.; ZUURBIER, F.: Empirical data on driving behaviour in stop-and-go traffic. In: Proceedings of IEEE Intelligent Vehicles Symposium, June 14-17 2004, Parma, Italy – ISBN 0780383109

[MZM+08] MITTAL, S.; ZEIGLER, B. P.; MARTIN, J. L. R.; SAHIN, F.; JAMSHIDI, M.: Modeling and Simulation for Systems of Systems Engineering. IN: JAMSHIDI, M. (Ed.): System of Systems Engineering: Innovations for the Twenty-first Century. 1st edition, John Wiley & Sons, New Jersey, USA, 2008, pp. 101-149 – ISBN 9780470195901

[NDW09] NEUMANN-COSEL, K.; DUPUIS, M.; WEISS, C.: Virtual Test Drive Provision of a consistent tool-set for [D,H,S,V]-in-the-loop. In: Proceedings of Driving Simulation Conference Europe, February 4-6 2009, Monaco, France – ISBN 9782857826712

[Neg07] NEGELE, J.: Anwendungsgerechte Konzipierung von Fahrsimulatoren für die Fahrzeugentwicklung. Ph.D. Dissertation, Fakultät für Maschinenwesen der Technischen Universität München, 2007

[NHT17-ol] NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION: http://www.nhtsa.gov/, 2017

[Nie95] NIEGEL, W.: Methodical structuring of Knowledge Used in an Intelligent Driving System. In: KOSKINEN, K.; HALME, A. (Ed.): Intelligent Autonomous Vehicles. 1st edition, Elsevier, Amsterdam, The Netherlands, 1995, pp. 381-386 – ISBN 9780080423661

[NRJ+13] NUCKELT, J.; ROSE, D.; JANSEN, T.; KÜRNER, T.: On the use of OpenStreetMap data for V2X channel modeling in urban scenarios. In: Proceedings of 7th European Conference on

Antennas and Propagation (EuCAP), April 8-12 2013, Gothenburg, Sweden – ISBN 9781467321877

[NS13] NIU, D.; SUN, J.: Eco-Driving Versus Green Wave Speed Guidance for Signalized Highway Traffic: A Multi-Vehicle Driving Simulator Study. In: Journal of Social and Behavioral Sciences, November 2013, Vol. 96, pp. 1079-1090 – ISBN 18770428

[OM15] OGITSU, T.; MIZOGUCHI, H.: A study on driver training on advanced driver assistance systems by using a driving simulator. In: Proceedings of International Conference on Connected Vehicles and Expo (ICCVE), October 19-23 2015, Shenzhen, China – e-ISBN 9781509002641

[ORB+06a] OREBAUGH, A.; RAMIREZ, G.; BURKE, J., PESCE, L.; WRIGHT, J.; MORRIS, G.: Introducing Network Analysis. In: OREBAUGH, A.; RAMIREZ, G.; BURKE, J., PESCE, L.; WRIGHT, J.; MORRIS, G. (Ed.): Wireshark & Ethereal Network Protocol Analyzer Toolkit, 2006, ScienceDirect, Elsevier, Amsterdam, The Netherlands, pp. 1-50 – ISBN 9781597490733

[ORB+06b] OREBAUGH, A.; RAMIREZ, G.; BURKE, J., PESCE, L.; WRIGHT, J.; MORRIS, G.: Introducing Wireshark: Network Protocol Analyzer. In: OREBAUGH, A.; RAMIREZ, G.; BURKE, J., PESCE, L.; WRIGHT, J.; MORRIS, G. (Ed.): Wireshark & Ethereal Network Protocol Analyzer Toolkit, 2006, ScienceDirect, Elsevier, Amsterdam, The Netherlands, pp. 51-99 – ISBN 9781597490733

[OS15] OELTZE, K.; SCHIEßL, C.: Benefits and challenges of multi-driver simulator studies. In: IEEE Journal of IET Intelligent Transport Systems, August 2015, Vol. 9, Issue 6, pp. 618-625 – ISSN 1751956X

[Oxf17-ol] OXFORD DICTIONARY OF ENGLISH: http://www.oxforddictionaries.com/, 2017

[PBF+07] PAHL, G.; BEITZ, W.; FELDHUSEN, J.; GROTE, K.-H.: Engineering Design: A Systematic Approach. 3rd edition, Springer-Verlag, Heidelberg, Germany, 2007 – ISBN 9781114243064

[PBT10] PANOU, M. C.; BEKIARIS, E. D.; TOULIOU, A. A.: ADAS module in driving simulation for training young drivers. In: Proceedings of 13th International IEEE Conference on Intelligent Transportation Systems (ITSC), September 2010, Funchal, Portugal – ISBN 21530009

[Pea17] PEARMINE, A.: Connected Vehicle. In: GENG, H. (Ed.) Internet of Things and Data Analytics Handbook. 1st edition, John Wiley & Sons, New Jersey, United States, March 2017, pp. 409-425 – ISBN 9781119173649

[PGZ+13] PRENDINGER, H.; GAJANANAN, K.; ZAKI, A.; FARES, A.; MOLENAAR, R.; ZAKI, A.; URBANO, D.; LINT, H.; GOMAA, W.: Tokyo Virtual Living Lab: Designing Smart Cities Based on the 3D Internet. In: IEEE Internet Computing Magazine, December 2013, Vol. 17, Issue 6, pp. 30-38 – ISSN 10897801

[Pla07] PLANING, P.: A Innovation Acceptance: The Case of Advanced Driver-Assistance Systems. Ph.D. Dissertation, Faculty of Mechanical Engineering, Leeds Metropolitan University, Leeds, UK 2007

[POC+14] PRENDINGER, H.; OLIVEIRA, J.; CATARINO, J.; MADRUGA, M.; PRADA, R.: iCO2 – A Networked Game for Collecting Large-Scale Eco-Driving Behavior Data. In: IEEE Internet Computing Magazine, March 2014, Vol. 18, Issue 3, pp. 28-35 – ISSN 10897801

[Pot11] POTUZAK, T.: Comparison of Road Traffic Network Division Based on Microscopic and Macroscopic Simulation. In: Proceedings of 13th International Conference on Computer Modelling and Simulation (UKSim), April 2011, Cambridge, UK – ISBN 9781612847054

[PRF+05] PECHT, M.; RAMAKRISHNAN, A.; FAZIO, J.; NASH, C.E.: The role of the U.S National Highway Traffic Safety Administration in automotive electronics reliability and safety assessment. In: IEEE Transactions on Components and Packaging Technologies, Vol. 28, Issue 3, August 2005, pp. 571-580 – ISSN 15213331

[PRR10] POPESCU-ZELETIN, R.; RADUSCH, I.; RIGANI, M. A.: Vehicular-2-X Communication: State-of-the-Art and Research in Mobile Vehicular Ad hoc Networks. 1st edition, Springer-Verlag, Berlin, Germany, 2010 – ISBN 9783540771425

[QK14] QUANDT, T; KRÖGER, S.: Introduction – Multiplayer Gaming as Social Media Entertainment. In: QUANDT, T; KRÖGER, S. (Ed.): Multiplayer – The Social Aspects of Digital Gaming. 1st edition, Routledge, Taylor & Francis Group, Abingdon, UK, 2014, pp. 3-9 – ISBN 9780415828857

[Ram02] RAMBHIA, A. M.: XML Distributed Systems Design. 1st edition, Sams Publishing, Indianapolis, Indiana, USA, March 2002 – ISBN 978-0672323287

[RG11] RAILSBACK, S. F., GRIMM, V.: Agent-Based and Individual-Based Modeling: A Practical Introduction. 1st edition, Princeton University Press, Princeton, New Jersey, USA, October 2011 – ISBN 9780691136738

[RMM+15] RITTGER, L.; MÜHLBACHER, D.; MAAG, C.; KIESEL, A.: Anger and bother experience when driving with a traffic light assistant: A multi-driver simulator study. In: Proceedings of Annual Conference of Human Factors and Ergonomics Society Europe Chapter, 2015, Lisbon, Portugal, pp.41-51 – ISSN 23334959

[Rum16] RUMPE, B.: Modeling with UML: Language, Concepts, Methods. 1st edition, Springer International Publishing, Cham, Switzerland, 2016 – ISBN 9783319339320

[RV15] ROYAKKERS, L.; VAN EST R.: Just Ordinary Robots: Automation from Love to War. 1st edition, CRC Press, Taylor and Francis Group, Boca Raton, Florida, United States, 2015, pp. 185-243 – ISBN 9781482260144

[SAE17-ol] SAE SOCIETY OF AUTOMOTIVE ENGINEERS: http://www.sae.org/, 2017

[SBH+13] SIEGFRIED, R.; BERTSCHIK, M.; HAHN, M.; HERRMANN, G.; LÜTHI, J.; ROTHER, M.: Effective and Efficient Training Capabilities through Next Generation Distributed Simulation Environments. In: Proceedings of NATO Modelling & Simulation Group (NMSG) Multi-Workshop, October 17-18 2013, Sydney, Australia, Paper 7, pp. 1-18 – Report ID STO-MP-MSG-111

[SBK13] STEVENS, A.; BRUSQUE, C.; KREMS, J.: Driver Adaptation to Information and Assistance Systems. 1st edition, Institution of Engineering and Technology, London, UK, 2013 – ISBN 9781849196390

[Sch89] SCHWEITZER, G.: Mechatronik-Aufgaben und Lösungen. VDI-Bericht, VDI-Verlag, Düsseldorf, Germany, 1989 – VDI No. 787

[SG16] STANCHEV, P.; GESKE, J.: Autonomous Cars. History. State of Art. Research Problems. In: VISHNEVSKIY, V. M.; KOZYREV, D. V. (Ed.): Distributed Computer and Communication Networks. 1st edition, Springer International Publishing, Switzerland, 2016, pp. 1-10 – ISSN 18650929

[SH12] SAWYER, B.; HANCOCK, P.: Development of a Linked Simulation Network to Evaluate Intelligent Transportation System Vehicle to Vehicle Solutions. In: Proceedings of 56th Annual Meeting of Human Factors and Ergonomics Society, October 22-26 2012, Boston, USA – ISBN 9780945289418

[Sie14] SIEGFRIED, R.: Modeling and Simulation of Complex Systems – A Framework for Efficient Agent-Based Modeling and Simulation. 1st edition, Springer International Publishing, Switzerland, 2014 – ISBN 9783658075286

[SJI+13] SWAKE, J.; JANNAT, M.; ISLAM, M.; HURWITZ, D.: Driver Response to Phase Termination at Signalized Intersections: Are Driving Simulator Results Valid?. In: Proceedings of 7th International Driving Symposium on Human Factors in Driver Assessment, Training, and Vehicle Design, June 17-20 2013, New York, USA – ISBN 9780615819723

[SK10] SHETTY, D.; KOLK, R. A.: Mechatronic System Design. 2nd edition, Cengage Learning, Stamford, USA, 2010, pp. 1-40 – ISBN 9781439061992

[SKH15] SHAH, S. R.; KNOOP, V. L.; HOOGENDOORN, S. P.: Driver Heterogeneity in Rubbernecking Behaviour at an Incident Site. In: CHRAIBI, M.; BOLTES, M.; SCHADSCHNEIDER, A.; SEYFRIED, A. (Ed.): Traffic and Granular Flow '13. 1st edition, Springer International Publishing, Switzerland, 2015, pp. 533-539 – ISBN 9783319106281

[Slo08] SLOB, J. J.: State-of-the-Art Driving Simulators – a Literature Survey, Control Systems Technology Group, Department of Mechanical Engineering, Eindhoven University of Technology, The Netherlands, August 2008 – Internal Report DCT 2008.107

[Slo08] SLOB, J.: State-of-the-Art Driving Simulators – a Literature Survey, Control Systems Technology Group, Department of Mechanical Engineering, Eindhoven University of Technology, The Netherlands, 2008, Internal Report DCT 2008.107

[Ste08] STEPHENS, R.: Introduction to Databases and Database Design. In: STEPHENS, R. (Eds.): Beginning Database Design Solutions. John Wiley & Sons, Hoboken, New Jersey, USA, 2008 – ISBN 9780470385494

[STN+10] STOIANOVICI, V.; TALABA, D.; NEDELCU, A.; PISU, M.; BARBUCEANU, F.; STAVAR, A.: A Virtual Reality based human-network interaction system for 3D internet applications. In: Proceedings of 12th IEEE International Conference on Optimization of Electrical and Electronic Equipment (OPTIM), May 20-22 2010, Brasov, Romania – ISBN 9781424470198

[SZ14] SARBAZI-AZAD, H.; ZOMAYA, A. Y.: Data Distribution Management. 1st edition, Wiley-IEEE Press, Hoboken, New Jersey, USA, 2014 – ISBN 9781118640708

[TBB+95] THOBEN, K.-D.; BERGER, U.; BAUER, J.; SCHMIDT, A.: Virtual prototyping using graphical simulation and advanced programming techniques. In: RIX, J.; HAAS, S.; TEIXEIRA, J. (Ed.): Virtual Prototyping – Virtual environments and the product design process. 1st edition, Springer International Publishing, USA, 1995, pp. 296-312 – ISBN 9780412721601

[TI11] TAN, J.; INAMURA, T.: What are required to simulate interaction with robot? SIGVerse – A simulation platform for human-robot interaction. In: Proceedings of International IEEE Conference on Robotics and Biomimetics (ROBIO), December 7-11 2011, Phuket, Thailand – ISBN 9781457721366

[Tri00] TRIANTAPHYLLOU, E.: Multi-Criteria Decision Making Methods. In: TRIANTAPHYLLOU, E. (Eds.): Multi-criteria Decision Making Methods – A Comparative Study. Springer International Publishing, Switzerland, 2000, Vol. 44, pp. 5-21 – ISBN 9781441948380

[TT15] MA, T.; LI, T.: A multi-driver simulator study Development and application of an integrated traffic simulation and multi-driving simulators. In: Journal of Simulation Modelling Practice and Theory, Vol. 59, December 2015, pp. 1-17 – ISSN 1569190X

[UE08] ULRICH, K. T.; EPPINGER, S. D.: Product design and development. 3rd edition, Irwin/McGraw-Hill, New York, USA, 2008 – ISBN 9780072471465

[VAC+11] VOGEL, O.; ARNOLD, I.; CHUGHTAI, A.; KEHRER, T.: Software Architecture: A Comprehensive Framework and Guide for Practitioners. 1st edition, Springer-Verlag, Berlin, Germany, 2011 – ISBN 9783642197352

[VDI2206] VDI VEREIN DEUTSCHER INGENIEURE: VDI-Guidelines 2206 “Design methodology for mechatronic systems”, Beuth-Verlag, Berlin, June 2004 – ICS 03.100.40; 31.220

[VDI4005] VDI VEREIN DEUTSCHER INGENIEURE: VDI-Guidelines 4005 “Einflüsse von Umweltbedingungen auf die Zuverlässigkeit technischer Erzeugnisse”, VDI-Verlag, Düsseldorf, August 1981 – DK 620.193

[VG12] VAßHOLZ, M.; GAUSEMEIER, J.: Cost-Benefit Analysis – Requirements for the Evaluation of Self-Optimizing Systems. In: Proceedings of the 1st Joint International Symposium on System Integrated Intelligence 2012 – New Challenges for Product and Production Engineering, June 27-29 2012, Hannover, Germany, 2012, pp. 14-16

[Völ12] VÖLKER, L.: Automatisierte Wahl von Kommunikationsprotokollen für das heutige und zukünftige Internet. Ph.D. Dissertation, Karlsruher Institut für Technologie (KIT), KIT Scientific Publishing, 2012 – ISBN 9783866449169

[Vos15] VOSS, W.: A Comprehensible Guide to Controller Area Network. 2nd edition, Copperhill Media Corporation, Greenfield, MA, USA, 2015 – ISBN 9780976511601

[War12] WARE, C.: Foundation for a Science of Data Visualization. IN: WARE, C. (Ed.): Information Visualization: Perception for Design. 3rd edition, Elsevier, Amsterdam, The Netherlands, 2012, pp. 4-27 – ISBN 9780123814647

[Wat09] WATTS, F. B.: Configuration Management Metrics. 1st edition, William Andrew, Elsevier, Norwich, NY, USA, 2009 – ISBN 9781437778342

[WGG10] WEHRLE, K.; GUNES, M.; GROSS, J.: Modeling and Tools for Network Simulation. 1st edition, Springer-Verlag, Heidelberg, Germany, 2010 – ISBN 9783642123306

[WGP11] WALL, M.; GAUSEMEIER, J.; PEITZ, C.: Technology Push-based Product Planning – Future Markets for Emerging Technologies. In: RAUSCHNABEL, P. A. (Ed.): International Journal of Technology Marketing, Inderscience Publishers, Olney, UK, March 2011, Vol. 8, Issue 1, pp. 61-81 – ISSN 1741878X

[WH15] WEI, L.; HAN, L. D.: Impacts of Vehicular Communication Networks on Traffic Dynamics and Traffic Efficiency. In: THOMOPOULOS, N.; GIVONI, M.; RIETVELD, P. (Eds.): ICT for Transport – Opportunities and Threats. Edward Elgar Publishing, Massachusetts, USA, 2015, pp. 161-178 – ISBN 9781783471287

[WH17] WATZENIG, D.; HORN, M.: Introduction to Automated Driving. In: WATZENIG, D; HORN, M. (Ed.): Automated Driving: Safer and More Efficient Future Driving. 1st edition, Springer International Publishing, Switzerland, 2017, pp. 3-16 – ISBN 9783319318936

[WHM+17] WARNERA, J.; HURWITZA, D. S.; MONSEREB, C. M.; FLESKESA, K.: A simulator-based analysis of engineering treatments for right-hook bicycle crashes at signalized intersections. In: Journal of Accident Analysis & Prevention, July 2017, Elsevier, Amsterdam, The Netherlands, Vol. 104, pp. 46-57 – DOI 10.1016/j.aap.2017.04.021

[Wil05] WILKINSON, L.: The Grammar of Graphics. 2nd edition, Springer-Verlag New York, USA, 2005 – ISBN 9780387245447

[WLL13] WANG, J.; LI, K.; LU. X.: Effect of Human Factors on Driver Behavior. In: CHEN Y.; LI, L. (Ed.): Advances in Intelligent Vehicles. 1st edition, Elsevier, Amsterdam, The Netherlands, 2013, pp. 111-157 – ISBN 9780123971999

[WP16] WOLSHON, B.; PANDE, A.: Traffic Engineering Handbook. Institute of Transportation Engineers (ITE), 7th edition, Wiley & Sons, New Jersey, USA, 2016, pp. 217-225 – ISBN 9781118762301

[WRF+15] WALDE, D. D.; ROEDLER, G. J.; FORSBERG, K. J.; HAMELIN, R. D.; SHORTELL, T. M.: INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities. 4th edition, John Wily & Sons, Hoboken, New Jersey, USA, August 2015 – ISBN 9781118999400

[WS08] WELLS, G. D.; SAGE, A. P.: Engineering of a System of Systems. IN: JAMSHIDI, M. (Ed.): System of Systems Engineering: Innovations for the Twenty-first Century. 1st edition, John Wiley & Sons, New Jersey, USA, 2008, pp. 44-76 – ISBN 9780470195901

[WSS15] WALKER, G. H.; STANTON, N. A.; SALMON, P. M.: Human Factors in Automotive Engineering and Technology. 1st edition, Press Taylor and Francis Group, Florida, USA, 2015 – ISBN 9781409447573

[WTB15] WUTHISHUWONG, C.; TRAECHTLER, A.; BRUNS, T: Safe Trajectory Planning for Autonomous Intersection Management by Using Vehicle to Infrastructure Communication. In: Journal on Wireless Communications and Networking, Springer International Publishing, Switzerland, February 2015, Vol. 33, No. 1, pp. 1-12 – ISSN 16871499

[WYX+09] WENGUANG, W.; YONGPINQ, X.; XIN, C.; QUN, L.; WEIPING, W.: High level architecture evolved modular federation object model. In: Journal of Systems Engineering and Electronics, June 2009, Vol. 20, Issue 3, pp. 625-635 – e-ISSN 10044132

[XSC+11] XU, C.; SONG, J.; CHEN, M.; CHEN, J.; YU, L.: Research on Adaptive State Update Strategy of Distributed Interactive Simulation. In: Proceedings of 3rd IEEE International Conference on Multimedia Information Networking and Security (MINES), November 4-6 2011, Shanghai, China – ISBN 9781457717956

[YBP08] YASAR, H.; BERBERS, Y.; PREUVENEERS, D.: A computational analysis of driving variations on distributed multiuser driving simulators. In: Proceedings of IASTED International Conference on Modelling and Simulation, May 26-28 2008, Quebec, Canada, Issue 19, pp. 178-186 – ISBN 9780889867642

[YLH+16] YANG, C.; LEE, T.; HUANG, C.; HSU, K.: Unity 3D production and environmental perception vehicle simulation platform. In: Proceedings of IEEE International Conference on Advanced Materials for Science and Engineering (ICAMSE), November 12-13 2016, Tainan, Taiwan – ISBN 9781509038695

[YWS06] YAMAGUCHI, M.; WAKASUGI, J.; SAKAKIMA, J.: Evaluation of Driver Stress Using Biomarker in Motor-vehicle Driving Simulator. In: Proceedings of 28th Annual International Conference on Medicine and Biology Society (EMBS), September 16-19 2006, New York, USA, pp. 1834-1837 – ISSN 1557170X

[Zwi69] ZWICKY, F.: Discovery, Invention, Research through the Morphological Approach. 1st edition, Macmillan Publishers Ltd, London, UK, 1969 – ISBN 9781114243064

[Zwi89] ZWICKY, F.: Morphologische Forschung – Wesen und Wandel materieller und geistiger struktureller Zusammenhänge. Schriftenreihe der Fritz-Zwicky-Stiftung, Band 4, Verlag Baeschlin, Glarus, 1989 – ISBN 9783813503142

Appendix

Contents

Page

A1 Amendment to the Implemented Prototype (Chapter 5)

In this part of the Appendix, the operation of the implemented prototype of the SoS configuration software is elaborated by presenting additional figures of its graphical user interface and providing hints for system users.

A1.1 Configuration Software – Main Operations

Figure A-1 shows the main start screen of the implemented prototype of the configuration software. This start screen was presented in Subsection 5.1.3 during the introduction of the implantation prototype. It is provided in this part of the Appendix again for the sake of completeness.

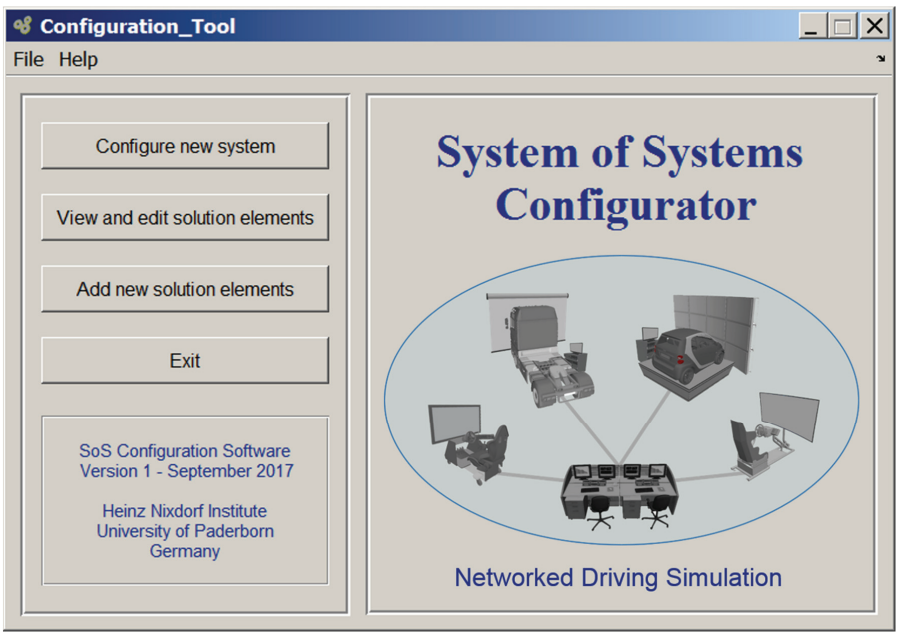


Figure A-1: Start window the SoS configuration software prototype

There are three possible operations at the start of the SoS configuration software: **configure new system**, **view and edit solution elements, and add new solution elements**. These main operations are elaborated in the following sections.

A1.2 Configure New System

The central operation of the SoS configuration software is to configure application-oriented systems for networked driving simulation. To carry out this operation, the system user is guided through particular steps. These steps correspond to the configuration sequence specified in Subsection 4.7.1 and embed the configuration approaches determined in Section 4.6.

Figure A-2 shows a screen shot for the first step, in which the system user defines the scenario mainly by specifying the desired system participants. Specifically, the user determines the number of networked driving simulators and the eventual utilization of a traffic simulator, a workstation, a database console, and a communication architecture in the networked driving simulation system. The SoS configuration software allows a minimum number of two driving simulators. For each driving simulator, the user specifies the driving task, driving response, and the type (ex. passenger car simulator or truck simulator). Consequently, the SoS configuration software determines the application class of each driving simulator. The application classes and the simulator types are added to the selection table upon user's confirmation.

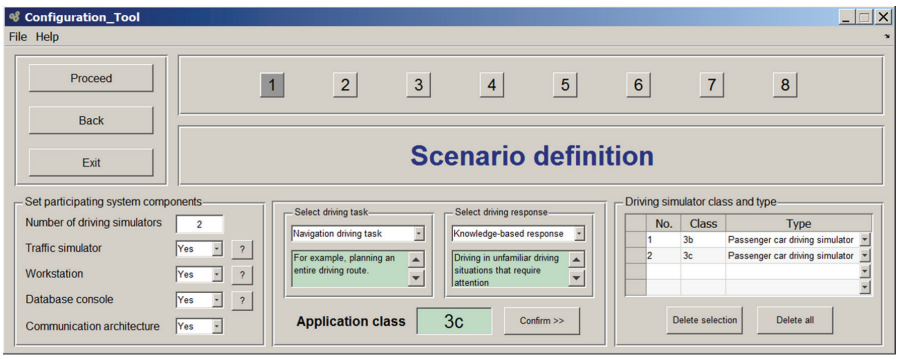


Figure A-2: Scenario definition and determining the utilization of system components

If a traffic simulator is required, the user has to determine some requirements, such as, e.g., granularity level and visualization type. Similarly, the user eventually determines the requirements of the workstation, database console, and communication architecture based on a predetermined requirements table. When all required entries are entered properly, the user is allowed to proceed to the next step.

Figure A-3 shows a screen shot for the second step, in which the SoS configuration software presents the driving simulators that best match the determined application classes. The specification/requirement radar chart of each driving simulator is presented so that the user can assess the matching extent. The user still can choose other driving simulators directly from the database to override the suggestions of the SoS configuration software. Alternatively, the user can also configure new driving simulators. In this case,

the user is guided through the steps of the configuration software discussed in Reference [Has14]. The user is allowed to proceed to the next step after confirming the selections.

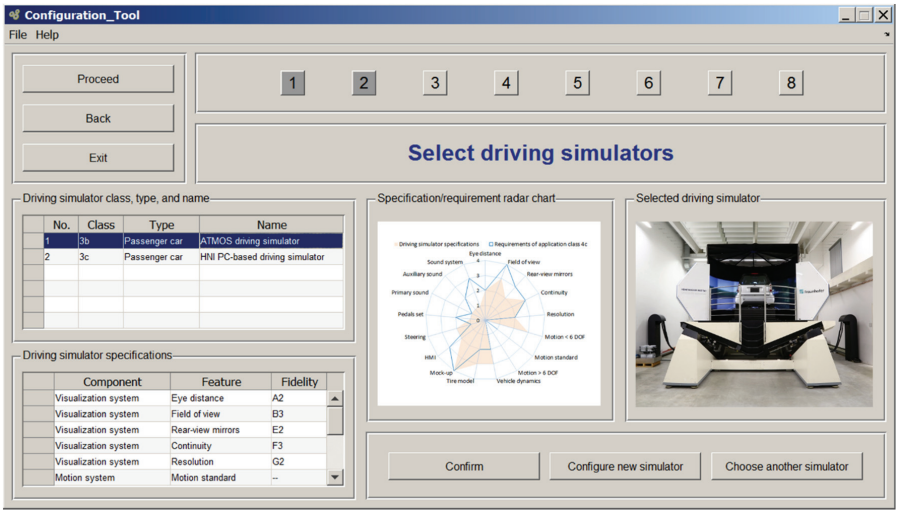


Figure A-3: Selecting the solution elements of the participating driving simulators

Figure A-4 shows a screen shot for the third selection step, in which the SoS configuration software presents the traffic simulators that best match the determined application classes. This selection step is carried out only if the preference to utilize a traffic simulator was indicated during the scenario definition step. The specification/requirement radar chart of each traffic simulator is presented so that the user can assess the matching extent. The user is allowed to select between the suggested traffic simulators or choose other traffic simulators directly from the database to override the suggestions of the SoS configuration software. The user is allowed to proceed to the next step after confirming the selections.

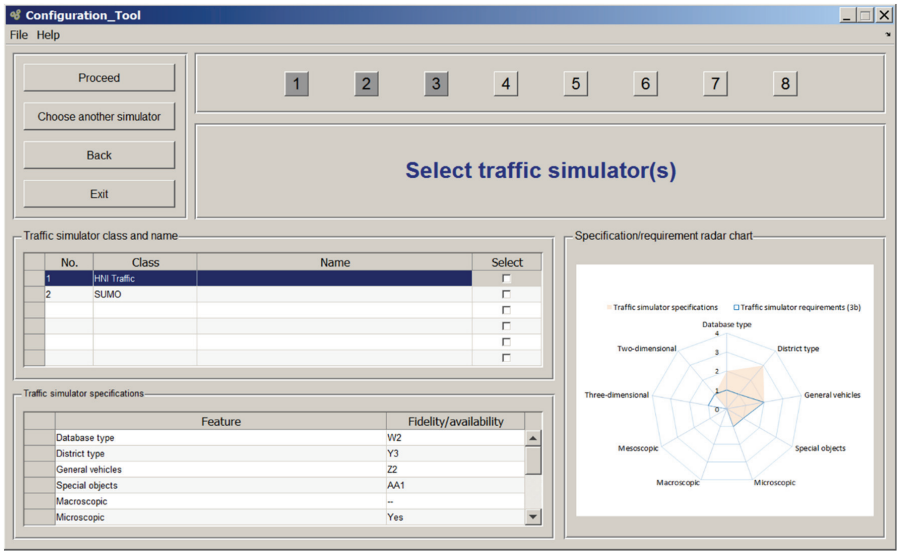


Figure A-4: Selecting the solution elements of the participating traffic simulators

Figure A-5 shows a screen shot for the fourth selection step, in which the SoS configuration software presents a workstation and/or a database console that best match the corresponding requirements. This selection step is carried out only if the preference to utilize a workstation and/or a database console was indicated during the scenario definition step. The specification/requirement comparison tables are presented so that the user can assess the matching extent. The user is allowed to choose another workstation and/or another database console directly from the corresponding database to override the suggestions of the SoS configuration software. The user is allowed to proceed to the next step after confirming the selections.

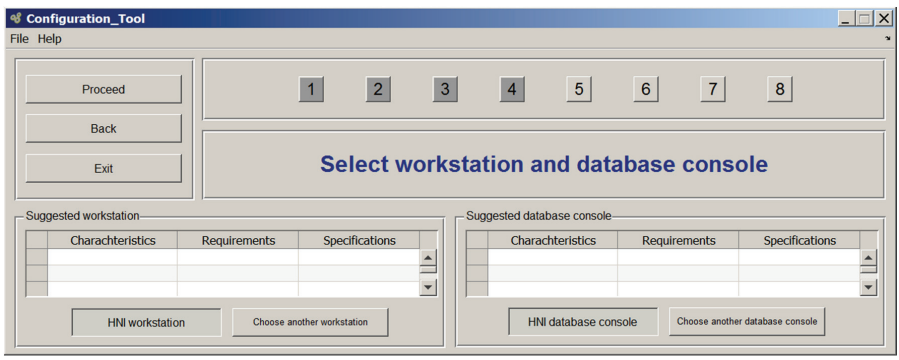


Figure A-5: Selecting the solution elements of the workstation and/or the database console

Figure A-6 shows a screen shot for the fifth selection step, in which the SoS configuration software allows the user to determine the signals sent and/or received by each participating system component. The user is allowed to proceed to the next step after determining the signals.

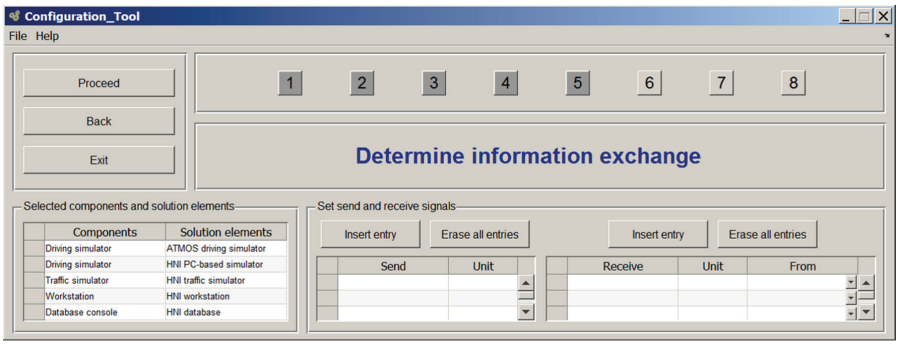


Figure A-6: Specifying the send/receive signals of the participating system components

Figure A-7 shows a screen shot for the sixth selection step, in which the SoS configuration software allows the user to specify and prioritize the network characteristics of the communication technology. Accordingly, the SoS configuration software finds the best matching communication technology. The specification/requirement radar chart of the suggested communication technology is presented so that the user can assess the matching extent. The user is allowed to select another communication technology directly from the database to override the suggestion of the SoS configuration software. The user is allowed to proceed to the next step after confirming the selection.

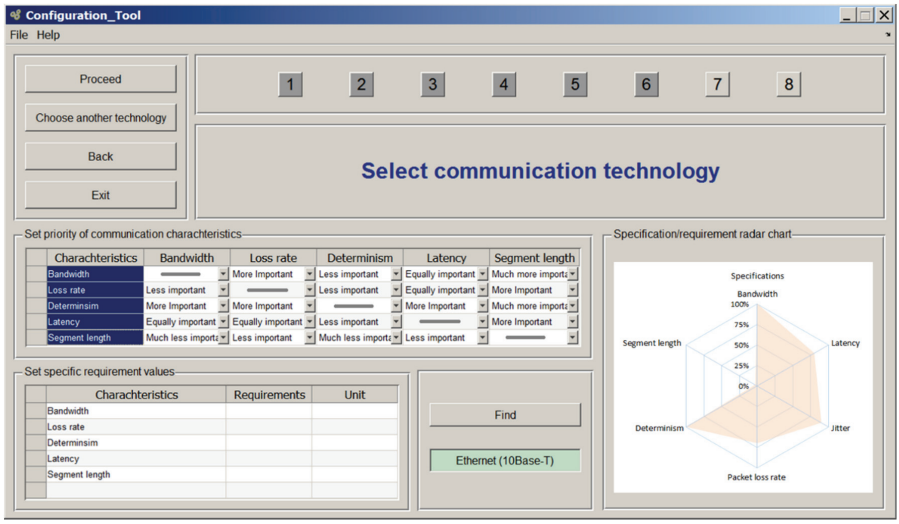


Figure A-7: Specifying and prioritizing the network characteristics of the communication technology

Figure A-8 shows a screen shot for the seventh selection step, in which the SoS configuration software allows the user to specify and prioritize the services and functions of the communication architecture. Accordingly, the SoS configuration software finds the best matching communication architecture. The specification/requirement radar chart of the suggested communication architecture is presented so that the user can assess the matching extent. The user is allowed to select another communication architecture directly from the database to override the suggestion of the SoS configuration software. The user is allowed to proceed to the next step after confirming the selection.

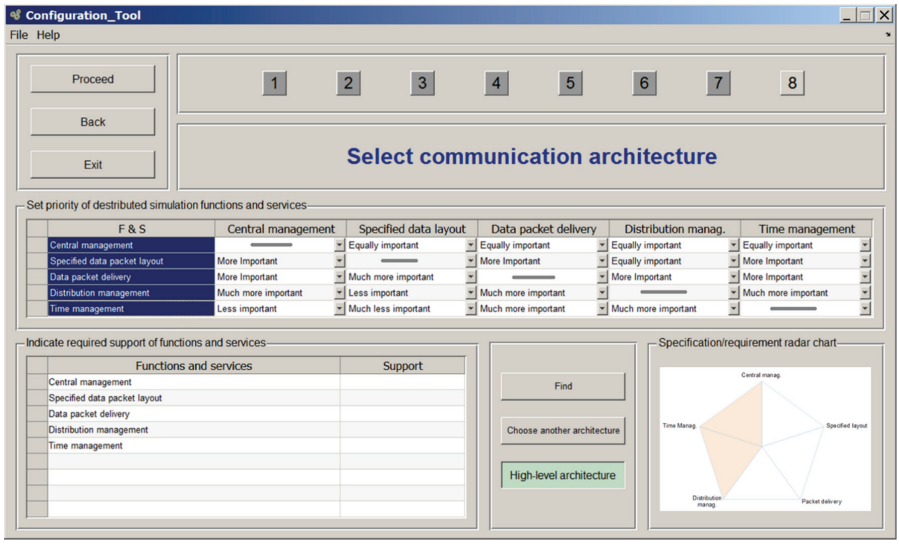


Figure A-8: Specifying and prioritizing the services and functions of the communication architecture

Figure A-9 shows a screen shot for the eighth selection step, in which the SoS configuration software presents the different parts of the generated system model. The user has the option to save and/or print the system model and close the configuration process.

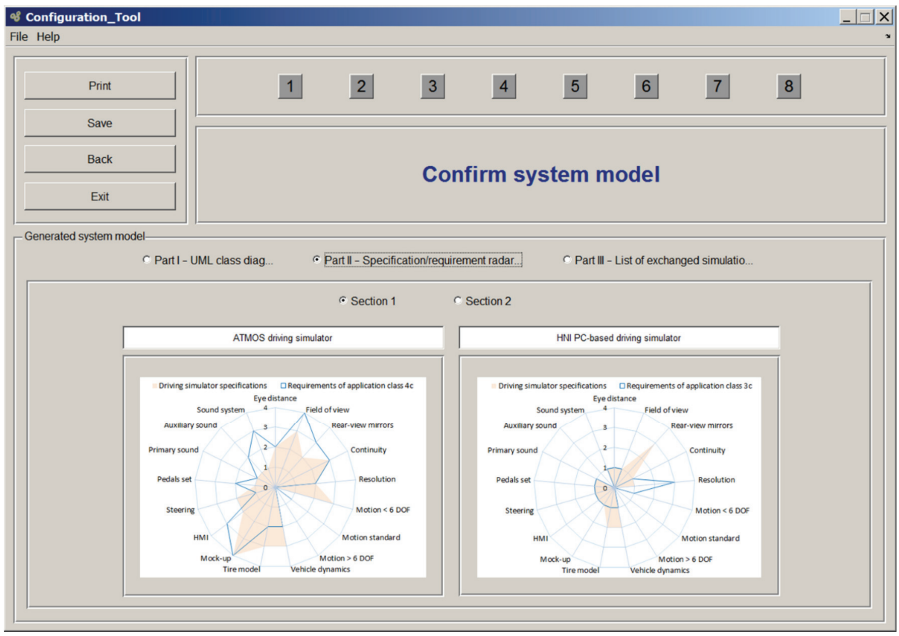


Figure A-9: Confirming the generated system model

The following section presents another operation of the SoS configuration software.

A1.3 View and Edit Solution Elements

The SoS configuration software allows the user to view and edit the entries of the solution elements registered in the system databases that were discussed in Section 4.5. Figure A-10 shows a screen shot for graphical user interface of this operation. The user has to select any particular system component so that the corresponding solution elements are displayed. The specifications are displayed in a table when the user selects a particular solution element. The user can edit the specifications of the solution elements or eventually delete the entries of particular solution elements.

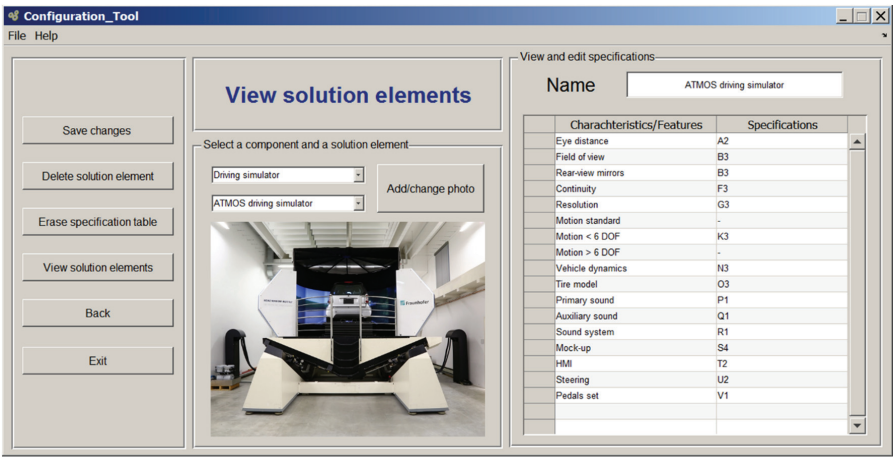


Figure A-10: Viewing and editing the entries of the solution elements registered in the databases

A1.4 Add New Solution Elements

The SoS configuration software allows the user to add entries of new solution elements to the system databases that were discussed in Section 4.5. Figure A-11 shows a screen shot for graphical user interface of this operation. The user has to select a particular system component, to which the solution element should belong. The specifications of the solution element can be entered in a table and saved in the database.

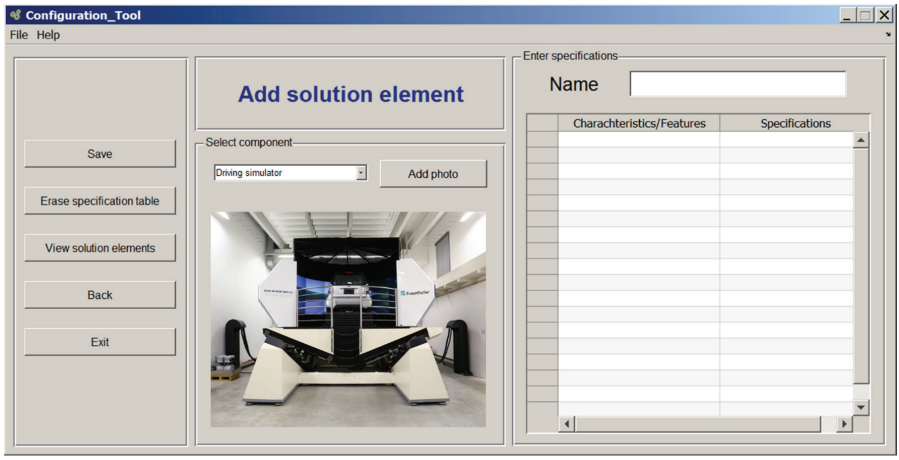


Figure A-11: Adding entries of new solution elements to the system databases

The following part of the Appendix presents tables to facilitate the interpretation of the specification/requirement radar charts that are included within the generated system models.

A2 Amendment to the Validation Examples (Chapter 5)

Three example multi-interactive application scenarios were presented in Section 5.2 in order to validate the design method and its associated SoS configuration software. A motivation and a comprehensive description were provided for each application scenario. Subsequently, a configuration process was conducted based on the defined application scenario and the determined requirements. The configuration process resulted in a concretized system model for networked driving simulation. The system model consists of three different parts: UML class diagram, specification/requirement radar charts, and a list of exchanged simulation data.

In order to facilitate the interpretation of the specification/requirement radar charts, this amendment provides a set of tables summarizing the features and fidelity levels of the individual driving simulator components. These tables were reproduced from NEGELE'S guidelines [Neg07]. The SoS configuration software displays these tables to the system users upon their request.

Table A-1 presents the features and fidelity levels of the visualization system according to NEGELE'S guidelines. A comprehensive discussion about these features and their fidelity levels is presented in Reference [Neg07, pp. 22–38].

Table A-1: Features and fidelity levels of the visualization system [Neg07, pp. 37–38]

Features and fidelity levels		
Feature	**Key**	**Fidelity levels**
Eye distance	A1	Eye distance < 0.8
	A2	Eye distance 2.5–3
	A3	Eye distance > 5
Field of view	B1	40°–60° field of view
	B2	120°–140° field of view
	B3	180°–220° field of view
Rear-view mirrors	E1	Real physical mirrors of the vehicle without modifications
	E2	Small display screens to represent the mirrors
	E3	Virtual mirrors within the front or side display systems
Continuity	F1	Physical vertical boundaries divide the field of view into sectors
	F2	No physical boundaries – virtual separation edges still visible
	F3	No physical or virtual separation edges – overlapping scenes
Resolution	G1	Arc minutes > 6
	G2	Arc minutes 2–3
	G3	Arc minutes < 1

Table A-2 presents the features and fidelity levels of the motion simulation system according to NEGELE'S guidelines. A comprehensive discussion about these features and their fidelity levels is presented in Reference [Neg07, pp. 38–50].

Table A-2: Features and fidelity levels of the motion simulation system [Neg07, pp. 49–50]

Features and fidelity levels		
Feature	**Key**	**Fidelity levels**
Motion standard 6 DOF	L1	Maximum 1.5 Hz, ±25° amplitude, 1.5 ton workload
	L2	Maximum 30 Hz and 20-50 mm amplitude
	L3	Maximum 1.5 Hz and 2.5 ton workload – flying screen
	L4	Maximum 1.5 Hz and > 2.5 ton workload – flying screen
Motion < 6 DOF	K1	1 DOF translation motion, 30 Hz, 20–50 mm
	K2	1 DOF translation, 2 DOF rotational , 1.5 Hz, ±10°
	K3	1 DOF translation + 2 DOF rotational, 1.5 Hz, ±25°
Motion > 6 DOF	M1	6 DOF standard, 1 DOF several meters translation, 1.5 Hz
	M2	6 DOF standard, 2 DOF several meters translation, 1.5 Hz
	M2	6 DOF standard, 2 DOF translation, 1 DOF rotational, 1.5 Hz
Vehicle dynamics	N1	Single-track vehicle model
	N2	Double-track vehicle model
	N3	Multi-body system model
Tire model	Q1	No combined longitudinal and lateral dynamics
	Q2	Combined longitudinal and lateral dynamics
	Q3	Half empirical tire model
	Q4	3D finite-element tire model

Table A-3 presents the features and fidelity levels of the acoustic system according to NEGELE'S guidelines. A comprehensive discussion about these features and their fidelity levels is presented in Reference [Neg07, pp. 50–52].

Table A-3: Features and fidelity levels of the acoustic system [Neg07, p. 52]

Features and fidelity levels		
Feature	**Key**	**Fidelity levels**
Primary sound	P1	Main sound effects of ego-vehicle, e.g., motor sound
	P2	Advanced sound effects, e.g., passing by trees
Auxiliary sound	Q1	Auxiliary sound of ego-vehicle, e.g., turn indicator
	Q2	Sound of traffic vehicles depending on distance to ego-vehicle
Sound system	R1	Two-channel sound system
	R2	Sound system with more than three channels
	R3	Sound system with pressure transducers for bass effects

Table A-4 presents the features and fidelity levels of the driver's platform according to NEGELE'S guidelines. This particular table was given in Subsection 3.1.1 as an example

during the state of the art analysis and evaluation. It is provided in this part of the Appendix again for the sake of completeness. A comprehensive discussion about the included features and their fidelity levels is presented in Reference [Neg07, pp. 52–56].

Table A-4: Features and fidelity levels of the driver's platform [Neg07, p. 56]

Features and fidelity levels		
Feature	**Key**	**Fidelity levels**
Mock-up	S1	Driving seat and HMI without chassis
	S2	Partial vehicle (quarter or half vehicle)
	S3	Complete vehicle – no modifications apparent
	S4	Series production vehicle – no modifications apparent
HMI	T1	Basic and simple HMI
	T2	Complete and realistic HMI
	T3	Complete HMI with reconfigurable display
Steering	U1	Steering moment proportional to steering angle
	U2	Electrical steering moment, damping, and friction
	U3	Electrical steering moment with high frequency
Pedals set	V1	Force feedback passive
	V2	Force feedback adaptive – characteristic curve modifiable
	V3	Force feedback active – effects of control systems tangible

Table A-5 presents the features and fidelity levels of the environment database and the traffic simulation according to NEGELE'S guidelines. A comprehensive discussion about these features and their fidelity levels is presented in Reference [Neg07, pp. 56–59]. In this work, the traffic simulation is considered as a separate task that is independent of the driving simulators in contrast to both NEGELE'S guidelines and HASSAN'S method.

Table A-5: Features and fidelity levels of the environment database and the traffic simulation [Neg07, p. 59]

Features and fidelity levels		
Feature	**Key**	**Fidelity levels**
Database type	W1	Static database (predetermined and not adjustable)
	W2	Modular, static database (modular and configurable offline)
	W3	Dynamic database (modular and configurable during runtime)
District type	Y1	Near simulation field only
	Y2	Far simulation field only
	Y3	Adjustable simulation field
General vehicles	Z1	Randomly generated and follow traffic rules
	Z2	Runtime configurable traffic characteristics
Special objects	AA1	Automatically adjustable behavior based on triggering events
	AA2	Bicyclists, pedestrians, or animals with adjustable behavior
	AA3	Manually controlled traffic vehicles available

Das Heinz Nixdorf Institut – Interdisziplinäres Forschungszentrum für Informatik und Technik

Das Heinz Nixdorf Institut ist ein Forschungszentrum der Universität Paderborn. Es entstand 1987 aus der Initiative und mit Förderung von Heinz Nixdorf. Damit wollte er Ingenieurwissenschaften und Informatik zusammenführen, um wesentliche Impulse für neue Produkte und Dienstleistungen zu erzeugen. Dies schließt auch die Wechselwirkungen mit dem gesellschaftlichen Umfeld ein.

Die Forschungsarbeit orientiert sich an dem Programm „Dynamik, Mobilität, Vernetzung: Eine neue Schule des Entwurfs der technischen Systeme von morgen“. In der Lehre engagiert sich das Heinz Nixdorf Institut in Studiengängen der Informatik, der Ingenieurwissenschaften und der Wirtschaftswissenschaften.

Heute wirken am Heinz Nixdorf Institut neun Professoren mit insgesamt 150 Mitarbeiterinnen und Mitarbeitern. Pro Jahr promovieren hier etwa 20 Nachwuchswissenschaftlerinnen und Nachwuchswissenschaftler.

Heinz Nixdorf Institute – Interdisciplinary Research Centre for Computer Science and Technology

The Heinz Nixdorf Institute is a research centre within the University of Paderborn. It was founded in 1987 initiated and supported by Heinz Nixdorf. By doing so he wanted to create a symbiosis of computer science and engineering in order to provide critical impetus for new products and services. This includes interactions with the social environment.

Our research is aligned with the program “Dynamics, Mobility, Integration: Enroute to the technical systems of tomorrow.” In training and education the Heinz Nixdorf Institute is involved in many programs of study at the University of Paderborn. The superior goal in education and training is to communicate competencies that are critical in tomorrows economy.

Today nine Professors and 150 researchers work at the Heinz Nixdorf Institute. Per year approximately 20 young researchers receive a doctorate.

Zuletzt erschienene Bände der Verlagsschriftenreihe des Heinz Nixdorf Instituts

Bd. 352 WALL, M.: Systematik zur technologieinduzierten Produkt- und Technologieplanung. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 352, Paderborn, 2016 – ISBN 978-3-942647-71-7

Bd. 353 CORD-LANDWEHR, A.: Selfish Network Creation - On Variants of Network Creation Games. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 353, Paderborn, 2016 – ISBN 978-3-942647-72-4

Bd. 354 ANACKER, H.: Instrumentarium für einen lösungsmusterbasierten Entwurf fortgeschrittener mechatronischer Systeme. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 354, Paderborn, 2016 – ISBN 978-3-942647-73-1

Bd. 355 RUDTSCH, V.: Methodik zur Bewertung von Produktionssystemen in der frühen Entwicklungsphase. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 355, Paderborn, 2016 – ISBN 978-3-942647-74-8

Bd. 356 SÖLLNER, C.: Methode zur Planung eines zukunftsfähigen Produktportfolios. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 356, Paderborn, 2016 – ISBN 978-3-942647-75-5

Bd. 357 AMSHOFF, B.: Systematik zur musterbasierten Entwicklung technologieinduzierter Geschäftsmodelle. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 357, Paderborn, 2016 – ISBN 978-3-942647-76-2

Bd. 358 LÖFFLER, A.: Entwicklung einer modellbasierten In-the-Loop-Testumgebung für Waschautomaten. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 358, Paderborn, 2016 – ISBN 978-3-942647-77-9

Bd. 359 LEHNER, A.: Systematik zur lösungsmusterbasierten Entwicklung von Frugal Innovations. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 359, Paderborn, 2016 – ISBN 978-3-942647-78-6

Bd. 360 GAUSEMEIER, J. (Hrsg.): Vorausschau und Technologieplanung. 12. Symposium für Vorausschau und Technologieplanung, Heinz Nixdorf Institut, 8. und 9. Dezember 2016, Berlin-Brandenburgische Akademie der Wissenschaften, Berlin, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 360, Paderborn, 2016 – ISBN 978-3-942647-79-3

Bd. 361 PETER, S.: Systematik zur Antizipation von Stakeholder-Reaktionen. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 361, Paderborn, 2016 – ISBN 978-3-942647-80-9

Bd. 362 ECHTERHOFF, O.: Systematik zur Erarbeitung modellbasierter Entwicklungsaufträge. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 362, Paderborn, 2016 – ISBN 978-3-942647-81-6

Bd. 363 TSCHIRNER, C.: Rahmenwerk zur Integration des modellbasierten Systems Engineering in die Produktentstehung mechatronischer Systeme. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 363, Paderborn, 2016 – ISBN 978-3-942647-82-3

Bd. 364 KNOOP, S.: Flachheitsbasierte Positionsregelungen für Parallelkinematiken am Beispiel eines hochdynamischen hydraulischen Hexapoden. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 364, Paderborn, 2016 – ISBN 978-3-942647-83-0

Bd. 365 KLIEWE, D.: Entwurfssystematik für den präventiven Schutz Intelligenter Technischer Systeme vor Produktpiraterie. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 365, Paderborn, 2017 – ISBN 978-3-942647-84-7

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Zuletzt erschienene Bände der Verlagsschriftenreihe des Heinz Nixdorf Instituts

Bd. 366 IWANEK, P.: Systematik zur Steigerung der Intelligenz mechatronischer Systeme im Maschinen- und Anlagenbau. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 366, Paderborn, 2017 – ISBN 978-3-942647-85-4

Bd. 367 SCHWEERS, C.: Adaptive Sigma-Punkte-Filter-Auslegung zur Zustands- und Parameterschätzung an Black-Box-Modellen. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 367, Paderborn, 2017 – ISBN 978-3-942647-86-1

Bd. 368 SCHIERBAUM, T.: Systematik zur Kostenbewertung im Systementwurf mechatronischer Systeme in der Technologie Molded Interconnect Devices (MID). Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 368, Paderborn, 2017 – ISBN 978-3-942647-87-8

Bd. 369 BODDEN, E.; DRESSLER, F.; DUMITRESCU, R.; GAUSEMEIER, J.; MEYER AUF DER HEIDE, F.; SCHEYTT, C.; TRÄCHTLER, A. (Hrsg.): Intelligente technische Systeme. Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 369, Paderborn, 2017 – ISBN 978-3-942647-88-5

Bd. 370 KÜHN, A.: Systematik zur Release-Planung intelligenter technischer Systeme. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 370, Paderborn, 2017 – ISBN 978-3-942647-89-2

Bd. 371 REINOLD, P.: Integrierte, selbstoptimierende Fahrdynamikregelung mit Einzelradaktorik. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 371, Paderborn, 2017 – ISBN 978-3-942647-90-8

Bd. 372 BÄUMER, F. S.: Indikatorbasierte Erkennung und Kompensation von ungenauen und unvollständig beschriebenen Softwareanforderungen. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 372, Paderborn, 2017 – ISBN 978-3-942647-91-5

Bd. 373 ECKELT, D.: Systematik zum innovationsorientierten Intellectual Property Management. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 373, Paderborn, 2017 – ISBN 978-3-942647-92-2

Bd. 374 GAUSEMEIER, J. (Hrsg.): Vorausschau und Technologieplanung. 13. Symposium für Vorausschau und Technologieplanung, Heinz Nixdorf Institut, 23. und 24. November 2017, Berlin-Brandenburgische Akademie der Wissenschaften, Berlin, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 374, Paderborn, 2017 – ISBN 978-3-942647-93-9

Bd. 375 WESTERMANN, T.: Systematik zur Reifegradmodell-basierten Planung von Cyber-Physical Systems des Maschinen- und Anlagenbaus. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 375, Paderborn, 2017 – ISBN 978-3-942647-94-6

Bd. 376 JÜRGENHAKE, C.: Systematik für eine prototypenbasierte Entwicklung mechatronischer Systeme in der Technologie MID (Molded Interconnect Devices). Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 376, Paderborn, 2017 – ISBN 978-3-942647-95-3

Bd. 377 WEBER, J.: Modellbasierte Werkstück- und Werkzeugpositionierung zur Reduzierung der Zykluszeit in NC-Programmen. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 377, Paderborn, 2018 – ISBN 978-3-942647-96-0

Bd. 378 OESTERSÖTEBIER, F.: Modellbasierter Entwurf intelligenter mechatronischer Systeme mithilfe semantischer Technologien. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, Verlagsschriftenreihe des Heinz Nixdorf Instituts, Band 378, Paderborn, 2018 – ISBN 978-3-942647-97-7

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn